

LNCs 13468

Agata Ciabattoni
Elaine Pimentel
Ruy J.G.B. de Queiroz (Eds.)

Logic, Language, Information, and Computation

28th International Workshop, WoLLIC 2022
Iași, Romania, September 20–23, 2022
Proceedings

 Springer



Founding Editors

Gerhard Goos, Germany
Juris Hartmanis, USA

Editorial Board Members

Elisa Bertino, USA
Wen Gao, China

Bernhard Steffen , Germany
Moti Yung , USA

FoLLI Publications on Logic, Language and Information

Subline of Lectures Notes in Computer Science

Subline Editors-in-Chief

Valentin Goranko, *Stockholm University, Sweden*
Michael Moortgat, *Utrecht University, The Netherlands*

Subline Area Editors

Nick Bezhanishvili, *University of Amsterdam, The Netherlands*
Anuj Dawar, *University of Cambridge, UK*
Philippe de Groote, *Inria Nancy, France*
Gerhard Jäger, *University of Tübingen, Germany*
Fenrong Liu, *Tsinghua University, Beijing, China*
Eric Pacuit, *University of Maryland, USA*
Ruy de Queiroz, *Universidade Federal de Pernambuco, Brazil*
Ram Ramanujam, *Institute of Mathematical Sciences, Chennai, India*

More information about this series at <https://link.springer.com/bookseries/558>

Agata Ciabattoni · Elaine Pimentel ·
Ruy J.G.B. de Queiroz (Eds.)

Logic, Language, Information, and Computation

28th International Workshop, WoLLIC 2022
Iași, Romania, September 20–23, 2022
Proceedings

Editors

Agata Ciabattoni 
Technische Universität Wien
Vienna, Austria

Elaine Pimentel 
University College London
London, UK

Ruy J.G.B. de Queiroz 
Universidade Federal de Pernambuco
Recife, Pernambuco, Brazil

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-031-15297-9 ISBN 978-3-031-15298-6 (eBook)
<https://doi.org/10.1007/978-3-031-15298-6>

© The Editor(s) (if applicable) and The Author(s), under exclusive license
to Springer Nature Switzerland AG 2022

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors, and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

This volume contains the papers presented at the 28th Workshop on Logic, Language, Information and Computation (WoLLIC 2022) held during September 20–23, 2022 at the Faculty of Computer Science, Alexandru Ioan Cuza University in Iasi, Romania. The WoLLIC series of workshops started in 1994 with the aim of fostering interdisciplinary research in pure and applied logic. The idea is to have a forum which is large enough in terms of the number of possible interactions between logic and the sciences related to information and computation, and yet small enough to allow for concrete and useful interaction among participants.

For WOLLIC 2022 there were 46 submissions. Each submission was reviewed by at least two Program Committee members. The committee decided to accept 25 papers. This volume includes all the accepted papers, together with the abstracts of the invited speakers at WOLLIC 2022:

- Anupam Das (University of Birmingham, UK),
- John Harty (University of Maryland, USA),
- Marie Kerjean (Université Sorbonne Paris Nord/CNRS, France),
- Dorel Lucanu (Alexandru Ioan Cuza University, Romania), and
- Francesca Poggiolesi (Université Sorbonne/CNRS/IHPST, France).

It also includes abstracts of the invited tutorials given by

- Gabriel Ciobanu (Romanian Academy, Romania),
- Sonia Marin (University of Birmingham, UK), and
- Marija Slavkovic (University of Bergen, Norway).

We would like to thank all the people who contributed to making WOLLIC 2022 a success. We thank the Program Committee and all additional reviewers for the work they put into reviewing the submissions. We thank the invited speakers and the tutorial presenters for their inspiring talks, the Steering Committee and the Advisory Committee for their advice, and the Local Organizing Committee members (especially Ștefan Ciobâcă) for their great support. Finally, we thank all the authors for their excellent contributions.

The help provided by the EasyChair system created by Andrei Voronkov is gratefully acknowledged. We also would like to acknowledge the scientific sponsorship of the following organizations: the Interest Group in Pure and Applied Logics (IGPL), the Association for Logic, Language and Information (FoLLI), the Association for Symbolic Logic (ASL), the European Association for Theoretical Computer Science

(EATCS), the European Association for Computer Science Logic (EACSL), and the Brazilian Logic Society (SBL).

September 2022

Agata Ciabattoni
Elaine Pimentel
Ruy de Queiroz

Organization

Program Committee Chairs

Agata Ciabattoni	Technische Universität Wien, Austria
Elaine Pimentel	University College London, UK

Steering Committee

Samson Abramsky	University College London, UK
Anuj Dawar	University of Cambridge, UK
Juliette Kennedy	University of Helsinki, Finland
Ulrich Kohlenbach	Technische Universität Darmstadt, Germany
Daniel Leivant	Indiana University Bloomington, USA
Leonid Libkin	University of Edinburgh, UK
Lawrence Moss	Indiana University Bloomington, USA
Luke Ong	University of Oxford, UK
Valeria de Paiva	Topos Institute/PUC-Rio, USA/Brazil
Ruy de Queiroz	Universidade Federal de Pernambuco, Brazil
Alexandra Silva	Cornell University, USA
Renata Wassermann	Universidade de São Paulo, Brazil

Program Committee

Arthur Azevedo de Amorim	Boston University, USA
Diana Costa	University of Lisbon, Portugal
Hans van Ditmarsch	Open Universiteit, The Netherlands
Rajeev Goré	Polish Academy of Sciences, Poland
Roman Kuznets	Technische Universität Wien, Austria
João Marcos	Federal University of Rio Grande do Norte, Brazil
Lawrence Moss	Indiana University Bloomington, USA
Valeria de Paiva	Topos Institute/PUC-Rio, USA/Brazil
Revantha Ramanayake	University of Groningen, The Netherlands
Mehrnoosh Sadzadeh	University College London, UK
Alexandra Silva	Cornell University, USA
Alex Simpson	University of Ljubljana, Slovenia
Sonja Smets	University of Amsterdam, The Netherlands
Alwen Tiu	Australian National University, Australia
Leon van der Torre	University of Luxembourg, Luxembourg
Andrea Aler Tubella	Umeå University, Sweden
Andrés Villaveces	Universidad Nacional de Colombia, Colombia
Renata Wassermann	University of São Paulo, Brazil

Additional Reviewers

Henrique Antunes	Federal University of Bahia, Brazil
James Brotherston	University College London, UK
Francicleber M. Ferreira	Federal University of Ceará, Brazil
Daniel Figueiredo	University of Aveiro, Portugal
Robert Freiman	Technische Universität Wien, Austria
Jakub Gajarský	University of Warsaw, Poland
Wesley Holliday	University of California, Berkeley, USA
Peter Jipsen	Chapman University, USA
Louwe B. Kuijer	University of Liverpool, UK
Timo Lang	University College London, UK
Giacomo Lenzi	University of Salerno, Italy
Xavier Parent	Technische Universität Wien, Austria
Todd Schmid	University College London, UK
Stephan Schulz	DHBW Stuttgart, Germany
Ian Shillito	Australian National University, Australia
Julian Sutherland	Imperial College London, UK
Sebastiaan Terwijn	Radboud University Nijmegen, The Netherlands
Apostolos Tzimoulis	Vrije Universiteit Amsterdam, The Netherlands
Daniel Ventura	University of Goiás, Brazil
Amanda Vidal	Spanish National Research Council, Spain

Invited Talks

On the Proof Theoretic Strength of Circular Reasoning

Anupam Das

University of Birmingham, UK
a.das@bham.ac.uk

Cyclic and *non-wellfounded* proofs are now a common technique for demonstrating metalogical properties of systems incorporating (co)induction, including modal logics, predicate logics, type systems and algebras. Unlike usual proofs, non-wellfounded proofs may have infinite branches: they are generated *coinductively* from a set of inference rules. Naturally, such ‘proofs’ may admit fallacious reasoning, and so one typically employs some global correctness condition inspired by ω -automaton theory.

A key motivation in cyclic proof theory is the so-called ‘Brotherston-Simpson conjecture’: are cyclic proofs and inductive proofs equally powerful? Naturally, the answer depends on how one interprets ‘equally powerful’, e.g. as provability, proof complexity, logical complexity etc., as well as on the logic at hand. In any case it is interesting to note that the tools employed in cyclic proof theory are often bespoke to the underlying logic, yielding a now myriad of techniques at the interface between several branches of mathematical and computational logic.

In this talk I will discuss a line of work that attempts to understand the expressivity of cyclic proofs via forms of proof theoretic strength. Namely, I address predicate logic in the guise of first-order arithmetic, and type systems in the guise of higher-order primitive recursion, and establish a recurring theme: circular reasoning buys precisely one level of ‘abstraction’ over inductive reasoning. Along the way we shall see some of the aforementioned interplays in action, in particular exploiting techniques from proof theory, reverse mathematics, automaton theory, metamathematics, rewriting theory and higher-order computability.

Open Texture and Defeasible Semantic Constraint

John Horty

University of Maryland, USA
horty@umd.edu

The concept of open-texture was introduced in [8], with its importance for legal theory noted shortly afterward in [2]. Due to the intrinsic interest and practical importance of the issues surrounding open-textured predicates, a substantial literature on the topic has evolved within legal theory; some highlights include [1, 5–7]. For the most part, however, this literature focuses on broader issues in the theory of open texture—the role of defeasible legal rules, policy arguments concerning the application of these rules, the impact of open-textured predicates on theories of legal interpretation, connections with philosophy of language very generally. The literature does not provide anything like a semantic theory of open-textured predicates. In this talk, I will attempt to supply such a theory.

The talk has four parts. In the first, I will review some of the problems presented by open-textured predicates, and suggest an explication of the concept according to which: the predicate P is open-textured just in case, given any description of an object a on the basis of which we can reasonably apply P to a , it is always possible consistently to extend this description in such a way that it is no longer reasonable to apply P to a . In the second, I will sketch an account of constraint in the common law presented in my own recent work [3, 4]. In the third part, I will show how this account can be adapted to help us understand open-textured predicates as well. Finally, in the fourth part, I will talk a bit about the reasoning involved in reaching decisions that satisfy the account of constraint, and show how this reasoning can be modeled in a simple defeasible logic.

References

1. Baker, G.: Defeasibility and meaning. In: Hacker, P., Raz, J. (eds.) *Law, Morality, and Society: Essays in Honour of H. L. A Hart*, pp. 26–57. Oxford University Press (1977)
2. Hart, H.L.A.: Positivism and the separation of law and morals. *Harvard Law Rev.* **71**, 593–629 (1958)
3. Horty, J.: Rules and reasons in the theory of precedent. *Legal Theor.* **17**, 1–33 (2011)
4. Horty, J.: Constraint and freedom in the common law. *Philosopher's Imprint* **15**(25) (2015)
5. MacCormick, N.: On open texture in law. In: Amslek, P., MacCormick, N. (eds.) *Controversies about Law's Ontology*, pp. 72–83. Edinburgh University Press (1991)

6. Schauer, F.: A critical guide to vehicles in the park. *New York University Law Review*, **83**, 1109–1134 (2008)
7. Tur, R.: Defeasibilism. *Oxford J. Legal Stud.* **21**, 355–368 (2001)
8. Waismann, F.: Verifiability. In: *Proceedings of the Aristotelian Society, Supplementary*. vol. 19, pp. 119–150. Harrison and Son (1945), reprinted in Anthony Flew (eds.) *Logic and Language*, pp. 117–144, Blackwell Publishing Company, 1951; pagination refers to this version.

∂ is for Dialectica

Marie Kerjean^{1,2} and Pierre-Marie Pédro³

¹ CNRS

² Université Sorbonne Paris Nord

³ Inria, France

kerjean@lipn.fr, pedrot@inria.fr

Dialectica was originally introduced by Gödel in a famous paper [7] as a way to constructively interpret an extension of $\mathbf{HA}$ [1], but turned out to be a very fertile object of its own. Judged too complex, it was quickly simplified by Kreisel into the well-known realizability interpretation that now bears his name. Soon after the inception of Linear Logic (LL), Dialectica was shown to factorize through Girard's embedding of LJ into LL, purveying an expressive technique to build categorical models of LL [13]. In its logical outfit, Dialectica led to numerous applications and was tweaked into an unending array of variations in the proof mining community [10].

The modern way to look at Dialectica is however to consider it as a program translation, or more precisely *two* mutually defined translations of the λ -calculus exposing intensional information [14].

In a different scientific universe, Automatic Differentiation [8] (AD) is the field that studies the design and implementation of *efficient* algorithms computing the differentiation of mathematical expression and numerical programs. Indeed, due to the chain rule, computing the differential of a sequence of expressions involves a choice, namely when to compute the value of a given expression and when to compute the value of its derivative. Two extremal algorithms coexist. On the one hand, forward differentiation [16] computes functions and their derivatives pairwise in the order they are provided, while on the other hand reverse differentiation [12] computes all functions first and then their derivative in reverse order. Depending on the setting, one can behave more efficiently than the other. Notably, reverse differentiation has been critically used in the fashionable context of deep learning.

Differentiable programming is a rather new and lively research domain aiming at expressing automatic differentiation techniques through the prism of the traditional tools of the programming language theory community. As such, it has been studied through continuations [15], functoriality [6], and linear types [4]. It led to a myriad of implementation over rich programming languages, proven correct through semantics of higher-order differentiable functions [11]. Surprisingly, these various principled explorations of automatic differentiation are what allows us to draw a link between Dialectica and differentiation in logic.

The simple, albeit fundamental claim of this talk is that, behind its different logical avatars, the Dialectica translation is in fact a reverse differentiation algorithm, where the linearity and involutivity of differentiation have been forgotten. In the domain of proof theory, differentiation has been very much studied from the point of view of

linear logic. This led to Differential Linear Logic [5] (DiLL), differential categories [3], or the differential λ -calculus. To support our thesis with evidence, we will formally state a correspondence between each of these objects and the corresponding Dialectica interpretation.

More generally, Dialectica is known for extracting *quantitative* information from proofs [10], and this relates very much with the quantitative point of view that differentiation has brought to λ -calculus [2]. Herbelin also notices at the end of its paper realizing Markov’s rule through delimited continuations that this axiom has the type of a differentiation operator [9]. If time permits, we will explore the possible consequences of formally relating reverse differentiation and Dialectica to proof mining and Herbelin’s work in the conclusion.

References

1. Avigad, J., Feferman, S.: Gödel’s functional (‘dialectica’) interpretation. In: Buss, S.R. (eds.) *Handbook of Proof Theory*. Elsevier Science Publishers (1998)
2. Barbarossa, D., Manzonetto, G.: Taylor subsumes scott, berry, kahn and plotkin. *Proc. ACM Program. Lang.* **4**(POPL), 1:1–1:23 (2020)
3. Blute, R.F., Cockett, J.R.B., Seely, R.A.G.: Differential categories. *Math. Structures Comput. Sci.* **16**(6) (2006)
4. Brunel, A., Mazza, D., Pagani, M.: Backpropagation in the simply typed lambda-calculus with linear negation. *POPL* (2020)
5. Ehrhard, T., Regnier, L.: Differential interaction nets. *Theor. Comput. Sci.* **364**(2) (2006)
6. Elliott, C.: The simple essence of automatic differentiation. *Proc. ACM Program. Lang.* (ICFP) (2018)
7. Gödel, K.: Über eine bisher noch nicht benützte Erweiterung des finiten Standpunktes. *Dialectica* **12**, 280–287 (1958)
8. Griewank, A., Walther, A.: *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. Society for Industrial and Applied Mathematics, USA, second edn. (2008)
9. Herbelin, H.: An intuitionistic logic that proves markov’s principle. *LICS* (2010)
10. Kohlenbach, U.: *Applied Proof Theory - Proof Interpretations and their Use in Mathematics*. Springer Monographs in Mathematics, Springer, Heidelberg (2008). DOI: <https://doi.org/10.1007/978-3-540-77533-1>
11. Krawiec, F., Peyton Jones, S., Krishnaswami, N., Ellis, T., Eisenberg, R.A., Fitzgibbon, A.: Provably correct, asymptotically efficient, higher-order reverse-mode automatic differentiation. *Proc. ACM Program. Lang.* **6**(POPL) (jan 2022)
12. Linnainmaa, S.: Taylor expansion of the accumulated rounding error. *BIT Numerical Mathematics* **16**, 146–160 (1976)
13. de Paiva, V.: A dialectica-like model of linear logic. In: *Category Theory and Computer Science* (1989)
14. Pédrot, P.: A functional functional interpretation. *CSL-LICS ‘14*, Vienna, Austria, July 14–18 (2014)
15. Wang, F., Zheng, D., Decker, J., Wu, X., Essertel, G.M., Rompf, T.: Demystifying differentiable programming: Shift/reset the penultimate backpropagator. **3**(ICFP) (2019)
16. Wengert, R.E.: A simple automatic derivative evaluation program. *Commun. ACM* **7**(8), 463–464 (1964)

How to Define Domain Specific Logics using Matching Logic

Dorel Lucanu

Alexandru Ioan Cuza University, Iași, Romania
dorel.lucanu@gmail.com

Matching logic [2–4] is a logic that allows to uniformly specify and reason about programming languages and properties of their programs. The syntax of matching logic is simple and compact:

$$\varphi ::= x \mid X \mid \sigma \mid \varphi_1 \varphi_2 \mid \perp \mid \varphi_1 \rightarrow \varphi_2 \mid \exists \cdot x \varphi \mid \mu X \cdot \varphi$$

These eighth syntax constructs build matching logic formulas, called *patterns*, which, semantically speaking, can be matched by a set of elements. Patterns can match structures that are of certain shapes, satisfy certain dynamic properties, or meet certain logical constraints, usually all of these together.

The matching logic is endowed with a proof system that defines the provability relation, written $\Gamma \vdash_{ML} \varphi$, which means that φ is formally derivable from the axioms in Γ , using the matching logic (Hilbert-style) proof system [2].

Many important logics and/or formal systems have been shown to be definable in matching logic as logical theories. In this we consider a different approach: starting from a matching logic theory specifying a domain D , we derive a logic (proof system) $\vdash_D$ that can be used independently to reason within D .

Next we present two matching logic theories: DEF and NAT. DEF introduces a new symbol **def**, called the *definedness* symbol, and defines the (Definedness) axiom. This symbol and its axioms is all it is needed to define predicates, its possible values being $\perp$ or $T \equiv \neg \perp$. Then, the equality, the inclusion, and the membership are introduced as notations for patterns using the new symbol.

theory DEF	theory NAT Imports: DEF
Symbols: def	Symbols: $\mathbb{N}, zero, succ$
Notations: $[\varphi] \equiv \mathbf{def} \ \varphi$	Notations:
Axioms: (Definedness) $\forall x. [x]$	$0 \equiv zero, 1 \equiv succ \ 0, 2 \equiv succ \ 1, \dots$
Notations:	$\forall x:\mathbb{N}. \varphi \equiv x \in \mathbb{N} \rightarrow \varphi$
(totality) $[\varphi] \equiv \neg[\neg\varphi]$	$\exists x:\mathbb{N}. \varphi \equiv x \in \mathbb{N} \wedge \varphi$
(equality) $\varphi_1 = \varphi_2 \equiv [\varphi_1 \leftrightarrow \varphi_2]$	Axioms:
(inclusion) $\varphi_1 \subseteq \varphi_2 \equiv [\varphi_1 \rightarrow \varphi_2]$	(Zero) $\exists x:\mathbb{N}. zero = x$
(membership) $x \in \varphi \equiv x \subseteq \varphi$	(Succ) $\forall x:\mathbb{N}. \exists y:\mathbb{N}. succ \ x = y$
endtheory	(Succ.1) $succ \ zero \neq zero$
	(Succ.2) $\forall x:\mathbb{N}. \forall y:\mathbb{N}. succ \ x = succ \ y \rightarrow x=y$
	(Domain) $\mathbb{N} = \mu D. zero \vee succ \ D$
	endtheory

The theory **NAT** specifies the natural numbers up to an isomorphism [1]. Note the 1–1 correspondence between the **NAT** axioms and the Peano axioms (see, e.g., <https://www.britannica.com/science/Peano-axioms>).

From the theory **DEF** we may derive the following the following inference system that can be used to reason about the equality and the membership:

$$\begin{array}{c}
\frac{}{\vdash_{\text{DEF}} \varphi = \varphi} \qquad \frac{\vdash_{\text{DEF}} \varphi}{\vdash_{\text{DEF}} \forall x. x \in \varphi} x \notin FV(\varphi) \\
\frac{}{\vdash_{\text{DEF}} x \in y = (x = y)} \qquad \frac{\vdash_{\text{DEF}} \varphi_1 = \varphi_2 \wedge \psi[\varphi_1/x] \rightarrow \psi[\varphi_2/x]}{\vdash_{\text{DEF}} \varphi_1 = \varphi_2 \wedge \psi[\varphi_1/x] \rightarrow \psi[\varphi_2/x]} \\
\frac{}{\vdash_{\text{DEF}} \forall x. x \in \varphi} x \notin FV(\varphi) \qquad \frac{\vdash_{\text{DEF}} \varphi_1 = \varphi_2}{\vdash_{\text{DEF}} \varphi_2 = \varphi_1} \\
\frac{}{\vdash_{\text{DEF}} x \in \neg \varphi = \neg(x \in \varphi)} \qquad \frac{\vdash_{\text{DEF}} (x \in \varphi_1 \wedge \varphi_2) = (x \in \varphi_1) \wedge (x \in \varphi_2)}{\vdash_{\text{DEF}} (x \in \varphi_1 \wedge \varphi_2) = (x \in \varphi_1) \wedge (x \in \varphi_2)} \\
\frac{}{\vdash_{\text{DEF}} \varphi_1 = \varphi_2} \qquad \frac{\vdash_{\text{DEF}} \varphi_1 = \varphi_2 \quad \vdash_{\text{DEF}} \varphi_2 = \varphi_3}{\vdash_{\text{DEF}} \varphi_1 = \varphi_3}
\end{array}$$

The derived inference system for **NAT** imports $\vdash_{\text{DEF}}$ (the first rule), includes the axioms of **NAT** as rules (the next four rules), and rules for inductive reasoning (the last three rules), obtained using the (PreFixPoint) and (Knaster-Tarski) from the matching logic proof system [2]:

$$\begin{array}{c}
\frac{\vdash_{\text{DEF}} \varphi}{\vdash_{\text{NAT}} \varphi} \qquad \frac{}{\vdash_{\text{NAT}} 0 \in \mathbb{N}} \qquad \frac{\vdash_{\text{NAT}} \varphi \in \mathbb{N}}{\text{NAT} \vdash \text{succ } \varphi \in \mathbb{N}} \\
\frac{}{\vdash_{\text{NAT}} \text{succ } 0 \neq 0} \qquad \frac{}{\vdash_{\text{NAT}} \forall x:\mathbb{N}. \forall y:\mathbb{N}. \text{succ } x = \text{succ } y} \\
\frac{\vdash_{\text{NAT}} \text{succ } \varphi \rightarrow \varphi}{\vdash_{\text{NAT}} \forall x:\mathbb{N}. (x \in \varphi \rightarrow \text{succ } x \in \varphi)} \qquad \frac{\vdash_{\text{NAT}} \forall x:\mathbb{N}. (x \in \varphi \rightarrow \text{succ } x \in \varphi)}{\vdash_{\text{NAT}} \text{succ } \varphi \rightarrow \varphi} \\
\frac{\vdash_{\text{NAT}} \text{zero} \rightarrow \varphi \quad \vdash_{\text{NAT}} \text{succ } \varphi \rightarrow \varphi}{\vdash_{\text{NAT}} \mathbb{N} \rightarrow \varphi}
\end{array}$$

We obviously have $\vdash_{\text{DEF}} \varphi$ implies $\text{DEF} \vdash_{\text{ML}} \varphi$ and $\vdash_{\text{NAT}} \varphi$ implies $\text{DEF} \vdash_{\text{ML}} \varphi$.

We start with a gentle introduction of matching logic, including its proof system, and then we use several canonical examples of domains specified in matching logic to show how we can derive their specific logics. These examples will involve both the inductive and coinductive reasoning.

References

1. Chen, X., Lucanu, D., Roşu, G.: Initial algebra semantics in matching logic. Technical Report. University of Illinois at Urbana-Champaign (July 2020). <http://hdl.handle.net/2142/107781>, submitted

2. Chen, X., Lucanu, D., Roşu, G.: Matching logic explained. *J. Log. Algeb. Meth. Program.* **120**, 100638 (2021)
3. Chen, X., Roşu, G.: Matching mu-logic. In: *Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2019) (To appear)* (2019)
4. Roşu, G.: Matching logic. *Log. Meth. Comput. Sci.* **13**(4), 1–61 (2017)

The Value of Normal Derivations in the Realm of Explanations

Francesca Poggiolesi

IHPST, UMR 8590 CNRS, Université Paris 1 Panthéon-Sorbonne

Abstract. The concept of explanation is and has long been the object of deep and wide philosophical debates; in particular it is the notion of causal explanation that has for decades dominated the general attention, e.g. see [11]. Beside the debate on causal explanation, in recent years another type of explanation has gained attention, namely *mathematical explanation*. The expression *mathematical explanations* is an umbrella term that indicates several different phenomena; in this context, we use it to refer to those mathematical proofs that not only show the theorem they prove to be true, but that they also reveal the *reasons* why the theorem is true. The idea that certain mathematical proofs have an explanatory power has been shown to be widespread amongst mathematicians (e.g. see [4]) and to have a long and illustrious philosophical pedigree (e.g. see [3] and [9]). Moreover it is a type of mathematical explanations that has been having a central role in the recent literature on the subject. *To date there has been a tendency to approach the topic of mathematical explanations by investigating the distinction between explanatory and non-explanatory proofs. This is very natural since it is widely acknowledged that some proofs are explanatory whilst other are not.* [1, p. 3]

In the attempt of better understanding mathematical explanatory proofs, some scholars have drawn an analogy with normalized derivations in natural deduction calculi, e.g. see [2, 8]. This analogy rests on a feature that both mathematical explanatory proofs and normalized derivations share, namely a complexity's increase from the assumptions to the conclusion of proofs/derivations. On the one hand, one of the main features of explanatory proofs amounts to the fact that they explain the theorem they prove by providing grounds or reasons that are simpler than the theorem they prove. On the other hand, normalized proofs typically satisfy the subformula property¹ and the subformula property can be seen as the formalization of this idea of complexity's increase from the premisses to the conclusion (e.g. see [10]). Although, for several reasons,² normalized derivations cannot be considered as a proper formalization of explanatory mathematical proofs, they nevertheless represent a first step towards this direction.

In this talk, the main aim is to deepen the analysis on the relationships between mathematical explanatory proofs and normal derivations; we will do that by proposing a novel model for mathematical explanations according to which when a mathematical proof is (thought of as) explanatory, then there

¹ At least under certain conditions.

² E.g. see [6].

exists a way to formalize it with a normal derivation where the undischarged assumptions are less complex than the conclusion. This modeling of explanatory proofs will involve the use of theorems or mathematical definitions (that occur in the mathematical proof in key positions) as rules of the derivation (e.g. see [5]), as well as the extension of the notion of logical complexity to the level of concepts (e.g. see [7]).

References

1. Baron, S., Colyvan, M., Ripley, D.: A counterfactual approach to explanation in mathematics. *Philos. Math.* **28**, 1–34 (2020)
2. Casari, E.: Matematica e verità. *Rivista di Filosofia*, **78**(3), 329–350 (1987)
3. Kitcher, P.: Explanatory unification. *Philos. Science*, **48**(4), 507–531 (1981) URL: <http://www.jstor.org/stable/186834>, doi:10.2307/186834
4. Mancosu, P.: Mathematical explanation: problems and prospects. *Topoi*, **20**(1), 97–117 (2001). URL: <http://link.springer.com/article/10.1023/A3A1010621314372>,
5. Marin, S., Miller, D., Pimentel, E., Volpe, M.: Synthetic inference rules for geometric theories. **1**, 1–28 (2021). Submitted
6. Poggiolesi, F.: On defining the notion of complete and immediate grounding. *Synthese*, **193**, 3147–3167 (2016)
7. Poggiolesi, F.: Explanations in mathematics: an analysis via formal proofs and conceptual complexity, 1–34 (2022). Submitted
8. Rumberg, A.: Bolzano’s theory of grounding against the background of normal proofs. *Rev. Symbol. Log.* **6**(3), 424–459 (2013)
9. Steiner, M.: Mathematical explanation. *Philos. Stud.*, **34**(2), 135–151 (1978). URL: <http://link.springer.com/article/10.1007/BF00354494>,
10. Troelstra, A.S., Schwichtenberg, H.: *Basic Proof Theory*. Cambridge University Press (1996)
11. Woodward, J.: Scientific explanation. In: Edward, N., Zalta, (eds.). *The Stanford Encyclopedia of Philosophy* (2017)

Tutorials

Cardinalities, Infinities and Choice Principles for Finitely Supported Sets

Andrei Alexandru¹ and Gabriel Ciobanu^{1,2}

¹ Romanian Academy, Institute of Computer Science, Iași
andrei.alexandru@iit.academiaromana-is.ro

² Alexandru Ioan Cuza University, Iași, Romania
gabriel@info.uaic.ro

Finitely supported sets are standard sets equipped with actions of a group of permutations of some basic elements (atoms) whose internal structure is ignored, sets satisfying a finite support requirement. They allow a discrete (finitary) representation of possibly infinite structures containing enough symmetries to be concisely handled. The results presented in this tutorial deal with three topics:

Results Regarding Choice. The choice principles **HP** (Hausdorff maximal principle), **ZL** (Zorn lemma), **DC** (principle of dependent choice), **CC** (principle of countable choice), **PCC** (principle of partial countable choice), **AC(fin)** (axiom of choice for finite sets), **Fin** (principle of Dedekind finiteness), **PIT** (prime ideal theorem), **UFT** (ultrafilter theorem), **OP** (total ordering principle), **KW** (Kinna-Wagner selection principle), **SIP** (principle of existence of right inverses for surjections), **FPE** (finite powerset equipollence principle) and **GCH** (generalized continuum hypothesis) fail in the framework of finitely supported sets.

Results Regarding Cardinalities. Two finitely supported sets X and Y have the same cardinality (i.e., $|X| = |Y|$) if and only if there exists a finitely supported bijection $f : X \rightarrow Y$. While some arithmetic properties of cardinalities (regarding sums, products and exponents) are naturally translated from the non-atomic framework, there are also some specific atomic properties. Let us consider:

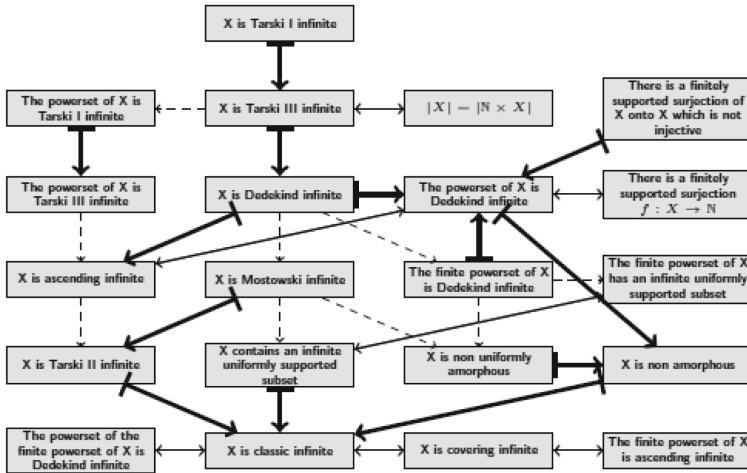
- $\leq$ given by $|X| \leq |Y|$ iff there is a finitely supported injection $f : X \rightarrow Y$;
- $\leq^*$ given by $|X| \leq^* |Y|$ iff there is a finitely supported surjection $g : Y \rightarrow X$.

We prove that the relation $\leq$ is equivariant, reflexive, anti-symmetric and transitive, but it is not total, while the relation $\leq^*$ is equivariant, reflexive and transitive, but it is not anti-symmetric, nor total.

Results Regarding Infinities. We present relationships between various pairwise non-equivalent forms of infinity defined below. Let X be a finitely supported set.

1. X is called *classic infinite* if it can be represented in the form $\{x_1, \dots, x_n\}$.
2. X is *covering infinite* if there is a finitely supported directed family $\mathcal{F}$ of finitely supported sets with the property that X is contained in the union of the members of $\mathcal{F}$, but there does not exist $Z \in \mathcal{F}$ such that $X \subseteq Z$.
3. X is called *Tarski I infinite (TI i)* if $|X| = |X \times X|$.

4. X is called *Tarski II infinite (TII i)* if there exists a finitely supported totally ordered family of finitely supported subsets of X having no maximal element.
5. X is called *Tarski III infinite (TIII i)* if $|X| = |X + X|$.
6. X is called *Mostowski infinite (M i)* if there exists an infinite finitely supported totally ordered subset of X .
7. X is called *Dedekind infinite (D i)* if there exists a finitely supported one-to-one mapping of X onto a finitely supported proper subset of X (or equivalently, iff there exists a finitely supported one-to-one mapping $f : \mathbb{N} \rightarrow X$).
8. X is called *ascending infinite (Asc i)* if there is a finitely supported increasing countable chain of finitely supported sets $X_0 \subseteq X_1 \subseteq \dots \subseteq X_n \subseteq \dots$ with $X \subseteq \bigcup X_n$, but there does not exist $n \in \mathbb{N}$ such that $X \subseteq X_n$.
9. X is called *non-amorphous (N-am)* if X contains two disjoint, infinite, finitely supported subsets.



Relationships between several forms of infinity; the ‘ultra thick arrows’ indicate strict implications, the ‘thin dashed arrows’ indicate implications for which we did not prove yet if they are strict or not, and the ‘thick bidirectional arrows’ indicate equivalences.

Examples of finitely supported sets satisfying various forms of infinity:

Set	TI i	TIII i	D i	M i	Asc i	TII i	N-am
A (the set of all atoms)	No	No	No	No	No	No	No
nA , A^n , $n > 1$	No	No	No	No	No	No	Yes
$\wp_{fs}(X)$, X not D i	No	No	No	No	Yes	Yes	Yes
$X_{fs}^{A^n}$, X not D i	No	No	No	No	Yes	Yes	Yes
$A \cup \mathbb{N}$	No	No	Yes	Yes	Yes	Yes	Yes
$A \times \mathbb{N}$, $\wp_{fs}(A \cup \mathbb{N})$	No	Yes	Yes	Yes	Yes	Yes	Yes
$A_{fs}^{\mathbb{N}}$, $\mathbb{N}_{fs}^A$, $\wp_{fs}(\wp_{fs}(\wp_{fs}(A)))$	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Reference

1. Alexandru, A., Ciobanu, G.: Foundations of finitely supported structures. A Set Theoretical Viewpoint. Springer, Cham (2020). DOI: <https://doi.org/10.1007/978-3-030-52962-8>

Intuitionistic Modal Proof Theory

Sonia Marin

University of Birmingham, UK
s.marin@bham.ac.uk

Intuitionistic modal logic, despite more than seventy years of investigation [4], still partly escapes our comprehension.

Already answering what is the intuitionistic variant of normal modal logic K is not obvious. Lacking De Morgan duality, there are several variants of the normal K axiom that are classically but not intuitionistically equivalent. Five axioms have been considered as primitives in the literature. An intuitionistic variant of K can then be obtained from intuitionistic propositional logic IPL by

- adding the *necessitation rule*: $\Box A$ is a theorem if A is a theorem; and
- adding a subset of the following five axioms:

$$\begin{array}{ll} k_1: \Box(A \rightarrow B) \rightarrow (\Box A \rightarrow \Box B) & k_3: \Diamond(A \vee B) \rightarrow (\Diamond A \vee \Diamond B) \\ k_2: \Box(A \rightarrow B) \rightarrow (\Diamond A \rightarrow \Diamond B) & k_4: (\Diamond A \rightarrow \Box B) \rightarrow \Box(A \rightarrow B) \\ & k_5: \Diamond \perp \rightarrow \perp \end{array}$$

Structural proof theoretic accounts of intuitionistic modal logic have adopted either the paradigm of *labelled deduction* in the form of labelled natural deduction and sequent systems [6], or the one of *unlabelled deduction* in the form of sequent [2] or nested sequent systems [1, 7]. In this tutorial, we would like to give an overview of the current landscape of intuitionistic modal proof theory and illustrate how “old and new” approaches can complement each other.

We will review ordinary sequent calculi, which are adequate to treat logics based on a subset of k_1 , k_2 , k_3 and k_5 , as well as *labelled* and *nested sequents*, which have been used to give deductive systems for the logics that cannot seem to be handled in ordinary sequent calculi, i.e., the ones that include k_4 .

Both of these approaches (labelled and unlabelled) are still under active investigation. A framework for fragments of intuitionistic modal logics was recently designed, based on unlabelled sequents but related to a new intuitionistic version of *neighbourhood semantics* [3]. Another one proposes a refined labelled approach taking full advantage of the more standard *birelational semantics* [5]. As these allow for a fine-grained account of intuitionistic modal logic, we hope they will help shed some light on the intricate world of intuitionistic modal logics.

References

1. Arisaka, R., Das, A., Straßburger, L.: On nested sequents for constructive modal logics. *Log. Methods Comput. Sci.* **11**(3) (2015)
2. Bierman, G.M., de Paiva, V.: Intuitionistic necessity revisited. School of Computer Sciences-University of Birmingham (1996). Technical Report
3. Dalmonte, T., Grellois, C., Olivetti, N.: Intuitionistic non-normal modal logics: a general framework. *J. Philos. Log.* **49**(5), 833–882 (2020)
4. Fitch, F.B.: Intuitionistic modal logic with quantifiers **7**(2), 113–118 (1948)
5. Marin, S., Morales, M., Straßburger, L.: A fully labelled proof system for intuitionistic modal logics. *J. Log. Comput.* **31**(3), 998–1022 (2021)
6. Simpson, A.: The Proof Theory and Semantics of Intuitionistic Modal Logic. Ph.D. Thesis, University of Edinburgh (1994)
7. Straßburger, L.: Cut elimination in nested sequents for intuitionistic modal logics. In: Pfenning, F. (ed.) *Foundations of Software Science and Computation Structures. FoSSaCS 2013. Lecture Notes in Computer Science*, vol. 7794. Springer, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37075-5_14

Automating Moral Reasoning

Marija Slavkovik 

University of Bergen, Norway
marija.slavkovik@uib.no

Abstract. Machine ethics has, as its topic of research, the behaviour of machines towards humans and other machines. One aspect of that research problem is enabling machines to reason about right and wrong. The automation of moral reasoning is on one end the field of dreams and speculative fiction, but on the other it is a very real need to ensure that the artificial intelligence used to automate various tasks that require intelligence does not neglect the ethical and value impact this ‘replacement’ of man with machine has. This tutorial introduces the problem of making moral decisions and gives a general overview of how a computational agent can be constructed to make moral decisions.

Keywords: Machine ethics · Artificial morality · Computational agency

What is Machine Ethics? Artificial intelligence (AI) is concerned with the problem of using computation to automate tasks that require intelligence [3]. In a society, we all affect each other with our activities and decisions. Ethics (or moral philosophy) is concerned with understanding and recommending right and wrong behaviours and decisions [6]. The right decisions being characterised by taking into consideration not only ones own interest, but also the interest of others [7]. The more computationally automated tasks are used to complement or replace people’s tasks, the more concerns we have to ensure that the resulting actions and choices are not only correct and rational, but also do not have a negative ethical impact on society.

One way to ensure that AI has a non-negative ethical impact on society is to consider that moral reasoning is itself a cognitive task that we can consider automating. Machine ethics, or artificial morality, is a sub-field in AI that is researching this approach. The problem of automating moral reasoning can be considered as a problem of moral philosophy, whereas one is interested in questions such as: should machines be enabled with ethical reasoning [5], which norms should machines follow [8], can machines ever be moral agents [4], etc. As a problem of computer science, machine ethics focuses on the question of how to automate moral reasoning [2, 9].

Here we are concerned with the question of how to automate moral reasoning. Although this problem, and machine ethics in general, have been raised since 2006 [1], it is an extremely difficult problem that requires a lot of improvement in the state of the art in AI and moral philosophy. We discuss the basic approaches in machine ethics, the advantages and challenges of each. These lecture notes are structured as follows.

Tutorial Overview. In this tutorial, first we discuss what is decision making and how decision-making is distinguished from moral decision-making. Decisions are made by an agent. Next we discuss what computational agents are, what does it mean for a computational agent to be autonomous and what kind of moral agents can computational agents be. One way to automate moral reasoning is to follow a specific moral theory. We give a very quick overview of what is a moral theory and some of the more known moral theories from moral philosophy. Next, In we discuss two general approaches to building artificial moral agents, we discuss open research problems and challenges.

Tutorial Scope. In this tutorial we do discussed specific examples of artificial moral agents. This tutorial is not intended to be a systematic review of implemented machine ethics systems. A very practical reason for avoiding discussing implementations of artificial agents here is that these implementations vary vastly in the approaches they use and considerable background knowledge in various reasoning and learning methods would be necessary to understand the implementations. However, references are given to these specific systems and the interested reader can follow them and explore them for learning more.

References

1. Anderson, M., Anderson, S.L.: The status of machine ethics: a report from the AAAI symposium. *Minds Mach.*, **17**(1), 1–10 (2007). doi:[10.1007/s11023-007-9053-7](https://doi.org/10.1007/s11023-007-9053-7)
2. Anderson, M., Anderson, S.L., et al.: *Machine Ethics*. Cambridge University Press (2011)
3. Bellman, R.E.: *An Introduction to Artificial Intelligence: Can Computers Think?* Boyd & Fraser Publishing Company (1978)
4. Brożek, B., Janik, B.: Can artificial intelligences be moral agents? *New Ideas in Psychology*, **54**, 101–106 (2019). URL: <https://www.sciencedirect.com/science/article/pii/S0732118X17300739>, doi:<https://doi.org/10.1016/j.newideapsych.2018.12.002>
5. Etzioni, A., Etzioni, O.: Incorporating ethics into artificial intelligence. *J. Ethics*, **21**, 403–418, (2017)
6. Fieser, J.: Ethics. In: Boylan, M., (eds.) *Internet Encyclopedia of Philosophy*. ISSN 2161-0002, (2021)
7. Hare, R.M.: *Community and Communication*, pp. 109–115. Macmillan Education UK, London, (1972). doi:[10.1007/978-1-349-00955-8_9](https://doi.org/10.1007/978-1-349-00955-8_9)
8. Malle, B.F., Bello, P., Scheutz, M.: Requirements for an artificial agent with norm competence. In: *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society, AIES 2019*, pp. 21–27, New York, NY, USA. Association for Computing Machinery (2019). doi:[10.1145/3306618.3314252](https://doi.org/10.1145/3306618.3314252).
9. Wallach, W., Allen, C.: *Moral machines: teaching robots right from wrong*. Oxford University Press, Inc., USA (2008)

Contents

A Proof of the Focusing Theorem via MALL Proof Nets	1
<i>Roberto Maieli</i>	
Time and Gödel: Fuzzy Temporal Reasoning in PSPACE	18
<i>Juan Pablo Aguilera, Martín Diéguez, David Fernández-Duque, and Brett McLean</i>	
Fixed Point Logics and Definable Topological Properties	36
<i>David Fernández-Duque and Quentin Gougeon</i>	
Correspondence Theory for Generalized Modal Algebras	53
<i>Zhiguang Zhao</i>	
Tense Logics over Lattices	70
<i>Xiaoyang Wang and Yanjing Wang</i>	
Expressing Power of Elementary Quantum Recursion Schemes for Quantum Logarithmic-Time Computability	88
<i>Tomoyuki Yamakami</i>	
Multityped Abstract Categorical Grammars and Their Composition	105
<i>Pierre Ludmann, Sylvain Pogodalla, and Philippe de Groote</i>	
Interval Probability for Sessions Types	123
<i>Bogdan Aman and Gabriel Ciobanu</i>	
Combinatorial Flows as Bicolored Atomic Flows	141
<i>Giti Omidvar and Lutz Straßburger</i>	
A Logic of “Black Box” Classifier Systems	158
<i>Xinghan Liu and Emiliano Lorini</i>	
What Kinds of Connectives Cause the Difference Between Intuitionistic Predicate Logic and the Logic of Constant Domains?	175
<i>Naosuke Matsuda and Kento Takagi</i>	
Logic of Visibility in Social Networks	190
<i>Rustam Galimullin, Mina Young Pedersen, and Marija Slavkovik</i>	
The Alternation Hierarchy of the μ -calculus over Weakly Transitive Frames	207
<i>Leonardo Pacheco and Kazuyuki Tanaka</i>	

Embedding Kozen-Tiuryn Logic into Residuated One-Sorted Kleene Algebra with Tests	221
<i>Igor Sedlár and Johann J. Wannenburg</i>	
The Limits to Gossip: Second-Order Shared Knowledge of All Secrets is Unsatisfiable	237
<i>Hans van Ditmarsch and Malvin Gattinger</i>	
Additive Types in Quantitative Type Theory	250
<i>Vít Šefl and Tomáš Svoboda</i>	
Strongly First Order, Domain Independent Dependencies: The Union-Closed Case	263
<i>Pietro Galliani</i>	
Towards an Intuitionistic Deontic Logic Tolerating Conflicting Obligations	280
<i>Tiziano Dalmonete, Charles Grellois, and Nicola Olivetti</i>	
Presburger Büchi Tree Automata with Applications to Logics with Expressive Counting.	295
<i>Bartosz Bednarczyk and Oskar Fiuk</i>	
Abstract Cyclic Proofs.	309
<i>Bahareh Afshari and Dominik Wehr</i>	
Subordination Algebras as Semantic Environment of Input/Output Logic	326
<i>Andrea De Domenico, Ali Farjami, Krishna Manoorkar, Alessandra Palmigiano, Mattia Panettiere, and Xiaolong Wang</i>	
Material Dialogues for First-Order Logic in Constructive Type Theory	344
<i>Dominik Wehr and Dominik Kirst</i>	
On the Computational Properties of the Uncountability of the Real Numbers	362
<i>Sam Sanders</i>	
Mining the Surface: Witnessing the Low Complexity Theorems of Arithmetic	378
<i>Amirhossein Akbar Tabatabai</i>	
Non-monotonic Reasoning via Dynamic Consequence	395
<i>Carlos Areces, Valentin Cassano, and Raul Fervari</i>	
Author Index	411



A Proof of the Focusing Theorem via MALL Proof Nets

Roberto Maieli^(✉) 

Department of Mathematics and Physics, “Roma TRE” University, Rome, Italy

maieli@uniroma3.it

<http://logica.uniroma3.it/maieli>

Abstract. We present a demonstration of Andreoli’s *focusing theorem* for proofs of linear logic (MALL) that avoids directly reasoning on sequent calculus proofs. Following Andreoli-Maieli’s strategy, exploited in the MLL case, we prove the focusing theorem as a particular sequentialization strategy for MALL proof nets that are in *canonical form*. Canonical proof nets satisfy the property that asynchronous links are always ready to sequentialization while synchronous focusing links represent clusters of links that are hereditarily ready to sequentialization.

Keywords: linear logic · sequent calculus · focusing proofs · proof nets

1 Introduction

Focusing is an efficient proof-search procedure for Linear Logic [4], based on a proof normalization result (the “Focusing Theorem”) that has been described by Andreoli in [1]. Focusing is described there in terms of the sequent system of (commutative) Linear Logic, which it refines in two steps: “Dyadic”, resp. “Triadic” system. Basically, each refinement eliminates redundancies in proof-search due to irrelevant sequentializations of inference figures in the sequent-based representation of proofs. The expressive power of Focusing is captured in a crisp way in a fully representative fragment of Linear Logic, called “LinLog”, introduced in [1] together with a normalization procedure from Linear Logic to LinLog. Usually the focusing theorem is proved in the linear sequent calculus and the proof is quite complex requiring an argument that makes use of a double induction. Andreoli and Maieli have shown in [2] that Focusing can also be interpreted in the proof net formalism, where it appears, at least in the multiplicative fragment of linear logic (MLL), to be a simple refinement of the “Splitting Lemma” for proof nets. The Splitting Lemma is at the core of the Sequentialization procedures for proof nets, and Focusing thus appears as a sequentialization strategy. This change of perspective allows the generalization of the Focusing result to (the multiplicative fragment of) any logic where the “Splitting Lemma” holds. Here we extend this idea of [2] to the case of MALL proof nets: we show how the focusing theorem for MALL can be interpreted as a refinement of a Focusing Lemma in which, in addition to the splitting case, it is also necessary to take into account clusters of tensor ($\otimes$) and plus ($\oplus$) links that are hereditarily ready to sequentialization (this is also known

Supported by Istituto Nazionale di Alta Matematica “Francesco Severi” (INdAM).

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

A. Ciabattini et al. (Eds.): WoLLIC 2022, LNCS 13468, pp. 1–17, 2022.

https://doi.org/10.1007/978-3-031-15298-6_1

as the “critical synchronous section”). In order to show this result we first need to fix a syntax for MALL proof nets. Unlike what happens for MLL proof nets, the syntax of MALL proof nets is not so peaceful and univocal [5,6,9]. There are essentially two syntaxes for MALL proof nets: that one by Girard [5] based on proof structures weighted with Boolean monomials, and a “more canonical” one by Hughes and Van Glabbeek [6]. Here, we choose the former syntax and we show how it is possible to transform monomial proof nets into *canonical forms*, in the same way as done by Hughes and Van Glabbeek. The canonical form of the (monomial) proof nets is now given by the adoption of an additive contraction link allowed only with atomic premises or with $\oplus$ premises coming from different instances of unary additive $\oplus$ links, i.e. $\oplus_1$ and $\oplus_2$. This new syntactic condition (on contraction links) allows to maximize the superposition of proof structures thus rendering them more canonical.

Paper Contributions. We characterize a (proper sub-)class of (monomial) proof nets, that is in correspondence with the class of focusing MALL sequent proofs: this is called the class of proof nets in canonical form (CPN, Definition 6). The correspondence between CPNs and focusing proofs is established via the sequentialization Theorem 4 which relies on the Focusing Theorem 3, a refinement of the Splitting and Ready Lemmas 4 and 5. The canonical form of a given proof net π ensures that the asynchronous conclusions (i.e., conclusions of type $\wp$, $\&$) of π are always ready to sequentialization, while the Focusing Theorem 3 allows to identify those synchronous conclusions links (of type $\oplus$, $\otimes$ links) of π that are hereditarily (i.e., recursively) ready to sequentialization.

2 The MALL Fragment of Linear Logic

In this paper, we consider only the pure (without units) multiplicative and additive fragment of Linear Logic (MALL). MALL formulas $A, B, \dots$ are built from literals (propositional variables $P, Q, \dots$ and their negations $P^\perp, Q^\perp, \dots$) and the binary connectives $\otimes$ (*tensor*), $\wp$ (*par*), $\&$ (*with*) and $\oplus$ (*plus*). Negation $(\cdot)^\perp$ extends to arbitrary formulas by the de Morgan laws: $(A \otimes B)^\perp = (A^\perp \wp B^\perp)$, $(A \wp B)^\perp = (A^\perp \otimes B^\perp)$, $(A \& B)^\perp = (A^\perp \oplus B^\perp)$, and $(A \oplus B)^\perp = (A^\perp \& B^\perp)$. A MALL sequent Γ is a multiset of formulas $A_1, \dots, A_n$. Sequents are one-sided, so we may omit turnstiles ($\vdash$). The rules of the proof system Σ_1 are depicted in the top part of Fig. 1. In the MALL fragment we consider, the refined focused system described in [1] can be reduced to Σ_2 of Fig. 1: it is called the “Dyadic System Σ_2 for MALL”. Connectives are split into two categories: *asynchronous* (or *negative*), $\wp$ and $\&$, corresponding to a kind of “don’t care non-determinism” and *synchronous* (or *positive*), $\otimes$ and $\oplus$, corresponding to a kind of “true non-determinism” w.r.t. proof-search. Furthermore, we assume that the class of atomic formulas is split into two dual, disjoint sub-classes: the *positive* atoms $X, Y, Z, \dots$ and their *negative* duals $X^\perp, Y^\perp, Z^\perp, \dots$ with $X^{\perp\perp} = X$ (but this distinction is only conventional). Focusing sequents are of two types: “ $\vdash \Gamma \uparrow L$ ” and “ $\vdash \Gamma \downarrow F$ ”, where Γ is a multiset of non-asynchronous formulas, L is a list of formulas and F is a single formula called the “focus” of the sequent. The Focusing system is justified by the following theorem (stated and proved in [1]):

Theorem 1 (Andreoli, 1992). *Let Γ be a multiset of non-asynchronous formulas and L an ordered list of formulas: $\vdash_{\Sigma_1} \Gamma, L$ if and only if $\vdash_{\Sigma_2} \Gamma \uparrow L$.*

The monadic sequent system Σ_1 for MALL:

$$\frac{}{A, A^\perp} \text{ax} \quad \frac{\Gamma, A \quad \Delta, A^\perp}{\Gamma, \Delta} \text{cut} \quad \frac{\Gamma, A \quad \Delta, B}{\Gamma, \Delta, A \otimes B} \otimes \quad \frac{\Gamma, A, B}{\Gamma, A \wp B} \wp \quad \frac{\Gamma, A \quad \Gamma, B}{\Gamma, A \& B} \& \quad \frac{\Gamma, A_i}{\Gamma, A_1 \oplus_i A_2} \oplus_{i=1,2}$$

The dyadic (focused) sequent system Σ_2 for MALL:

- Logical rules: $\wp, \&, \otimes, \oplus_{i=1,2}$

$$\frac{\Gamma \uparrow L, F, G}{\Gamma \uparrow L, F \wp G} \wp \quad \frac{\Gamma \uparrow L, F \quad \Gamma \uparrow L, G}{\Gamma \uparrow L, F \& G} \& \quad \frac{\Gamma \Downarrow F \quad \Delta \Downarrow G}{\Gamma, \Delta \Downarrow F \otimes G} \otimes \quad \frac{\Gamma \Downarrow F_i}{\Gamma \Downarrow F_1 \oplus F_2} \oplus_{i=1,2}$$

- Identity $[id]$: if F is a positive atom
- Reaction $[R \uparrow]$: if F is not asynchronous
- Reaction $[R \Downarrow]$: if F is neither synchronous nor a positive atom
- Decision $[D]$: if F is synchronous or a positive atom

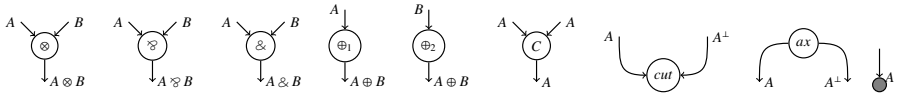
$$\frac{}{F^\perp \Downarrow F} id \quad \frac{\Gamma, F \uparrow L}{\Gamma \uparrow L, F} R \uparrow \quad \frac{\Gamma \uparrow F}{\Gamma \Downarrow F} R \Downarrow \quad \frac{\Gamma \Downarrow F}{\Gamma, F \uparrow} D$$

Fig. 1. the monadic Σ_1 (top) and dyadic Σ_2 (bottom) systems for MALL

The original proof consists in showing that any proof of Γ, L , in the standard sequent system Σ_1 , can be mapped, by permutation of inferences and deletion of dummy sub-proofs, into a proof of $\Gamma \uparrow L$ in the focusing system Σ_2 and vice-versa. In the following, we show a different proof of this result as a refinement of the sequentialization of MALL proof nets. Focusing basically appears as a strategy in the choice of the sequentializable formulas in the Sequentialization procedure. For doing that, we need first to choose a syntax for MALL proof structures which, unlike the MLL case, is neither standard nor univocal [5, 6, 9]. For several reasons¹ we prefer the syntax of [8] (refinement of [5]).

3 Proof Structures

Definition 1 (pre-proof structure). A MALL pre-proof structure (PPS) π is a directed graph such that each edge is labelled by a MALL formula, each node has a type in $\{ax, cut, \otimes, \wp, \&, \oplus_1, \oplus_2, C, \bullet\}$ and built according to the following typing constraints:



¹ Compared with the syntax of [6], monomial proof structures [8] are technically simpler; they allow us to easily extend to the MALL case arguments originally used for the MLL case such as Laurent's Splitting Lemma [7] and Andreoli-Maieli's Focusing Theorem [2]. Monomial proof structures have a natural presentation in terms of Coherent Spaces [4] and their correctness criterion can be also formulated in terms of "graph retraction steps" à la Danos [9].

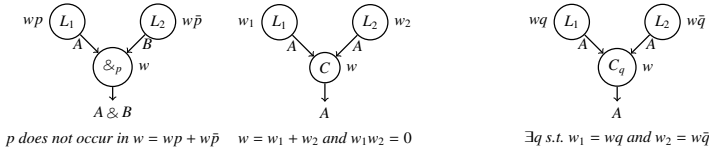
1. the entering (resp., outgoing or exiting) edges of a node L are the premises (resp., the conclusions) of L ;
2. each edge must be conclusion of exactly one node and premise of at most one node;
3. “•” denotes a dummy node whose unique premise is also called conclusion of π ;
4. a node whose conclusion (resp., conclusions) is conclusion (resp. are conclusions) of π are called terminal or conclusion nodes of π .

We call link the pseudo-graph made by a node together with its premise(s) and its conclusion(s) (if any); e.g. the previous figure displayed the so called MALL links.

Proof structures are PPSs equipped with boolean weights. Assume a set $\mathcal{B}$ of Boolean variables denoted by $p, q, r, \dots$, then a *monomial weight* (simply, a weight) $w, v, \dots$, over $\mathcal{B}$ is a product (conjunction) of variables or negation of variables of $\mathcal{B}$. We replace $p.p$ by p . Often, in a product of weights, v and w , we omit “.” and we write “ vw ” instead of “ $v.w$ ”. As usual in a Boolean algebra, we define the standard order relation “ $\leq$ ” between two weights v and w as follows: $v \leq w$ if there exists a weight v' s.t. $v = v'.w$. We also assume the following notation: 1 for the empty product, 0 for a product where both p and its negation $\bar{p}$ appear and ϵ_p for a variable p or $\bar{p}$. We say that a weight w *depends* on a variable p when ϵ_p appears in w ; two weights, v and w , are *disjoint* when $v.w = 0$.

Definition 2 (proof structure). A MALL proof structure (shortly, PS) is any PPS π whose nodes are equipped with monomial weights as follows:

1. we associate a Boolean variable ($p, q, \dots$), called *eigen weight*, to each $\&$ -node of π (eigen weights are supposed to be different);
2. we associate a weight w , i.e., a product (conjunction) of eigen weights or negations of eigen weights of π ($p, \bar{p}, q, \bar{q}, \dots$), to each node with the constraint that two nodes have the same weight if they have a common edge, except when the edge is the premise of a $\&$ or C-node: in these cases we proceed as follows:
 - (a) if w is the weight of a $\&$ -link and p is its eigen weight then w does not depend on p and its premise links, L_1 and L_2 , must have weights resp., $w.p$ and $w.\bar{p}$;
 - (b) if w is the weight of a C-link and w_1, w_2 are the weights of its premise links, L_1 and L_2 , then $w = w_1 + w_2$ and $w_1 w_2 = 0$ (see the two l.h.s. pictures below);



3. every node that is conclusion of π has weight 1 (dummy nodes • have weight 1);
4. (dependence condition²) if w is the weight of a $\&$ -link with eigen weight p and w' is a weight depending on p and appearing in the proof-structure then $w' \leq w$.

Fact 1. Since the weights associated to a PS are products (monomials) of the Boolean algebra generated by the eigen weights associated to a proof structure then, for each weight w associated to a contraction node, there exists a unique eigen weight q that splits w into $w_1 = wq$ and $w_2 = w\bar{q}$. We sometimes index a contraction node C with its splitting variable q , that is C_q as in the rightmost hand side picture above.

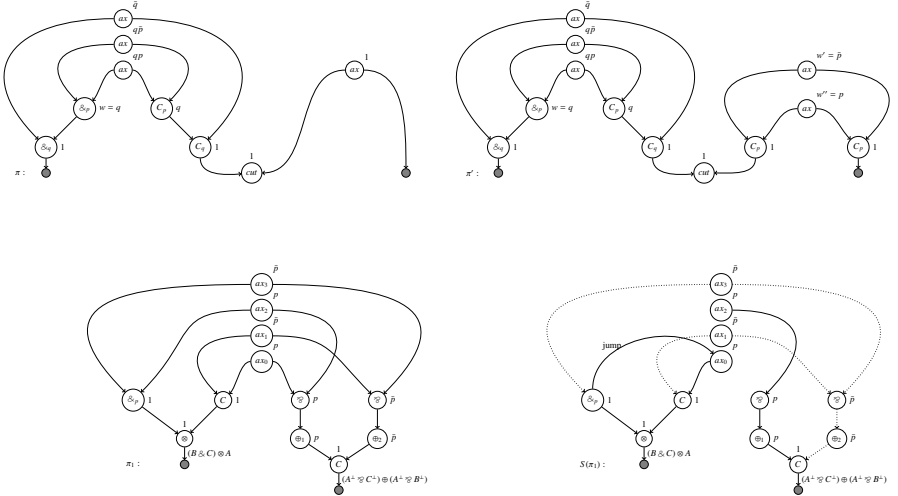
² The dependence condition corresponds to the *resolution condition* of [6].

Definition 3 (valuation, slice, switching). A valuation φ for a PS π is a function from the set of all weights of π into $\{0, 1\}$. Fixed a valuation φ for π then:

- a slice $\varphi(\pi)$ is the graph obtained from π by keeping only those nodes with weight 1 together with its outgoing edges (conclusion(s));
- a multiplicative switching (induced by φ), $S_m(\pi)$ of π , is the un-directed graph built on the nodes and edges of $\varphi(\pi)$ with the modification that for each $\wp$ -node we take only one premise and we cut the remaining one (it is called, left/right $\wp$ -switch);
- an additive switching (induced by φ) of π (or simply a switching), denoted $S_a(\pi)$ or simply $S(\pi)$, is a multiplicative switching where for each $\&$ -node L we cut the (unique) premise in $S_m(\pi)$ and we add an directed edge, called jump, from L to a node L' whose weight depends on the eigen weight of the $\&$ -node L .

Definition 4 (proof-net). A MALL PS π is correct, so it is a MALL proof net (PN), if every (additive) switching $S(\pi)$ is an acyclic and connected graph (ACC).

Example 1. The PPS π on the l.h.s. below is a PS while the PPS π' on the r.h.s. is not so since there exist a node whose weight $w' = \bar{p}$ (resp., $w'' = p$) depends on a $\&_p$ -node, whose weight is $w = q$, but $\bar{p} \not\leq q$ (resp., $p \not\leq q$), contradicting Definition 2(4). Observe that jumps are necessary for the correctness criterion (Definition 4), otherwise proof structures that are not image of any sequent proof of MALL would be correct. Consider e.g. the PS π_1 on the l.h.s. below (bottom). Actually π_1 is “correct” if we reason only by multiplicative slices although its conclusions $(B \& C) \otimes A, (A^\perp \wp C^\perp) \oplus (A^\perp \wp B^\perp)$ are not a provable sequent in MALL. Actually, fixed a valuation φ s.t. $\varphi(p) = 1$, there exists an additive switching $S(\pi_1)$ (induced by φ) that is not ACC as the one in the right hand side (note that $S(\pi)$ consists of the sub-graph with solid edges).

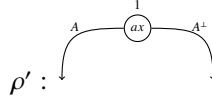
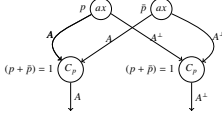


Definition 5 (substitution, restriction, empire, spreading). Let π be a PS, p and q eigen weights and w a weight in π , then:

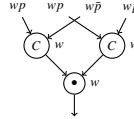
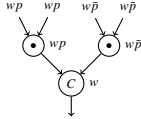
- the substitution of p by q in π , denoted with $\pi[q/p]$ is the graph π' obtained from π by replacing each occurrence of p (resp., $\bar{p}$) by q (resp., $\bar{q}$);
- the restriction of π w.r.t. p (resp., of π w.r.t. $\bar{p}$), denoted $\pi \downarrow^p$ (resp., $\pi \downarrow^{\bar{p}}$), is what remains of π when we replace p with 1 and $\bar{p}$ with 0 (resp., we replace p with 0 and $\bar{p}$ with 1) and keep only those vertexes and edges whose weight is still non null;
- the empire (or the dependency graph) of the eigen weight p w.r.t. π , denoted $\mathcal{E}_p$, is the (possibly disconnected) subgraph of π made by all links depending on p ;
- the spreading of w over π , denoted by $w[\pi]$, is the product of w for π , i.e., π in which we replace each weight v with the product of weights vw ;
- $\pi[\rho'/\rho]$ denotes the substitution³ in π of a sub-graph (or module) ρ with a graph ρ' .

Definition 6 (canonical form). Let π be a PS, ρ a sub-graph of π and π' and ρ' two graphs. We say that π commutes to π' , denoted $\pi \rightsquigarrow \pi'$, if π' is obtained from π by replacing ρ with ρ' , i.e. $\pi' = \pi[\rho'/\rho]$, by one of the following commutation rules:

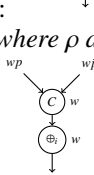
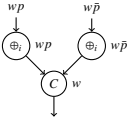
- R0: $\pi \rightsquigarrow \pi$, by R0, if $\pi' = \pi[\rho'/\rho]$ where ρ and ρ' are the two modules below:



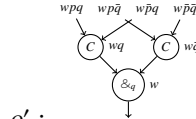
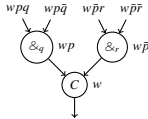
- R1: $\pi \rightsquigarrow \pi'$, by R1, if $\pi' = \pi[\rho'/\rho]$ where ρ and ρ' are as below and $\bullet \in \{\otimes, \wp\}$:



- R2: $\pi \rightsquigarrow \pi'$, by R2, if $\pi' = \pi[\rho'/\rho]$ where ρ and ρ' are as below, for $i = 1, 2$:



- R3: $\pi \rightsquigarrow \pi'$, by R3, if $\pi' = \pi[\rho'/\rho]$ where ρ and ρ' are as below with the condition that every weight w in π containing an occurrence of r (resp., of $\bar{r}$) has been replaced in π' by the weight $w' = w[q/r]$ (resp., by the weight $w' = w[\bar{q}/\bar{r}]$).



In every rule, ρ is called a redex of π (resp., a reductum of π') moreover, the (unique) contraction node that occurs in the redex is called the contraction (node) in commutation condition. We say that a PS π is in canonical form (CPS) iff it does not contain any redex ρ (or, equivalently, it has no contraction in commutation condition).

A proof net π is in canonical form (it is a **canonical proof net**, CPN) iff π is a CPS.

³ Observe that in general substitutions may not preserve the property of being a proof structure.

Proposition 1 (canonical form). *Let π be a PS, R_i ($0 \leq i \leq 3$) be one of the commutation rules of Definition 6 and π' be the graph s.t. $\pi \rightsquigarrow \pi'$ by R_i . If π is a PN then:*

- (1) π' is a PN too with the same conclusions of π ;
- (2) there exists a CPN π^c , with the same conclusions of π , s.t. it can be obtained from π by applying a finite number of instances of commutation rules, i.e. $\pi \rightsquigarrow^* \pi^c$.

Proof. (1) Commutation rules $R0, R1$ and $R2$, trivially preserve the property of being a PN. In case of an instance of $R3$ we need first to ensure that the dependence condition 4 of Definition 2 is preserved. We show that if L' is a node of π' whose weight w' depends on an eigen variable q then $w' \leq w$ where w is the weight of the $\&_q$ -link in π' .

[P1]: observe that since π is correct by hypothesis then, neither ϵ_r nor ϵ_q nor ϵ_r may occur in w and, for similar reasons, if v is the weight of the $\&_p$ then, neither ϵ_r nor ϵ_q nor ϵ_p may occur in the weight v of $\&_p$ neither in π nor in π' .

There are two possible cases for w' :

1. either w' is not effected by the substitution $[q/r]$, i.e. $w' = w'[q/r]$ and in this case since $w' \leq wp$ (by hypothesis is in π) then $w' \leq w$ also in π' (by [P1]);
2. or w' is effected by the substitution $[q/r]$ that is, $w' = w''[q/r]$ where w'' was the weight of L' in π before the substitution of the commutation rule $R3$. Since w'' is the weight of L' in π depending on the weight of $\&_r$, we know that $w'' \leq w\bar{p}$ so w'' has one of the following forms, $w'' = v'rw\bar{p}$ or $w'' = v'\bar{r}w\bar{p}$. Assume $w'' = v'rw\bar{p}$, then $w'' = v'rw\bar{p} \leq w\bar{p}$ and so, by transitivity, $w'' = v'rw\bar{p} \leq w$. Now, since ϵ_r does not occur in w (by [P1]), the substitution $v'rw\bar{p}[q/r] = w''[q/r] = w'$ thus $w' \leq w$.

It is not difficult to show that if L' is a node of π depending on an eigen weight $s \neq r$ then $w'' = w'[q/r] \leq v$ where w'' and v are, resp., the weights of L' and $\&_s$ in π' . We only show the case when $s = p$ and we omit the rest, that is, we show that if w' is the weight of a node L' depending on the eigen weight p of the node $\&_p$, then $w' \leq v$ where v is the weight of $\&_p$ in π' . Observe that since π is correct then v cannot depend neither on q nor on r in π and so v cannot depend on q (after eventually the substitution) in π' (otherwise we can easily find a switching in π containing a cycle). Now, if w' in π has the following form $w' = v'\epsilon_r\epsilon_p$, since by correctness of π neither ϵ_r nor ϵ_q nor ϵ_p may occur in v , from $w' = v'\epsilon_r\epsilon_p \leq v$ in π we conclude $w'[q/r] \leq v$.

Finally, in order to show that an instance of commutation rule $R3$ also preserves correctness we reason by contradiction. Assume by absurdum that π' is not correct, let us say that there exists a switching $S_\varphi(\pi')$ with a cycle or a disconnected component. Assume e.g. that $S_\varphi(\pi')$ (with e.g. $\varphi(\bar{q}) = 1$) contains a cycle, then this cycle must contain at least a node L' whose weight w' contains an occurrence of ϵ_q that replaced an occurrence of ϵ_r in w'' where w'' is the weight of L' in π' before the substitution (i.e., $w' = w''[q/r]$ or equivalently, $w'' = v.\epsilon_r$ and $w' = v.\epsilon_q$). Since w'' depends on $\&_r$ in π then w'' has the form $w'' = v.w\bar{p}\epsilon_r$ and since $w' = w''[\epsilon_q/\epsilon_r]$, then w' must have the form $w' = v.w\bar{p}\epsilon_r$ then it is easy to find a switching $S_\varphi(\pi)$ containing a cycle as in the two leftmost graphs of Fig. 2, contradicting the assumption that π is correct. We reason in a similar way in case $S_\varphi(\pi')$ contains a disconnected component.

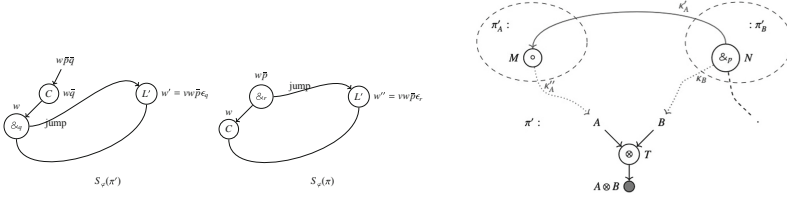
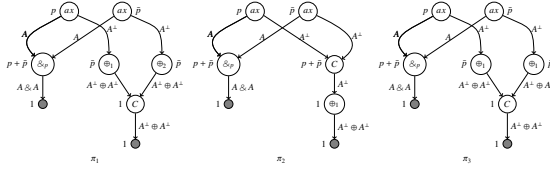


Fig. 2. cycles inside switchings (l.h.s.) and a blocking path for T (rightmost h.s.)

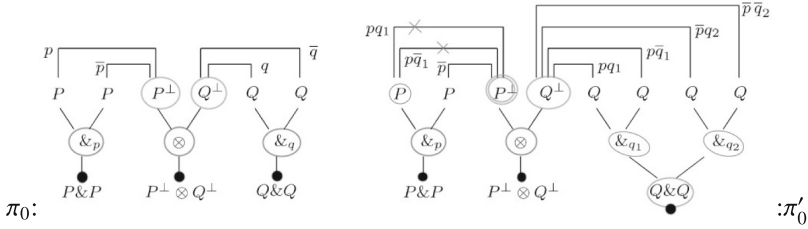
- (2) By induction on the sum of the logical degrees⁴ of the formulas that are conclusions of the uppermost contraction nodes that are in commutation condition in π .

□

Instances of CPNs are given below: indeed, only π_1 and π_2 are canonical (proof nets).



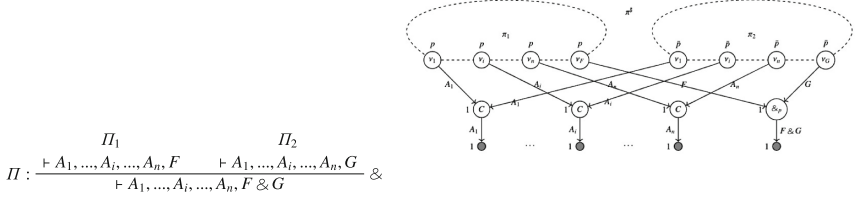
Actually, the notion of canonical form allows us to exclude redundant structures from the realm of MALL proof nets. Consider e.g. the two instances of monomial PS, π_0 and π'_0 below, with same conclusions, given in the Appendix 2 of [6]: only π_0 is a CPN; indeed, π_0 can be obtained from π'_0 by iterating the commutation rules of Definition 6: the only allowed (canonical) contractions are the blue ones while the two red ones, with conclusions resp., P and $P^\perp$ (in π'_0), are not allowed because they contract two identical axioms (thus, we can apply $R0$); moreover, the rightmost contraction with conclusion $Q \& Q$ is in commutation condition thus, by rule $R3$, it can be permuted with the two $\&$ -nodes above, $\&_{q_1}$ and $\&_{q_2}$. In the following we show that the notion of cut-free canonical proof net is sound and complete w.r.t. the notion of focusing sequent proof.



Theorem 2 (de-sequentialization). *A proof Π of a sequent Γ in Σ_1 can be mapped (i.e., de-sequentialized) into a CPN with same conclusions Γ .*

⁴ The *logical degree* of a formula F , denoted $\partial(F)$, is defined by induction on the height of F : if F is atomic then $\partial(F) = 0$, else F has the form $F_1 \circ F_2$, with $\circ \in \{\bar{\vee}, \otimes, \&, \oplus\}$, and $\partial(F) = \partial(F_1) + \partial(F_2) + 1$.

Proof. By induction of the height of Π via Proposition 1. All cases are easy except the case when last rule of Π is an instance of the $\&$ -rule, in this case we need to apply Proposition 1 in order to get a canonical proof net. Assume last rule of Π is a $\&$ -rule. By hypothesis of induction, Π_1 (resp., Π_2) de-sequentializes into the canonical proof net π_1 (resp., π_2) with same conclusions. We may then link together π_1 and π_2 by a $\&$ -link between conclusions F and G and by adding n a contraction C -links, one contraction for each pair of identical conclusions A_i, A_i coming from resp., π_1 and π_2 . Thus we build the proof net $\pi^\sharp$, as in the picture below, on which we may finally apply Proposition 1(3) in order to get the canonical proof net π which Π de-sequentializes to.



□

Definition 7 (ready and splitting links). Let π be a CPN. A link L of π is ready (to sequentialization) whenever deleting everything of L except its premise(s) produces one or more sub-proof nets having among their conclusions the premise(s) of L . A conclusion of π is ready if it is the conclusion of a ready link.

If L is a terminal $\otimes$ -link of π of type $\frac{A \ B}{A \otimes B}$, we say that L is splitting for π when removing L from π (we erase everything of L except its premises) splits π in two sub-proof nets: π_A , having A among its conclusions and π_B , having B among its conclusions. $\text{Split}(\pi)$ denotes the set of terminal tensor links that are splitting for π .

We say that π with at least a terminal tensor link is in splitting condition iff it does not contain neither an asynchronous conclusion nor a ready conclusion of type $\oplus$.

Fact 2 (terminal links of type $\oplus_i$ or $\wp$ are ready). Assume π is a CPN with conclusions Γ, F . If L is a terminal link of type $\frac{A_i}{A_1 \oplus A_2} \oplus_i$ with $F = A_1 \oplus A_2$ (resp., of type $\frac{A \ B}{A \wp B}$ with $F = A \wp B$) then, L is a ready link and removing L as in Definition 7 produces a sub-proof net π_{A_i} (resp., $\pi_{A,B}$) with conclusions Γ, A_i (resp., Γ, A, B).

Note that a terminal tensor link of a proof net π may be not ready to sequentialization (it may be “non splitting” for π). A contraction link is never ready “alone”: its “readiness” is subordinate to that one of the $\&$ -link which this contraction depends on. In the following, we adopt some notions of [7] adapted to the case of MALL cut-free CPNs.

Definition 8 (switching, descending and blocking paths).

- Given a CPS π , a **jump graph** for π (or a jumped PS π), denoted $J(\pi)$, is the graph obtained by adding to π some (possibly none) jumps; we allow in $J(\pi)$ jumps from an $\&_p$ -node to a C_p node depending on p ⁵.
- Fixed a $J(\pi)$ for π , a **switching path** γ in $J(\pi)$ is a path that exists in some switching $S(\pi)$ of π . We say that n switching paths $\gamma_1, \dots, \gamma_n$ of $J(\pi)$ are compatible iff there exists a switching $S(\pi)$ s.t. $\gamma_1, \dots, \gamma_n$ are paths of $S(\pi)$.

⁵ Note that a $J(\pi)$ differs from a switching $S(\pi)$ for the following facts: (i) we do not consider slices, (ii) we do not mutilate premises and (iii) there can be multiple (possibly, none) jumps exiting from a $\&_p$ -node and going to different nodes depending on p or C_p nodes.

- If e is an edge of π , its **descending path** $\delta(e)$ is the unique directed path starting from e and ending with the premise of a terminal node. If N is a node other than the axiom then, $\delta(N)$ denotes the descending path of the unique conclusion of N . $\delta(N)$ is empty if and only if N is terminal.
- Let T be a $\otimes$ node of a proof net π and $J(\pi)$ one of its possible jump graphs:
 - a **correctness/blocking node** for T is a node N of type $\nabla \in \{\wp, \&\}$ with two disjoint switching paths of $J(\pi)$, κ_0 and κ_1 , going from T to N and s.t. both paths start with a premise of T and end with a premise of N or a jump of N in case N is a $\&$ -node; κ_0 and κ_1 are called **correctness paths** for T ;
 - a **blocking path** for T is a path γ in $J(\pi)$ that goes from one premise to the other of T (without passing through the conclusion of T) and bouncing on both the premises (resp., on one premise and one jump) of a blocking node N of type $\wp$ (resp. of type $\&$); in other words, γ starts from one premise of T , it ends with the second premise of T and it also enters one premise of N and immediately exits the other premise of N (or it enter one premise of N and exits with a jump, or the other way round, in case $N = \&$); thus $\gamma = \kappa_A \cdot N \cdot \kappa_B$ appears as in the graph (A) of Fig. 3 where “ $\cdot$ ” denotes the concatenation of switching paths.

E.g. graph (F) of Fig. 3 is an instance of jumped CPN in which the (unique) $\wp$ -node is a blocking node for $\otimes_3$ while the (unique) $\&_p$ -node is a blocking node for $\otimes_2$.

Next two Facts and Lemmas 1, 2 and 3 are used to prove Lemma 4 which is necessary for the Ready Lemma 5, the “pivot” lemma of the Focusing Theorem 3.

Fact 3. *The two correctness paths, κ_A and κ_B for $T = \frac{A \cdot B}{A \otimes B}$, are compatible switching paths and $\kappa \cdot T \cdot \kappa_B$, i.e., the path going from κ_A to κ_B (or the other way round) and bouncing on the two premises of T (without going through the conclusion of T), is a compatible switching path too.*

Fact 4. *If N is a node of a CPN other than the ax-node, then $\delta(N)$ is a switching path.*

Lemma 1 (blocking contraction). *Let π be a CPN with conclusions $\Gamma = A_1, \dots, A_n$ s.t. none of them is conclusion of a terminal $\&$ -link.*

- (i) *If some A_i is conclusion of a terminal contraction link L_i and N is the $\&$ -node which L_i depends on, then neither A_i (i.e., L_i) nor N is ready (N is called the blocking contraction node for L) see picture (B) of Fig. 3.*
- (ii) *Moreover, there does not exist any switching path exiting the conclusion of N and stopping (downwards) at L_i as e.g. γ' and γ'' in pictures (C) and (D) of Fig. 3.*

Proof. (i) - By definition of PS, every contraction link depends on the eigen variable of some $\&$ link of π , thus in particular any terminal contraction link L_i depends on the eigen weight p of some $\&_p$ link of π and since by assumption none terminal link of π is of type $\&$, the $\&_p$ -node must be above some conclusion A_j of π (by correctness of π , $i \neq j$); thus the $\&_p$ -link is not ready to sequentialization yet as in Fig. 3(B) of where the terminal link $\circ$ below the $\&_p$ -link is such that $\circ \in \{\oplus, \otimes, \wp\}$.

- (ii) - it follows by correctness of π (see graphs (C) and (D) of Fig. 3). □

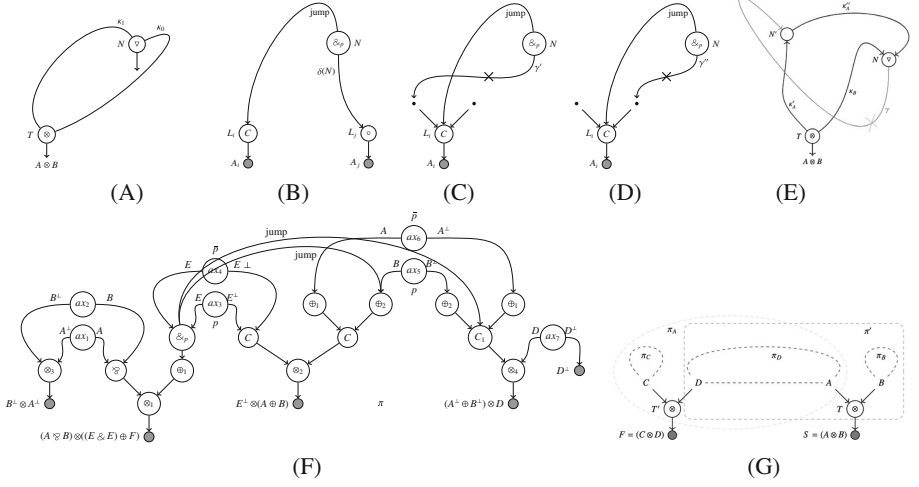


Fig. 3. switching, blocking, descendent and correctness paths, and splitting conclusions

E.g., the $\mathcal{S}_p$ -node of picture (F) of Fig. 3 is blocking for contraction node C_1 .

Lemma 2. *If π is a PN with conclusions $A_1, \dots, A_n$ then there exists at least a conclusion A_i that is not conclusion of a contraction link.*

Proof. Assume π is a PN having only terminal contraction links, $L_1, \dots, L_n$. Then, by Lemma 1 if N is the node $\mathcal{S}_p$ which L_i depends on then, N is not ready so it must be above a terminal contraction link L_j with $i \neq j$, by correctness of π . Since the weight of L_i is 1 (by definition of PS), by the dependency condition, also the weight of N must be 1 but then the $\mathcal{S}_p$ -link cannot be above any contraction link L_j , a contradiction. $\square$

Lemma 3 (blocking splitting). *Let π be a PN with a terminal tensor link T that is not splitting for π then: (i) there exists a blocking node N of type $\nabla \in \{\wp, \&\}$ for T moreover; (ii) every switching path exiting the conclusion of N and compatible with κ_0 and κ_1 cannot contain any node of the correctness paths, κ_0 and κ_1 , for T .*

Proof. By induction on the number n of $\mathcal{S}$ -nodes of π . If $n = 0$ (we are in the MLL case, [7]) then let T be a terminal $\otimes$ node and $S(\pi)$ be a (multiplicative) switching: the removal of T splits $S(\pi)$ into two connected components (by the ACC-correctness). If all $\wp$ nodes are such that both their premises belong to the same connected component, then T is a splitting node for π since the removal of T in π has two connected components as well (which are ACC-correct). Otherwise there exists a $\wp$ node N with a premise in each connected component of the removal of T in $S(\pi)$. Each of these components contains a premise of T and a premise of N and (by connectivity) a path from the first to the second. The two obtained paths are switching and disjoint. Finally, in the component containing N , the obtained path cannot contain the conclusion of N otherwise, one could connect the two paths and obtain a cycle in the correctness graph $S'(\pi)$ obtained from $S(\pi)$ by changing the choice of the premise of N ; for similar reasons (since by Fact 4, $\delta(N)$ is a switching path), $\delta(N)$ cannot meet neither κ_A nor κ_B (see graph (E) of Fig. 3).

Otherwise, $n > 0$ then, we assume by absurdum that π is the smallest (w.r.t. the graphical size) proof net containing a terminal $\otimes$ node T that is not splitting for π and such that there exists no blocking node N for T . Let us choose a $\&$ node N in π in such a way that it is as low as possible (i.e., the weight of N is 1). It is always possible to find such a node N in a correct proof net, otherwise we can find a switching for π containing a cycle, contradicting the correctness assumption. Let p be the eigen weight of N and let $\pi \downarrow^p$ the restriction of π w.r.t. p . Clearly $\pi \downarrow^p$, after having properly removed the residual unary $\&$ or C nodes left, is a correct proof net, let us say π' , of smaller size than π but with still the same terminal tensor links of π (although the premises of these tensors may have been changed). Thus, in π' every terminal tensor link is either splitting or there exists a blocking node for such a tensor.

1. if T is not splitting for π' then there exists a blocking node N' of type ∇ for T which is also blocking for T in π : we can consider a switching $S'_\varphi(\pi)$ that is the switching $S'(\pi)$ induced by a valuation φ s.t. $\varphi(p) = 1$ (resp., $\varphi(\bar{p}) = 0$);
 2. otherwise, the removal of T splits π' into two sub-proof nets, π'_A and π'_B with A and B the two premises of T . Now, if we restore the restriction $\pi \downarrow^p$ then, the removal of T from $\pi \downarrow^p$ induces two graphs, $(\pi \downarrow^p)_A$ $(\pi \downarrow^p)_B$, corresponding resp., to π'_A and π'_B after removing the residual “unary” node $\&_p$ and all residual “unary” contraction links of type C_p . Note that the residual unary node $\&_p$ must occur either in $(\pi \downarrow^p)_A$ or in $(\pi \downarrow^p)_B$; let us say $\&_p$ stays in $(\pi \downarrow^p)_B$. Then, by correctness we know that: [P1]: for every switching $S_\varphi(\pi \downarrow^p)_B$ there is a path from B to the unary node $\&_p$. There are two cases for $(\pi \downarrow^p)_A$:
 - (a) either no residual unary C_p node depending on p occurs in $(\pi \downarrow^p)_A$; this means that π'_A (the proof net obtained from the restriction $(\pi \downarrow^p)_A$, after the removal of residual unary node, contains none node depending on p therefore it π'_A is a sub-proof net of π therefore T is splitting for π ; a contradiction;
 - (b) or at least a residual unary C node M depending on p occurs in $(\pi \downarrow^p)_A$. Then we can easily build a jump graph for π' , let us say $J(\pi')$ containing a blocking path for T as in the rightmost h.s. picture of Fig. 2, a contradiction:
 - i. consider the switching path κ_A that starts with the jump from $\&_p$ to M (i.e., κ'_A) and continues up to the left premise A of T (i.e., κ'_A), and then consider
 - ii. the second switching path κ_B starting from the unique premise of $\&_p$ and continuing up to the right premise B of T ; this path exists by [P1].
- See e.g., also Fig. 3(F) with $T = \otimes_2$ and $N = \&_p$.

In order to show (ii), observe that $\delta(N)$ cannot meet κ_A (resp., κ_B) in a node, let us say M' (resp., M'') otherwise we could find a switching for π' containing a cycle starting from the conclusion of $N = \&_p$ and following the red path in the graph (E) of Fig. 3. $\square$

Lemma 4 (splitting). *If π is a CPN in splitting condition then $\mathbf{Split}(\pi) \neq \emptyset$ (see Definition 7).*

Proof. Assume π is a CPN in splitting condition, then by Definition 7 and by Contraction Lemma 1 none conclusion of π that is conclusion of a contraction link is ready (that holds in particular for all synchronous conclusions of type $F \oplus G$). Assume by absurdum that none synchronous conclusion is ready thus none tensor conclusion is splitting for

π . We show that there exists a jump graph $J(\pi)$ for π containing a switching path with a cycle, contradicting the assumption π is correct. Let $F_i = A_i \otimes B_i$ be the conclusion of a terminal tensor link T_i of π .

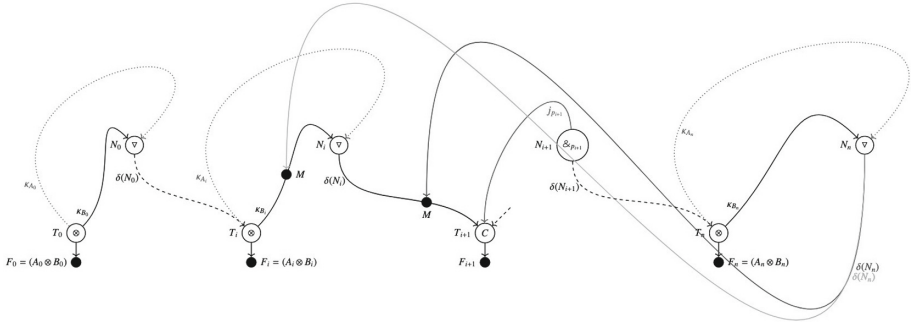
[P.1] Since by assumption T_i is not splitting then, by Lemma 3, there exists a blocking node N_i (of type ∇) for T_i with two correction paths, κ_{A_i} and κ_{B_i} . Starting from N_i , we follow $\delta(N_i)$ until we reach a terminal node T_j which, by Lemma 3(ii), must be different from T_i . By correctness of π , κ_{A_i} , κ_{B_i} and $\delta(N_0)$ are compatible switching paths. There are now two cases for T_j :

1. either T_j is a terminal tensor link then, we continue as before in [P.1] (with T_j at the place of T_i) until we reach a new conclusion T_k which must be different from all already visited conclusions, by Lemma 3(ii);
2. or T_j is a terminal contraction link then, we then continue following the jump j_p (taken in the opposite direction) to the $\otimes_p$ -node N_j on which the contraction T_j depends on and then we continue with the descendent path $\delta(N_j)$ until we reach a new conclusion T_k which, by Lemma 1, must be different from all terminal nodes previously visited. Observe that, by correctness of π , the composition of switching paths, $T_j \cdot j_p \cdot (\otimes_p = N_j) \cdot \delta(N_j)$, entails a compatible switching path.

Iterating steps 1 and 2 above, we build an infinite sequence

$$\nu = \kappa_{A_i} \cdot N_i \cdot \delta(N_i) \cdots T_j \cdot j_p \cdot N_j \cdot \delta(N_j) \cdots T_k \cdot \kappa_{B_k} \cdot N_k \cdot \delta(N_k) \cdots$$

where A_i, B_i are the premises of a generic terminal tensor link $T_i : \frac{A_i \ B_i}{A_i \otimes B_i}$ and j_{p_i} denotes a jump (taken in the opposite direction) going from a $\otimes_p$ node to a terminal contraction node T_j depending on p . Since π is finite, ν must visit twice a same node M . Observe that ν exists in a $J(\pi)$ and it is not difficult to show that there exists a switching $S(\pi)$ containing such a ν (all components of ν are compatible since steps 1 and 2 preserve compatibility), contradicting the correctness of π . Next figure illustrates two possibilities for node M depending on whether the descendent path $\delta(N_n)$ meets a descendent path (black option) or a correction path (grey option) already visited by ν ; indeed, it does not matter the type of node M we can always find a cycle in a switching path.



□

E.g., $\text{Split}(\pi) = \{\otimes_1, \otimes_4\}$ for the CPN π of Fig. 3(F) (neither $\otimes_2$ nor $\otimes_3$ is splitting for π).

Lemma 5 (ready). *If π is a CPN s.t. it is not an ax-link and it has no asynchronous conclusions, then there exists a terminal link that is a ready $\oplus_i$ -link or a splitting $\otimes$ -link.*

Proof. Let π be a CPN s.t. it is not an *ax*-link and it has no asynchronous conclusion. If π has no terminal contraction links and, since π is not reduced to an axiom link, it must contain at least a terminal synchronous links L ($\oplus_i$ or $\otimes$); if L is an $\oplus_i$ -link then it is trivially ready, otherwise π has only terminal $\otimes$ -links then it is in splitting conditions and so, by the Splitting Lemma 4, there exists a splitting link.

Otherwise, if π contains terminal contractions then, by Lemma 2, it cannot contain only terminal contraction links and, since it is not reduced to an axiom link and it has no asynchronous conclusions, it must contain at least a synchronous terminal link L ; if L is an $\oplus_i$ -link then it is trivially ready, otherwise π has only terminal links that are contractions (of non asynchronous formulas) or terminal $\otimes$ -links then it is in splitting condition and so, by the Splitting Lemma 4, there exists a splitting link. $\square$

Definition 9 (focusing conclusions). Let π be a CPN and F be one of its conclusions. F is focusing for π (we write, $F \in \mathbf{Foc}(\pi)$) iff one of the following conditions holds:

1. F is a positive atom and π is reduced to an axiom link.
2. F is the conclusion of a terminal $\oplus_i$ -link L of type $\frac{A_i}{A_1 \oplus A_2} \oplus_i$ and A_i is asynchronous or a negative atom or $A_i \in \mathbf{Foc}(\pi_{A_i})$, for $1 \leq i \leq 2$.
3. $F = (A \otimes B) \in \mathbf{Split}(\pi)$ and π is split at F into two sub-PNs, π_A and π_B , and
 - (a) A is asynchronous or a negative atom or $A \in \mathbf{Foc}(\pi_A)$ and
 - (b) B is asynchronous or a negative atom or $B \in \mathbf{Foc}(\pi_B)$;

where π_{A_i} (resp., π_A, π_B) is (resp. are) the sub-proof net(s) obtained by removing from π the vertex $\oplus_i$ (resp., $\otimes$) of L together with its outgoing edge $A_1 \oplus A_2$ (resp., $A \otimes B$).

Proposition 2. Let π be a CPN with no asynchronous conclusion.

1. If L is a terminal $\otimes$ -link, $\frac{A \otimes B}{A \otimes B} \otimes$, that is splitting for π and π_A and π_B are the two CPNs obtained by splitting π at L and A is not a negative atom then $\mathbf{Foc}(\pi_A) \setminus \{A\} \subseteq \mathbf{Foc}(\pi)$ (and similarly for the B side).
2. If L is a terminal $\oplus_i$ link, $\frac{A_i}{A_1 \oplus A_2} \oplus_i$, for $1 \leq i \leq 2$, and π_{A_i} is the sub-CPN obtained by removing L and A_i is a non negative atom then $\mathbf{Foc}(\pi_{A_i}) \setminus \{A_i\} \subseteq \mathbf{Foc}(\pi)$.

Proof. We only discuss case 1 (case 2 is simpler so we omit it). Assume $S = A \otimes B$ is splitting for π with no asynchronous conclusion. We reason by induction on the size of π . We show that if $F \in \mathbf{Foc}(\pi_A) \setminus \{A\}$ then $F \in \mathbf{Foc}(\pi)$. Since F is focusing in π_A , there are three cases to consider according to Definition 9:

1. F is a positive atom and π_A is reduced to an axiom link, with conclusions F and $F^\perp$, one of which being A . But, by hypothesis, A is not a negative atom, hence $A \neq F^\perp$; moreover, by hypothesis, $F \in \mathbf{Foc}(\pi_A) \setminus \{A\}$, hence $A \neq F$. Contradiction.
2. $F = (C \otimes D) \in \mathbf{Foc}(\pi_A)$ and π_A is split at F into two sub-CPNs, π_C and π_D s.t.:
 - [★1] C is asynchronous or a negative atom or $C \in \mathbf{Foc}(\pi_C)$;
 - [★2] D is asynchronous or a negative atom or $D \in \mathbf{Foc}(\pi_D)$.

Since A is a conclusion of π_A different from F (by hypothesis, $F \in \mathbf{Foc}(\pi_A) \setminus \{A\}$) and π_A is split at F into π_C and π_D then A must be in the conclusions of π_C or π_D . We assume, without loss of generality, that A is a conclusion of π_D (other than D , obviously). Let π' be the PS consisting of π_D and π_B and the splitting link of π at S , as in the picture (G) of Fig. 3. It is not difficult to see that:

[★3] π' is a CPN split at S into π_D and π_B and

[★4] π is split at F into π_C and π' .

In case $D \in \text{Foc}(\pi_D)$, then D must be synchronous otherwise π_D must be an axiom and, since $A \neq D$, we would get $A = D^\perp$, contradicting the assumption; now, since π' is smaller (in size) than π , by the induction hypothesis applied to [★3], we infer:

[★5] $\text{Foc}(\pi_D) \setminus \{A\} \subseteq \text{Foc}(\pi')$ so $D \in \text{Foc}(\pi')$ since $D \in \text{Foc}(\pi_D) \setminus \{A\}$ and $A \neq D$.

From [★5], [★2] and $D \neq A$, we get:

[★6] D is asynchronous or a negative atom or $D \in \text{Foc}(\pi')$.

From [★1], [★6] and [★4], by Definition 9, we conclude that $F \in \text{Foc}(\pi)$.

3. $F = C \oplus D$ is a synchronous formula of π_A that is conclusion of a ready terminal link $\oplus_1$ (resp., $\oplus_2$) and π_C (resp., π_B) is the sub-CPN s.t. C (resp., D) is asynchronous or a negative atom or $C \in \text{Foc}(\pi_C)$ (resp., $D \in \text{Foc}(\pi_D)$) then, we proceed as in case 2. $\square$

Theorem 3 (focusing). *If π is a PN with no asynchronous conclusion then, $\text{Foc}(\pi) \neq \emptyset$.*

Proof. We proceed by contradiction. Assume there exists a CPN π with no asynchronous conclusion and s.t. $\text{Foc}(\pi) = \emptyset$. We choose π of minimal size. There are two cases:

1. Either π has no synchronous conclusion then, since it contains neither asynchronous conclusion (by assumption) and since by Lemmas 1 and 2 it cannot contain any contraction link, π must be an axiom link. But then, one of the two conclusions must be a positive atom which, by Definition 9, is focusing for π . Contradiction.
2. Or π does contain at least one synchronous conclusion, and since it contains no asynchronous conclusion, by application of the Ready Lemma 5, we know that there exists a synchronous conclusion F of π that is either a ready conclusion of type $A \oplus B$ or a splitting conclusion of type $A \otimes B$. We only discuss the latter case (the former case is similar, so omitted). Assume there exists a synchronous conclusion $F = A \otimes B$ of π which splits π into two sub-proof-nets, π_A and π_B . Suppose that:

[★1] A is neither asynchronous nor a negative atom.

By construction, the conclusions of π_A other than A are conclusions of π hence not asynchronous. Since A itself is not asynchronous by [★1], then none of the conclusions of π_A are asynchronous. Since π_A is strictly smaller than π , which is a PN of minimal size without asynchronous nor focusing conclusions, we infer that:

[★2] $\text{Foc}(\pi_A) \neq \emptyset$.

Now, A is not a negative atom by [★1], hence by Proposition 2, we have that:

[★3] $\text{Foc}(\pi_A) \setminus \{A\} \subseteq \text{Foc}(\pi)$.

Since $\text{Foc}(\pi) = \emptyset$, by [★3], we conclude that $\text{Foc}(\pi_A) \subseteq \{A\}$ and thus $\text{Foc}(\pi_A) = \{A\}$, by [★2]. Hence $A \in \text{Foc}(\pi_A)$. Thus, by discharging hypothesis [★1], we conclude:

[★4] A is asynchronous or a negative atom or $A \in \text{Foc}(\pi_A)$.

Symmetrically, we can equally prove that:

[★5] B is asynchronous or a negative atom or $B \in \text{Foc}(\pi_B)$.

From [★4] and [★5], by Definition 9, we conclude that $F \in \text{Foc}(\pi)$. Contradiction. $\square$

Consider e.g. the CPN π of Fig. 3(F) then, $\text{Foc}(\pi) = \{(A \wp B) \otimes ((E \& E) \oplus F)\}$.

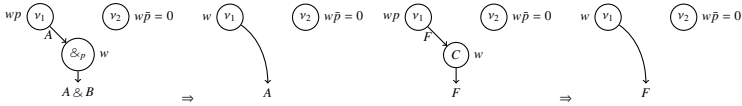
The canonical form of PN (Proposition 1) together with the Focusing Theorem 3 assure a sequentialization strategy (Theorem 4) mapping CPNs in to focusing proofs.

Theorem 4 (focusing sequentialization). *A cut-free CPN π with conclusions Γ sequentializes into a focusing MALL sequent proof $\pi^\sharp$ with conclusion:*

- ★ $\vdash \Gamma' \Downarrow F$, if $\Gamma = \Gamma'$, F does not contain asynchronous conclusion and $F \in \text{Foc}(\pi)$;
- ★ $\vdash \Gamma' \Uparrow L$ otherwise, where $\Gamma = \Gamma', L$ and Γ' is a multiset of non-asynchronous formulas and L is a list of formulas.

Proof. By induction on the size of π . If π is an (atomic) axiom link with conclusions $A, A^\perp$ then, $\pi^\sharp : \frac{\vdash A^\perp \Downarrow A}{\vdash A^\perp, A \Uparrow}$. Otherwise, if π contains an asynchronous conclusion F then since π is in canonical form, F is conclusion of a terminal asynchronous link $L, \wp$ or $\&$.

1. If $L : \frac{A \ B}{A \wp B}$ and π has conclusions $\Gamma = \Gamma', A \wp B$ then we can remove L (the vertex $\wp$ together with its conclusion edge labelled by $A \wp B$) and get a CPS $\pi_{A,B}$ with conclusions Γ', A, B which is trivially correct. By hypothesis of induction $\pi_{A,B}$ sequentializes in to $\pi''^\sharp : \vdash \Gamma'' \Uparrow L, A, B$ from which we conclude by an instance of $\wp$ -rule $\pi^\sharp : \frac{\vdash \Gamma'' \Uparrow L, A, B}{\vdash \Gamma'' \Uparrow L, A \wp B}$. Note that some instances of $R \Uparrow$ can be applied in case that A or B were no longer asynchronous formulas.
2. If L is a link of type $\&$, i.e. $L : \frac{A \ B}{A \& B}$, with eigen weight p and $\Gamma = \Gamma', A \& B$, then:
 - (a) take the restriction of π w.r.t. p , $\pi \downarrow^p$ (resp., the restriction of π w.r.t. $\bar{p}$, $\pi \downarrow^{\bar{p}}$);
 - (b) in $\pi \downarrow^p$ (resp., $\pi \downarrow^{\bar{p}}$) erase the (unique) vertex labeled by $\&_p$ and merge its emergent edge (its conclusion) together with its unique incident edge (its unique non-null premise) labelled by A (resp., by B), as in the figure below;
 - (c) in $\pi \downarrow^p$ (resp., $\pi \downarrow^{\bar{p}}$) erase every residual (unary) vertex of type C_p and merge its outgoing edge (the conclusion) together with its unique incident edge (the unique non-null premise) labelled by the contracted formula F (as below).



The resulting graph is a proof net π_A (resp., π_B) with conclusions Γ', A (resp., Γ', B). By hypothesis of induction, π_A and π_B sequentialize in to $\pi_1^\sharp : \vdash \Gamma'' \Uparrow L, A$ and $\pi_2^\sharp : \vdash \Gamma'' \Uparrow L, B$ thus by an instance of $\&$ -rule we conclude $\pi^\sharp : \frac{\vdash \Gamma'' \Uparrow L, A \quad \vdash \Gamma'' \Uparrow L, B}{\vdash \Gamma'' \Uparrow L, A \& B}$ where $\Gamma' = \Gamma'', L$. Note that some instances of $R \Uparrow$ could be applied upwards in case that A or B were no longer asynchronous formulas.

In case π has no asynchronous conclusions, since by hypothesis is not an axiom link, at least one of its conclusions is conclusion of a synchronous link (by the Ready Lemma 5) then, by Focusing Theorem 3, there exists $F \in \text{Foc}(\pi)$. Assume $F = A \otimes B$.

1. If $A \in \text{Foc}(\pi_A)$ and $B \in \text{Foc}(\pi_B)$, we apply the hypothesis of induction on π_A and π_B and we get two focusing proofs, $\pi_1^\sharp : \vdash \Gamma'_1 \Downarrow A$ and $\pi_2^\sharp : \vdash \Gamma'_2 \Downarrow B$, that we can assemble together in to the proof $\pi^\sharp$ as in the l.h.s. below.

2. Otherwise, in the case A (resp., B) is a negative atom or an asynchronous formula, we apply the hypothesis of induction on π_A (resp., π_B) and we get a focusing proof $\pi_1^\# \vdash \Gamma'_1, A \uparrow$ (resp., $\pi_2^\# \vdash \Gamma'_2 \uparrow B$) from which we conclude with a proof $\pi^\#$ as in the r.h.s. below (where e.g., A is a negative atom and B is an asynchronous formula).

$$\pi^\# : \frac{\frac{\Gamma'_1 \Downarrow A \quad \Gamma'_2 \Downarrow B}{\Gamma' \Downarrow A \otimes B} \otimes \quad \frac{\frac{\frac{\Gamma'_1, A \uparrow}{\Gamma'_1 \uparrow A} R \uparrow \quad \frac{\Gamma'_1 \uparrow B}{\Gamma'_2 \Downarrow B} R \Downarrow}{\Gamma' \Downarrow A \otimes B} \otimes}{\pi^\#}$$

□

4 Conclusions

We are finally ready to give a proof of **Andreoli's Theorem 1**:

Proof. Let Π be a proof in Σ_1 of the sequent $\vdash \Gamma, L$ which, by Theorem 2 and Proposition 1, de-sequentializes in to the canonical proof net π of Γ, L which finally sequentializes, by Theorem 4, in a proof Π' of the sequent $\vdash_{\Sigma_2} \Gamma \uparrow L$ in Σ_2 (see the diagram).

$$\begin{array}{ccc} \text{sequent proof : } \Pi & \xrightarrow{\text{de-sequentialization}} & \pi : \text{proof net} \\ \downarrow & & \downarrow \\ \text{focused proof : } \Pi' & \xleftarrow{\text{foc-sequentialization}} & \pi' : \text{canonical proof net} \end{array}$$

□

References

1. Andreoli, J.-M.: Logic programming with focusing proofs in linear logic. JLC 2(3), 297–347 (1992). <https://doi.org/10.1093/logcom/2.3.297>
2. Andreoli, J.-M., Maieli, R.: Focusing and proof-nets in linear and non-commutative logic. In: Ganzinger, H., McAllester, D., Voronkov, A. (eds.) LPAR 1999. LNCS (LNAI), vol. 1705, pp. 320–336. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48242-3_20
3. Danos, V., Regnier, L.: The structure of multiplicatives. AML 28, 181–203 (1989). <https://doi.org/10.1007/BF01622878>
4. Girard, J.-Y.: Linear logic. TCS 50, 1–102 (1987). [https://doi.org/10.1016/0304-3975\(87\)90045-4](https://doi.org/10.1016/0304-3975(87)90045-4)
5. Girard, J.-Y.: Proof-nets: the parallel syntax for proof theory. In: Logic and Algebra (1996). <https://doi.org/10.1201/9780203748671>
6. Hughes, D., Van Glabbeek, R.: Proof nets for unit-free multiplicative-additive linear logic. In: Proceedings of IEEE Logic in Computer Science (2003). <https://doi.org/10.1109/LICS.2003.1210039>
7. Laurent, O.: Sequentialization of multiplicative proof nets. Manuscript, April 2013. <https://perso.ens-lyon.fr/olivier.laurent/seqmll.pdf>
8. Laurent, O., Maieli, R.: Cut elimination for monomial MALL proof nets. In: LICS 2008 (2008). <https://doi.org/10.1109/LICS.2008.31>
9. Maieli, R.: Retractable proof nets of the purely multiplicative and additive fragment of linear logic. In: Dershowitz, N., Voronkov, A. (eds.) LPAR 2007. LNCS (LNAI), vol. 4790, pp. 363–377. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75560-9_27



Time and Gödel: Fuzzy Temporal Reasoning in PSPACE

Juan Pablo Aguilera^{1,2} , Martín Diéguez³ , David Fernández-Duque^{1,4} ,
and Brett McLean¹

¹ Department of Mathematics WE16, Ghent University, Ghent, Belgium
{Juan.Aguilera,David.FernandezDuque,Brett.McLean}@UGent.be

² Institute of Discrete Mathematics and Geometry, Vienna University of Technology,
Vienna, Austria

³ LERIA, University of Angers, Angers, France
Martin.DieguezLodeiro@univ-angers.fr

⁴ ICS of the Czech Academy of Sciences, Prague, Czech Republic

Abstract. We investigate a non-classical version of linear temporal logic whose propositional fragment is Gödel–Dummett logic (which is well known both as a superintuitionistic logic and a t-norm fuzzy logic). We define the logic using two natural semantics, a real-valued semantics and a bi-relational semantics, and show that these indeed define one and the same logic. Although this Gödel temporal logic does not have any form of the finite model property for these two semantics, we show that every falsifiable formula is falsifiable on a finite quasimodel, which yields decidability of the logic. We then strengthen this result by showing that this Gödel temporal logic is PSPACE-complete.

Keywords: Gödel–Dummett logic · linear temporal logic · intuitionistic logic · fuzzy logic · PSPACE-complete

1 Introduction

The importance of temporal logics and, independently, of fuzzy logics in computer science is well established. The potential usefulness of their combination is clear: for instance, it would provide a natural framework for the specification of programs dealing with vague data. Sub-classical linear temporal logics have been extensively studied in the context of here-and-there logic, which allows for three truth values and is the basis for temporal answer set programming [1, 2, 6].

One may, however, be concerned that infinite-valued temporal logics could lead to an explosion in computational complexity, as has been known to happen when combining fuzzy logic with transitive modal logics: these combinations are often undecidable [19], or decidable with only an exponential upper bound being known [7]. As we will see, this need not be the case: the combination of Gödel–Dummett logic with linear temporal logic, which we call Gödel temporal logic (GTL), remains PSPACE-complete, the minimal possible complexity given that

classical LTL embeds into it. This is true even when the logic is enriched with the dual implication [14], which has been argued in [8] to be useful for reasoning with incomplete or inconsistent information.

The decidability of GTL is already quite surprising, as it does not even enjoy the finite model property and its ‘modal companion’ $S4.3 \times LTL$ is not recursively axiomatisable [15]. In fact, GTL possesses two natural semantics, corresponding to whether it is viewed as a fuzzy logic or a superintuitionistic logic. As a fuzzy logic, propositions take values in $[0, 1]$, and truth values of compound propositions are defined using standard operations on the real line. As a superintuitionistic logic, models consist of bi-relational structures equipped with a partial order to interpret implication intuitionistically and a function to interpret the LTL tenses. Remarkably, the two semantics give rise to the same set of valid formulas, which offers two potential avenues to prove decidability of GTL via the finite model property. Unfortunately, as we will see, GTL does not enjoy the finite model property for either of these semantics.

In the setting of intuitionistic Gödel logics, it is often possible to prove decidability despite the lack of finite model property by considering modifications of the semantics (see e.g. [9]). For example, the logic GS4 does not enjoy the real-valued finite model property, but it does enjoy the bi-relational finite model property [7]. Since GTL does not enjoy either version of the finite model property, we instead introduce quasimodels, which do enjoy their own version of the finite model property. Quasimodels are not ‘true’ models in that the functionality of the ‘next’ relation is lost, but they give rise to standard bi-relational models by unwinding. Similar structures were used to prove upper complexity bounds for dynamic topological logic [11, 12] and intuitionistic temporal logic [13], but they are particularly effective in the setting of Gödel temporal logic, as they yield an optimal PSPACE upper bound.

Structure of Paper. In Sect. 2 we introduce the temporal language that we work with, and then introduce both the real-valued semantics and bi-relational semantics for Gödel temporal logic. In Sect. 3 we prove the equivalence of these two semantics, that is, that they yield the same validities (Theorem 1). In Sect. 5 we first note that we do not have a finite model property for either of these semantics. But then we define quasimodels (Definition 6), and in later sections we show that our Gödel temporal logic is sound and complete for the class of finite quasimodels. First, in Sect. 5, we show that Gödel temporal logic is sound for (all) quasimodels, constructing a bi-relational model from an arbitrary quasimodel by unwinding selected paths within the quasimodel. Then, in Sect. 6, given a bi-relational model falsifying a formula, we describe how to produce a finite (exponential in the length of the formula) quasimodel also falsifying the formula. This proves completeness of Gödel temporal logic for finite quasimodels and the decidability of Gödel temporal logic (Theorem 2). Finally, in Sect. 7, we refine this decidability result, showing that Gödel temporal logic is in fact PSPACE-complete (Theorem 6).

2 Syntax and Semantics

In this section we first introduce the temporal language we work with and then two possible semantics for this language: real-valued semantics and bi-relational semantics.

Fix a countably infinite set $\mathbb{P}$ of propositional variables. Then the **Gödel temporal language** $\mathcal{L}$ is defined by the grammar (in Backus–Naur form):

$$\varphi, \psi := p \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \Rightarrow \psi \mid \varphi \Leftarrow \psi \mid \bigcirc \varphi \mid \Diamond \varphi \mid \Box \varphi,$$

where $p \in \mathbb{P}$. Here, $\bigcirc$ is read as ‘next’, $\Diamond$ as ‘eventually’, and $\Box$ as ‘henceforth’. The connective $\Leftarrow$ is coimplication and represents the operator dual to implication [20]. We also use $\top$ as a shorthand for $p \Rightarrow p$ and $\perp$ as a shorthand for $p \Leftarrow p$, and also $\neg \varphi$ as a shorthand for $\varphi \Rightarrow \perp$.

We now introduce the first of our semantics for the Gödel temporal language: *real-valued semantics*, which views $\mathcal{L}$ as a fuzzy logic (enriched with temporal modalities). In the definition, $[0, 1]$ denotes the real unit interval.

Definition 1 (real-valued semantics). A **flow** is a pair $\mathcal{T} = (T, S)$, where T is a set and $S: T \rightarrow T$ is a function. A **real valuation** on $\mathcal{T}$ is a function $V: \mathcal{L} \times T \rightarrow [0, 1]$ such that, for all $t \in T$, the following equalities hold.

$V(\varphi \wedge \psi, t) = \min\{V(\varphi, t), V(\psi, t)\}$	$V(\varphi \vee \psi, t) = \max\{V(\varphi, t), V(\psi, t)\}$
$V(\varphi \Rightarrow \psi, t) = \begin{cases} 1 & \text{if } V(\varphi, t) \leq V(\psi, t) \\ V(\psi, t) & \text{otherwise} \end{cases}$	$V(\varphi \Leftarrow \psi, t) = \begin{cases} 0 & \text{if } V(\varphi, t) \leq V(\psi, t) \\ V(\varphi, t) & \text{otherwise} \end{cases}$
$V(\bigcirc \varphi, t) = V(\varphi, S(t))$	
$V(\Diamond \varphi, t) = \sup_{n < \omega} V(\varphi, S^n(t))$	$V(\Box \varphi, t) = \inf_{n < \omega} V(\varphi, S^n(t))$

A flow $\mathcal{T}$ equipped with a valuation V is a **real (Gödel temporal) model**.

Informally, $V(\varphi, t)$ records the degree of truth of φ at time t .

The second semantics, *bi-relational semantics*, views $\mathcal{L}$ as an intuitionistic logic (temporally enriched). Below, define $\vec{S}(w, t) = (w, S(t))$.

Definition 2 (bi-relational semantics). A **(Gödel temporal) bi-relational frame** is a quadruple $\mathcal{F} = (W, T, \leq, S)$ where $(W, \leq)$ is a linearly ordered set and (T, S) is a flow. A **bi-relational valuation** on $\mathcal{F}$ is a function $\llbracket \cdot \rrbracket: \mathcal{L} \rightarrow 2^{W \times T}$ such that, for each $p \in \mathbb{P}$, the set $\llbracket p \rrbracket$ is downward closed in its first coordinate, and the following equalities hold.

$\llbracket \varphi \wedge \psi \rrbracket = \llbracket \varphi \rrbracket \cap \llbracket \psi \rrbracket$	$\llbracket \varphi \vee \psi \rrbracket = \llbracket \varphi \rrbracket \cup \llbracket \psi \rrbracket$
$\llbracket \varphi \Rightarrow \psi \rrbracket = \{(w, t) \in W \times T \mid \forall v \leq w ((v, t) \in \llbracket \varphi \rrbracket \text{ implies } (v, t) \in \llbracket \psi \rrbracket)\}$	
$\llbracket \varphi \Leftarrow \psi \rrbracket = \{(w, t) \in W \times T \mid \exists v \geq w ((v, t) \in \llbracket \varphi \rrbracket \text{ and } (v, t) \notin \llbracket \psi \rrbracket)\}$	
$\llbracket \bigcirc \varphi \rrbracket = \vec{S}^{-1} \llbracket \varphi \rrbracket$	$\llbracket \Diamond \varphi \rrbracket = \bigcup_{n < \omega} \vec{S}^{-n} \llbracket \varphi \rrbracket$
$\llbracket \Box \varphi \rrbracket = \bigcap_{n < \omega} \vec{S}^{-n} \llbracket \varphi \rrbracket$	

A bi-relational frame $\mathcal{F}$ equipped with a valuation $\llbracket \cdot \rrbracket$ is a **(Gödel temporal) bi-relational model**.

Informally, $\{w \in W \mid (w, t) \in \llbracket \varphi \rrbracket\}$ records the degree of truth of φ at time t . This semantics combines standard semantics for the implications based on $\leq$ (read downward) and for the tenses based on S : for example, $(w, t) \in \llbracket \Diamond \varphi \rrbracket$ if and only if there exists $n \geq 0$ such that $(w, S^n(t)) \in \llbracket \varphi \rrbracket$. Note that, by structural induction, the valuation of *any* $\varphi \in \mathcal{L}$ is downward closed in its first coordinate, in the sense that if $(w, t) \in \llbracket \varphi \rrbracket$ and $v \leq w$, then $(v, t) \in \llbracket \varphi \rrbracket$.

For both semantics, validity of $\mathcal{L}$ -formulas is defined in the usual way.

Definition 3 (validity). *Given a real model $\mathcal{X} = (T, S, V)$ and a formula $\varphi \in \mathcal{L}$, we say that φ is **globally true** on $\mathcal{X}$, written $\mathcal{X} \models \varphi$, if for all $t \in T$ we have $V(\varphi, t) = 1$. Given a bi-relational model $\mathcal{X} = (\mathcal{F}, \llbracket \cdot \rrbracket)$ and a formula $\varphi \in \mathcal{L}$, we say that φ is **globally true** on $\mathcal{X}$, written $\mathcal{X} \models \varphi$, if $\llbracket \varphi \rrbracket = W \times T$.*

*If $\mathcal{X}$ is a flow or a bi-relational frame, we write $\mathcal{X} \models \varphi$ and say φ is **valid** on $\mathcal{X}$, if φ is globally true for every valuation on $\mathcal{X}$. If Ω is a class of flows, frames, or models, we say that $\varphi \in \mathcal{L}$ is **valid** on Ω if, for every $\mathcal{X} \in \Omega$, we have $\mathcal{X} \models \varphi$. If φ is not valid on Ω , it is **falsifiable** on Ω .*

We define the logic $\text{GTL}_{\mathbb{R}}$ to be the set of $\mathcal{L}$ -formulas that are valid over the class of all flows and the logic GTL to be the set of $\mathcal{L}$ -formulas that are valid over the class of all bi-relational frames.

3 Real-Valued Versus Bi-relational Validity

In this section, we show that an arbitrary $\mathcal{L}$ -formula is real valid if and only if it is bi-relationally valid. That is, $\text{GTL}_{\mathbb{R}} = \text{GTL}$. The two directions of this equivalence are Lemma 1 and Lemma 2 below.

Lemma 1. *Suppose that φ is an $\mathcal{L}$ -formula that is not real valid. Then φ is not bi-relationally valid.*

Proof. Let (T, S) be a flow, V a real valuation on (T, S) , and $t_0 \in T$ be such that $V(\varphi, t_0) < 1$. Since we are only concerned with the valuation at t_0 , we may assume without loss of generality that $T = \mathbb{N}$, $t_0 = 0$, and S is the successor function; in particular, that T is countable. Let X_0 be the set of all real numbers x such that $V(\psi, t) = x$ for some $\mathcal{L}$ -formula ψ and some $t \in T$. Thus, X_0 is a countable subset of $[0, 1]$. Let $X = (0, 1) \setminus X_0$.

We consider the bi-relational frame $\mathcal{F} = (X, T, \leq, S)$, where $\leq$ is the usual order on real numbers, and the bi-relational valuation $\llbracket \cdot \rrbracket$ given by

$$(x, t) \in \llbracket p \rrbracket \leftrightarrow V(p, t) > x. \quad (1)$$

We may then prove by induction that (1) holds for arbitrary $\mathcal{L}$ -formulas ψ and arbitrary $x \in X$ and $t \in T$. Then letting $x \in X$ be such that $V(\varphi, t_0) < x$ (this exists because X_0 is countable), we have $(x, t_0) \notin \llbracket \varphi \rrbracket$, so that $(\mathcal{F}, \llbracket \cdot \rrbracket)$ is a bi-relational countermodel for φ , as needed.

Lemma 2. *Suppose that φ is an $\mathcal{L}$ -formula that is not bi-relationally valid. Then φ is not real valid.*

Proof. Suppose that there is a bi-relational frame $\mathcal{F} = (W, T, \leq, S)$ and a valuation $\llbracket \cdot \rrbracket$ such that $(w, t_0) \notin \llbracket \varphi \rrbracket$. Since we are only concerned with the valuation of φ in (w, t) , we may as well assume that $T = \mathbb{N}$, that S is the successor function, and that $t_0 = 0$. By a routine downward Löwenheim–Skolem-type argument, we may assume that W is countable. We define a binary relation on $\mathcal{L} \times T$ by

$$(\psi, t_1) \leq (\chi, t_2) \leftrightarrow \forall w \in W (w, t_1) \in \llbracket \psi \rrbracket \rightarrow (w, t_2) \in \llbracket \chi \rrbracket,$$

that is, if the valuation of ψ at t_1 is contained in that of χ at t_2 . Since valuations of formulas are downward closed in their first coordinates, this reflexive transitive relation is total (any two elements are comparable). Let $(L, \leq)$ denote the linear order of all equivalence classes $[\psi, t]$ under $\leq$, and let 0 and 1 denote the equivalence classes of $[\perp, t]$ and $[\psi \Rightarrow \psi, t]$, respectively (this is independent of the choice of t and ψ).

We claim that this linear order respects valuations, in the sense that it satisfies the following properties for each $t \in T$:

- 0 is the least element and 1 is the greatest element of L ;
- $[\psi \wedge \chi, t] = \min\{[\psi, t], [\chi, t]\}$ and $[\psi \vee \chi, t] = \max\{[\psi, t], [\chi, t]\}$;
- $[\chi \Rightarrow \psi, t] = 1$ if $[\chi, t] \leq [\psi, t]$, and $[\chi \Rightarrow \psi, t] = [\psi, t]$ otherwise;
- $[\chi \Leftarrow \psi, t] = 0$ if $[\chi, t] \leq [\psi, t]$, and $[\chi \Leftarrow \psi, t] = [\chi, t]$ otherwise;
- $[\bigcirc \psi, t] = [\psi, t + 1]$;
- $[\bigtriangledown \psi, t] = \sup\{[\psi, t + n] \mid n \in \mathbb{N}\}$ and $[\bigtriangleup \psi, t] = \inf\{[\psi, t + n] \mid n \in \mathbb{N}\}$.

These properties easily follow from the definitions.

Now, $(L, \leq)$ is a countable linear order with endpoints, so it can be continuously (with respect to the order topology) embedded into the interval $[0, 1]$ in such a way that the images of 0 and 1 are, respectively, 0 and 1. Let ρ be such an embedding (we refer the reader to e.g., the proof of [4, Theorem 5.1] for an explicit construction of such a ρ). We define a real valuation V by setting $V(\psi, t) = \rho([\psi, t])$. By the properties above, V is indeed a real valuation and $V(\varphi, t_0) = \rho([\varphi, t_0]) < 1$.

Theorem 1. $\text{GTL}_{\mathbb{R}} = \text{GTL}$.

4 Labelled Systems and Quasimodels

Our decidability proof for the Gödel temporal validities GTL is based on (non-deterministic) quasimodels, originally introduced in [11] for *dynamic topological logic*, a classical predecessor of *intuitionistic temporal logic*, for which quasimodels were also used in [13]. As the bi-relational semantics makes evident, Gödel temporal logic is closely related to intuitionistic temporal logic. In this section we will introduce labelled spaces, labelled systems, and finally, quasimodels. Quasimodels can be viewed as a sort of temporally nondeterministic generalisation of bi-relational models.

Of course, many decidability proofs for classical modal logics are obtained via the finite model property, so it is worthwhile to first note that this strategy

cannot work for GTL because finite model properties do not hold. The finite model properties we define are of the form: *falsifiable* implies *falsifiable in a finite model*. It is worth remarking that in sub-classical logics, it is indeed the notion of falsifiability that is relevant, as it is falsifiability that is (by definition) the complement to validity. However, in view of our inclusion of coimplication, we may define $\sim\varphi \equiv \top \Leftarrow \varphi$, and then it is not hard to check that $\sim\varphi$ is *satisfiable* (in the sense of having a non-zero truth value) if and only if φ is falsifiable. Thus in view of the fact that our logic is (as we will see) PSPACE-complete, validity, satisfiability, and falsifiability are all inter-reducible.

Definition 4. The **strong finite model property** for GTL is the statement that if $\varphi \in \mathcal{L}$ is falsifiable on a bi-relational model, then it is falsifiable on a bi-relational model $\mathcal{F} = (W, T, \leq, S, \llbracket \cdot \rrbracket)$ where both W and T are finite.

The **order finite model property** for GTL is the statement that if $\varphi \in \mathcal{L}$ is falsifiable on a bi-relational model, then it is falsifiable on a bi-relational model $\mathcal{F} = (W, T, \leq, S, \llbracket \cdot \rrbracket)$ where W is finite.

The **temporal finite model property** for GTL is the statement that if $\varphi \in \mathcal{L}$ is falsifiable on a bi-relational model, then it is falsifiable on a bi-relational model $\mathcal{F} = (W, T, \leq, S, \llbracket \cdot \rrbracket)$ where T is finite.

Proposition 1. None of the finite model properties for GTL listed in Definition 4 hold. In particular, $\Diamond(p \Rightarrow \bigcirc p)$ is falsifiable, yet it is valid over the class of finite bi-relational models.



Fig. 1. Left: A bi-relational model falsifying $\Diamond(p \Rightarrow \bigcirc p)$; right: W and T are necessarily infinite. Time is represented by the horizontal axes and truth by the vertical.

Proof. To see that $\Diamond(p \Rightarrow \bigcirc p)$ is falsifiable, let $-\omega$ denote $\dots, -2, -1, 0$ with the usual ordering, and take the flow (ω, S) , where S is successor. Consider the model $(-\omega, \omega, \leq, S, \llbracket \cdot \rrbracket)$, where $\llbracket p \rrbracket = \{(-n, m) \in -\omega \times \omega \mid n \geq m\}$. Then at each $(-i, i)$, the formula p holds but $\bigcirc p$ does not. Hence at each $(0, i)$, the formula $p \Rightarrow \bigcirc p$ is falsified. Thus $\Diamond(p \Rightarrow \bigcirc p)$ is falsified at $(0, 0)$. (See Fig. 1 (left).)

To see that $\Diamond(p \Rightarrow \bigcirc p)$ can only be falsified on a model $(W, T, \leq, S, \llbracket \cdot \rrbracket)$ for which both W and T are infinite, suppose $\Diamond(p \Rightarrow \bigcirc p)$ is falsified at (w, t) . Then there must be a sequence $w \geq w_0 > w_1 > \dots$ such that for each $i \in \omega$, the formula p holds on each $(w_i, S^i(t))$ but not on $(w_i, S^{i+1}(t))$. This clearly forces W to be infinite, and by downward closure of $\llbracket \cdot \rrbracket$, it forces T to be infinite too. (See Fig. 1 (right).)

The same example as in Proposition 1 shows that under real-valued semantics it is also the case that some formulas can only be falsified on an infinite flow with infinitely many realised truth values, as falsification of $\Diamond(p \Rightarrow \bigcirc p)$ forces $V(p, t) > V(p, S(t))$ for all t .

Note that we have refuted all these finite model properties without using the $\Leftarrow$ connective, thus in fact proving the stronger result that the finite model properties fail for the $\Leftarrow$ -free fragment.

We now introduce the structures we will use to mitigate the failure of these finite model properties. Given a set $\Sigma \subseteq \mathcal{L}$ that is closed under subformulas, we say that $\Phi \subseteq \Sigma$ is a Σ -**type** if the following occur.

1. $\perp \notin \Sigma$.
2. If $\varphi \wedge \psi \in \Sigma$, then $\varphi \wedge \psi \in \Phi$ if and only if $\varphi, \psi \in \Phi$.
3. If $\varphi \vee \psi \in \Sigma$, then $\varphi \vee \psi \in \Phi$ if and only if $\varphi \in \Phi$ or $\psi \in \Phi$.
4. If $\varphi \Rightarrow \psi \in \Sigma$, then
 - (a) $\varphi \Rightarrow \psi \in \Phi$ implies that $\varphi \notin \Phi$ or $\psi \in \Phi$,
 - (b) $\psi \in \Phi$ implies that $\varphi \Rightarrow \psi \in \Phi$.
5. If $\varphi \Leftarrow \psi \in \Sigma$, then
 - (a) $\varphi \Leftarrow \psi \in \Phi$ implies $\varphi \in \Phi$,
 - (b) $\varphi \in \Phi$ and $\psi \notin \Phi$ implies that $\varphi \Leftarrow \psi \in \Phi$.

The set of Σ -types will be denoted by $\mathbb{T}_\Sigma$. Often we want Σ to be finite, in which case we write $\Sigma \in \mathcal{L}$ to indicate that $\Sigma \subseteq \mathcal{L}$ and Σ is finite and closed under subformulas.

A partially ordered set $(A, \leq)$ is **locally linear** if it is a disjoint union of linear posets.

In the following, we use the common notation $|\mathcal{S}|$ for the domain of a structure $\mathcal{S}$; this will not cause confusion with the same notation used for cardinalities.

Definition 5. Let $\Sigma \subseteq \mathcal{L}$ be closed under subformulas. A Σ -**labelled space** is a triple $\mathcal{W} = (|\mathcal{W}|, \leq_{\mathcal{W}}, \ell_{\mathcal{W}})$, where $(|\mathcal{W}|, \leq_{\mathcal{W}})$ is a locally linear poset and $\ell: |\mathcal{W}| \rightarrow \mathbb{T}_\Sigma$ an inversely monotone function (in the sense that $w \leq v$ implies $\ell_{\mathcal{W}}(w) \supseteq \ell_{\mathcal{W}}(v)$) such that for all $w \in |\mathcal{W}|$

- whenever $\varphi \Rightarrow \psi \in \Sigma \setminus \ell_{\mathcal{W}}(w)$, there is $v \leq w$ such that $\varphi \in \ell_{\mathcal{W}}(v)$ and $\psi \notin \ell_{\mathcal{W}}(v)$;
- whenever $\varphi \Leftarrow \psi \in \ell_{\mathcal{W}}(w)$, there is $v \geq w$ such that $\varphi \in \ell_{\mathcal{W}}(v)$ and $\psi \notin \ell_{\mathcal{W}}(v)$.

The Σ -labelled space $\mathcal{W}$ **falsifies** $\varphi \in \mathcal{L}$ if $\varphi \in \Sigma \setminus \ell_{\mathcal{W}}(w)$ for some $w \in W$. The **height** of $\mathcal{W}$ is the supremum of all n such that there is a chain $w_1 <_{\mathcal{W}} w_2 <_{\mathcal{W}} \dots <_{\mathcal{W}} w_n$.

If $\mathcal{W}$ is a labelled space, elements of $|\mathcal{W}|$ will sometimes be called **worlds**. When clear from context we will omit subscripts and write, for example, $\leq$ instead of $\leq_{\mathcal{W}}$.

We now enrich labelled spaces with a relation capturing temporal information.

A **convex relation** between posets $(A, \leq_A)$ and $(B, \leq_B)$ is a binary relation $R \subseteq A \times B$ such that for each $x \in A$ the image set $\{y \in B \mid x R y\}$ is convex with respect to $\leq_B$, and for each $y \in B$ the preimage set $\{x \in A \mid x R y\}$ is convex with respect to $\leq_A$. The relation R is **fully confluent** if it validates the four following conditions:

forth-down if $x \leq_A x' R y'$ there is y such that $x R y \leq_B y'$,
forth-up if $x' \geq_A x R y$ there is y' such that $x' R y' \geq_B y$,
back-down if $x' R y' \geq_B y$ there is x such that $x' \geq_A x R y$,
back-up if $x R y \leq_B y'$ there is x' such that $x \leq_A x' R y'$.

Definition 6. Let $\Sigma \subseteq \mathcal{L}$ be closed under subformulas. Suppose that $\Phi, \Psi \in \mathbb{T}_\Sigma$. The ordered pair (Φ, Ψ) is **sensible** if

1. for all $\circ\varphi \in \Sigma$, we have $\circ\varphi \in \Phi$ if and only if $\varphi \in \Psi$,
2. for all $\diamond\varphi \in \Sigma$, we have $\diamond\varphi \in \Phi$ if and only if $\varphi \in \Phi$ or $\diamond\varphi \in \Psi$,
3. for all $\square\varphi \in \Sigma$, we have $\square\varphi \in \Phi$ if and only if $\varphi \in \Phi$ and $\square\varphi \in \Psi$.

A pair (w, v) of worlds in a labelled space $\mathcal{W}$ is **sensible** if $(\ell(w), \ell(v))$ is sensible. A relation $R \subseteq |\mathcal{W}| \times |\mathcal{W}|$ is **sensible** if every pair in R is sensible. Further, R is **ω -sensible** if it is serial (that is, for each $w \in |\mathcal{W}|$ there exists $v \in |\mathcal{W}|$ with $w R v$) and

- whenever $\diamond\varphi \in \ell(w)$, there are $n \geq 0$ and v such that $w R^n v$ and $\varphi \in \ell(v)$;
- whenever $\square\varphi \notin \ell(w)$, there are $n \geq 0$ and v such that $w R^n v$ and $\varphi \notin \ell(v)$.

A **labelled system** is a labelled space $\mathcal{W}$ equipped with a fully confluent convex sensible relation $R_\mathcal{W} \subseteq |\mathcal{W}| \times |\mathcal{W}|$. If moreover $R_\mathcal{W}$ is ω -sensible, we say that $\mathcal{W}$ is a **Σ -quasimodel**.

Any bi-relational model can be regarded as a Σ -quasimodel. If $\mathcal{X} = (W, T, \leq, S, \llbracket \cdot \rrbracket)$ is a bi-relational model and $x \in W \times T$, we can assign a Σ -type $\ell_\mathcal{X}(x)$ to x given by $\ell_\mathcal{X}(x) = \{\psi \in \Sigma \mid x \in \llbracket \psi \rrbracket\}$. We also set $R_\mathcal{X} = \{((w, t), (w, S(t))) \mid w \in W, t \in T\}$; it is obvious that $R_\mathcal{X}$ is ω -sensible. Henceforth we will tacitly identify $\mathcal{X}$ by its associated Σ -quasimodel.

5 From Quasimodels to Bi-relational Models

If $\mathcal{Q} = (|\mathcal{Q}|, \leq_\mathcal{Q}, \ell_\mathcal{Q}, R_\mathcal{Q})$ is a quasimodel, then we may not view $\mathcal{Q}$ directly as a bi-relational model for the primary reason that $R_\mathcal{Q}$ is not necessarily a function. However, we can extract bi-relational models from quasimodels via an unwinding construction. More precisely, given $\Sigma \in \mathcal{L}$, suppose φ is falsified on the Σ -quasimodel $\mathcal{Q}$. In this section we show how to obtain from $\mathcal{Q}$ a bi-relational model $\tilde{\mathcal{Q}}$ falsifying φ . We call the resulting bi-relational model a *limit model* of $\mathcal{Q}$. This proves GTL is sound for the class of quasimodels.

The general idea for determinising $\mathcal{Q}$ is to consider infinite paths on $\mathcal{Q}$ as points in the limit model. However, we will only select paths $\vec{w}$ with the property that, if $\diamond\varphi$ occurs in $\vec{w}$, then φ must also occur at a later time, with a dual condition for $\square$. These are the realising paths of $\mathcal{Q}$.

Definition 7. A *path* in a Σ -quasimodel $\mathcal{Q} = (|\mathcal{Q}|, \leq_{\mathcal{Q}}, \ell_{\mathcal{Q}}, R_{\mathcal{Q}})$ is any sequence $(w_i)_{i < \alpha}$ with $\alpha \leq \omega$ such that $w_i R w_{i+1}$ whenever $i + 1 < \alpha$. An infinite path $\vec{w} = (w_i)_{i < \omega}$ is **realising** if for all $i < \omega$

- for all $\Diamond\psi \in \ell(w_i)$, there exists $j \geq i$ such that $\psi \in \ell(w_j)$,
- for all $\Box\psi \in \Sigma \setminus \ell(w_i)$, there exists $j \geq i$ such that $\psi \in \Sigma \setminus \ell(w_j)$.

Denote the set of realising paths by $|\vec{\mathcal{Q}}|$, and let $(v_i)_{i < \omega} \leq (w_i)_{i < \omega}$ if and only if $v_i \leq w_i$ for all $i < \omega$. The worlds of the limit model will be a linearly ordered (with respect to $\leq$) subset of $\vec{\mathcal{Q}}$.

Given our $\Sigma \in \mathcal{L}$ and a formula φ falsified in $\mathcal{Q}$, the limit model $\vec{\mathcal{Q}}_{\varphi}$ will be of the form $(W, \omega, \leq, S, \llbracket \cdot \rrbracket)$, where $W \subseteq |\vec{\mathcal{Q}}|$, the flow function $S: \omega \rightarrow \omega$ is successor, and $\llbracket p \rrbracket = \{(\vec{w}, i) \mid p \in \ell(w_i)\}$ (extended to compound formulas in accordance with Definition 2). We now describe how to select the linearly ordered subset of realising paths W .

Definition 8. A *finite grid* is a linearly ordered finite set of paths in $\mathcal{Q}$ of some uniform length $k < \omega$ (with the pointwise ordering). A finite grid P' with paths of length k' , **extends** a finite grid P with paths of length k , if $k' > k$ and $P \subseteq P'|_k$, where $P'|_k$ is the restriction of the paths in P' to their initial k -length segments.

Definition 9 (defects). Let P be a finite grid.

- A **$\Diamond$ -defect** of P is a pair $((w_i)_{i < k}, \Diamond\varphi) \in P \times \Sigma$ such that $\Diamond\varphi \in \ell(w_{k-1})$, but $\varphi \notin \ell(w_{k-1})$.
- A **$\Box$ -defect** of P is a pair $((w_i)_{i < k}, \Box\varphi) \in P \times \Sigma$ such that $\Box\varphi \notin \ell(w_{k-1})$, but $\varphi \in \ell(w_{k-1})$.
- An **$\Rightarrow$ -defect** of P is a triple $((w_i)_{i < k}, \varphi \Rightarrow \psi, j) \in P \times \Sigma \times k$ such that $\varphi \Rightarrow \psi \notin \ell(w_j)$, but there is no $(v_i)_{i < k} \leq (w_i)_{i < k}$ also in P such that $\varphi \in \ell(v_j)$, but $\psi \notin \ell(v_j)$.
- A **$\Leftarrow$ -defect** of P is a triple $((w_i)_{i < k}, \varphi \Leftarrow \psi, j) \in P \times \Sigma \times k$ such that $\varphi \Leftarrow \psi \in \ell(w_j)$, but there is no $(v_i)_{i < k} \geq (w_i)_{i < k}$ also in P such that $\varphi \in \ell(v_j)$, but $\psi \notin \ell(v_j)$.
- A **seriality defect** is a path $(w_i)_{i < k} \in P$.

Note that because Σ is finite, any finite grid has a finite number of defects. We select the set $W \subseteq |\vec{\mathcal{Q}}|$ as follows. We maintain a (finite) first-in-first-out queue D of defects and a finite grid P that is extended each time we process a defect. (We will ensure that the constituents of D continue to be defects of the grid.) Choose some $w \in |\mathcal{Q}|$ falsifying φ . We initialise the grid to the single sequence (w) (of length 1) and add all defects of this grid to D .

Using the properties of quasimodels, we may extend P to a new grid P' , so that the defect at the head of the queue D is not a defect of P' . (Because of seriality defects, D is never empty.) The construction of P' is realized via a case-by-case analysis: e.g. for a $\Diamond$ -defect $((w_i)_{i < k}, \Diamond\varphi)$, extend $(w_i)_{i < k}$ to a longer path $(w_i)_{i < k'}$ so that $\varphi \in \ell(w_{k'-1})$; such a path exists by ω -sensibility.

Then extend all other paths in P to also have length k' using the confluence properties; the extended paths form the grid P' . The most interesting case is a $\Rightarrow$ -defect $((w_i)_{i < k}, \varphi \Rightarrow \psi, j)$. Choose $v_j \in |\mathcal{Q}|$ so that $v_j < w_j$ and $\varphi \in \ell(v_j)$, but $\psi \notin \ell(v_j)$. Let $(u_i)_{i < k} \in P$ be minimum subject to $u_j > v_j$ and $(t_i)_{i < k} \in P$ be maximum subject to $t_j < v_j$, assuming it exists. (The case where $(t_i)_{i < k}$ does not exist is similar but simpler.) Now show one can complete the sequence $(v_i)_{i < k}$ so that $(t_i)_{i < k} < (v_i)_{i < k} < (u_i)_{i < k}$ (left as an exercise). Then $P' = P \cup \{(v_i)_{i < k}\}$.

Next we must update D so that it contains defects of P' , and all the elements of D that are not defects of P' are removed (including the head). Finally, we update $P := P'$.

We set W to be the limit of this sequence of finite grids. More precisely, let the sequence of grids be $P_0, P_1, \dots$, containing paths of length $k_0, k_1, \dots$, respectively; then W is the set of paths $(w_i)_{i < \omega}$ for which there exists N such that for all $n > N$ the initial segment $(w_i)_{i < k_n}$ is in P_n . (This gives us the limit we would expect and forbids ‘diagonal’ sequences whose initial segments appear in $P_0, P_1, \dots$ merely infinitely often.)

By construction, $(W, \leq)$ is a linearly ordered set, and for each $p \in \mathbb{P}$, the set $\llbracket p \rrbracket$ is downward closed in its first coordinate. Thus the limit model $\vec{\mathcal{Q}}_\varphi = (W, \omega, \leq, S, \llbracket \cdot \rrbracket)$ indeed defines a bi-relational model. Of course $\vec{\mathcal{Q}}_\varphi$ is only useful if $\llbracket \cdot \rrbracket$ ‘matches’ with ℓ on *all* formulas of Σ , not just propositional variables. Fortunately, this turns out to be the case.

Lemma 3. *Let $\Sigma \in \mathcal{L}$, $\mathcal{Q}$ be a Σ -quasimodel, $\varphi, \psi \in \Sigma$, and $\vec{\mathcal{Q}}_\varphi = (W, \omega, \leq, S, \llbracket \cdot \rrbracket)$ be as described above. Then $\llbracket \psi \rrbracket = \{(\vec{w}, i) \mid \psi \in \ell(w_i)\}$.*

Proof. The proof goes by standard induction of formulas. The induction steps for $\wedge, \vee$, and $\perp$ are immediate. The cases for $\Rightarrow, \circ, \diamond$, and $\square$ follow straightforwardly from the construction of W , because every defect is eventually eliminated. Indeed every w_i in the limit is also in some P_n , at which point all the defects associated to w_i are in the finite first-in-first-out queue D ; thus they will eventually reach the front of D and be eliminated.

We obtain the main result of this section, which in particular implies that GTL is sound for the class of quasimodels.

Proposition 2. *Let $\Sigma \in \mathcal{L}$ and $\mathcal{Q}$ be any Σ -quasimodel, and suppose φ is falsified on $\mathcal{Q}$. Then there exists a bi-relational model $\vec{\mathcal{Q}}_\varphi$ that falsifies φ .*

6 From Bi-relational Models to Finite Quasimodels

As we noted earlier, every bi-relational model can be naturally viewed as a quasimodel. However, we wish to show that, given a finite and subformula-closed Σ , we can from each bi-relational model $\mathcal{X}$ produce a *finite* quasimodel falsifying exactly the same formulas from Σ as $\mathcal{X}$. In this section we do just this by transforming $\mathcal{X}$ in two steps. First, we will take a bisimulation quotient to obtain a finite Σ -labelled space equipped with a fully confluent ω -sensible relation. The

second step will be to extend the ω -sensible relation to be convex, yielding a finite quasimodel.

We describe the quotient explicitly, noting afterwards that it is a particular type of bisimulation quotient.

Let Σ be a subformula-closed subset of $\mathcal{L}$, and let $\mathcal{X} = (W, T, \leq, S, \llbracket \cdot \rrbracket)$ be a bi-relational model. For $x \in W \times T$, define $\ell_{\mathcal{X}}(x)$ by $\ell_{\mathcal{X}}(x) = \{\psi \in \Sigma \mid x \in \llbracket \psi \rrbracket\}$, and define $L_{\mathcal{X}}(x) = \{\ell(y) \mid \pi_2(x) = \pi_2(y)\}$, where $\pi_2: W \times T \rightarrow T$ is the projection $(w, t) \mapsto t$. We define the equivalence relation $\sim$ on $W \times T$ by

$$x \sim y \iff (\ell(x), L(x)) = (\ell(y), L(y)).$$

If Σ is finite, then clearly $(W \times T)/\sim$ is finite.

Note that $\sim$ is the largest relation that is simultaneously a bisimulation with respect to the relations $\leq$ and $\geq$, with Σ treated as the set of atomic propositions that bisimilar worlds must agree on.

Now define a partial order $\leq_{\mathcal{Q}}$ on the equivalence classes $(W \times T)/\sim$ of $\sim$ by

$$[x] \leq_{\mathcal{Q}} [y] \iff L(x) = L(y) \text{ and } \ell(x) \supseteq \ell(y),$$

noting that this is well-defined and is indeed a partial order.

Since each set $L(x)$ can be linearly ordered by inclusion and $\ell(x) \in L(x)$, the poset $((W \times T)/\sim, \leq_{\mathcal{Q}})$ is a disjoint union of linear posets. By defining $\ell_{\mathcal{Q}}$ by $\ell_{\mathcal{Q}}([x]) = \ell(x)$ we obtain a Σ -labelled space $((W \times T)/\sim, \leq_{\mathcal{Q}}, \ell_{\mathcal{Q}})$; it is not hard to check that this labelling is inversely monotone and that the clauses for $\Rightarrow$ and $\Leftarrow$ hold with this labelling.

Now define the binary relation $R_{\mathcal{Q}}$ on $(W \times T)/\sim$ to be the smallest relation such that $[(w, t)] R_{\mathcal{Q}} [(w, S(t))]$, for all $(w, t) \in W \times T$.

Lemma 4. *The relation $R_{\mathcal{Q}}$ is fully confluent and ω -sensible.*

As promised, we now have a Σ -labelled space equipped with a fully confluent ω -sensible relation. We now transform this labelled space into a Σ -quasimodel by making the equipped relation convex by fiat.

Define $R_{\mathcal{Q}}^+$ by $X R_{\mathcal{Q}}^+ Y$ if and only if there exist $X_1 \leq_{\mathcal{Q}} X \leq_{\mathcal{Q}} X_2$ and $Y_1 \leq_{\mathcal{Q}} Y \leq_{\mathcal{Q}} Y_2$ such that $X_2 R_{\mathcal{Q}} Y_1$ and $X_1 R_{\mathcal{Q}} Y_2$. Now define $\mathcal{Q} = ((W \times T)/\sim, \leq_{\mathcal{Q}}, \ell_{\mathcal{Q}}, R_{\mathcal{Q}}^+)$.

Lemma 5. *The structure $\mathcal{Q}$ is a Σ -quasimodel.*

Proof. We already know that $((W \times T)/\sim, \leq_{\mathcal{Q}}, \ell_{\mathcal{Q}})$ is a Σ -labelled space. First we must check $R_{\mathcal{Q}}^+$ is still fully confluent and ω -sensible.

For the forth-down condition, suppose $X \leq_{\mathcal{Q}} X' R_{\mathcal{Q}}^+ Y'$. Then by the definition of $R_{\mathcal{Q}}^+$, there are some $X_2 \geq_{\mathcal{Q}} X'$ and $Y_1 \leq_{\mathcal{Q}} Y'$ such that $X_2 R_{\mathcal{Q}} Y_1$. Since $X \leq_{\mathcal{Q}} X' \leq_{\mathcal{Q}} X_2$, by the forth-down condition for $R_{\mathcal{Q}}$ there is some $Y \leq_{\mathcal{Q}} Y_1$ with $X R_{\mathcal{Q}} Y$ and therefore $X R_{\mathcal{Q}}^+ Y$. Since $Y \leq_{\mathcal{Q}} Y_1 \leq_{\mathcal{Q}} Y'$, we are done. The proof that the forth-up condition holds is just the order dual of that for forth-down. The proofs of the back-down and back-up conditions are similar.

To see that R_Q^+ is sensible, suppose $X R_Q^+ Y$ and that $\bigcirc\varphi \in \Sigma$. Take $X_1 \leq_Q X \leq_Q X_2$ and $Y_1 \leq_Q Y \leq_Q Y_2$ such that $X R_Q Y_1$. Then

$$\begin{aligned} \bigcirc\varphi \in \ell_Q(X) &\implies \bigcirc\varphi \in \ell_Q(X_1) &\implies \varphi \in \ell_Q(Y_2) &\implies \varphi \in \ell_Q(Y) \\ &\implies \varphi \in \ell_Q(Y_1) &\implies \bigcirc\varphi \in \ell_Q(X_2) &\implies \bigcirc\varphi \in \ell_Q(X), \end{aligned}$$

so $\bigcirc\varphi \in \ell_Q(X) \iff \varphi \in \ell_Q(Y)$. The $\diamond$ and $\square$ cases are similar. It is now clear that R_Q^+ is ω -sensible since $R_Q \subseteq R_Q^+$, and the three conditions for a sensible relation to be ω -sensible are all monotone.

Finally, we show that R_Q^+ is convex. Firstly, for the image condition, if $X R_Q^+ Y_1$ and $X R_Q^+ Y_2$ with $Y_1 \leq_Q Y \leq_Q Y_2$, then by the definition of R_Q^+ we can find $X_2 \geq_Q X$ and $Y'_1 \leq_Q Y_1$ with $X_2 R_Q Y'_1$, and similarly $X_1 \leq_Q X$ and $Y'_2 \geq_Q Y_2$ with $X_1 R_Q Y'_2$. Since then $X_1 \leq_Q X \leq_Q X_2$ and $Y'_1 \leq_Q Y \leq_Q Y'_2$, by the definition of R_Q^+ we conclude that $X R_Q^+ Y$. The preimage condition is completely analogous. This completes the proof that Q is a Σ -quasimodel.

Lemma 6. *Let $\varphi \in \Sigma$. Then $\mathcal{X}$ falsifies φ if and only if Q falsifies φ .*

Proof. We have: $\mathcal{X}$ falsifies φ if and only if $\exists(w, t) \in (W \times T) \setminus \llbracket \varphi \rrbracket$ if and only if $\exists[(w, t)] \in (W \times T)/\sim$ with $\varphi \in \Sigma \setminus \ell_Q([(w, t)])$ if and only if Q falsifies φ .

In order to use Lemma 6 to prove decidability, we need to compute a bound on the size of the quasimodel Q in terms of the size of Σ , when Σ is finite.

Lemma 7. *Suppose Σ is finite, and write $|\Sigma|$ for its cardinality. Then the height of Q is bounded by $|\Sigma| + 1$, and the cardinality of the domain $(W \times T)/\sim$ of Q is bounded by $(|\Sigma| + 1) \cdot 2^{|\Sigma|(|\Sigma|+1)+1}$.*

Proof. Each element of the domain of Q is a pair (ℓ, L) where L is a (nonempty) subset of $\wp\Sigma$ and $\ell \in L$. Since L is linearly ordered by inclusion, it has height at most $|\Sigma| + 1$. There are $(2^{|\Sigma|})^i$ subsets of $\wp\Sigma$ of size i , so there are at most $\sum_{i=1}^{|\Sigma|+1} (2^{|\Sigma|})^i$ distinct L . The sum is bounded by $2^{|\Sigma|(|\Sigma|+1)+1}$. The factor of $|\Sigma| + 1$ corresponds to choice of an $\ell \in L$, for each L .

Thus we have an exponential bound on the size of Q . Hence any falsifiable formula is falsifiable on an effectively bounded quasimodel, and it follows that GTL is decidable.

Theorem 2. *The logic $\text{GTL}_{\mathbb{R}}$ of $\mathcal{L}$ -formulas that are valid on all flows and the logic GTL of $\mathcal{L}$ -formulas that are valid on all bi-relational frames are equal and decidable.*

7 PSPACE Completeness

Note that the proof of Theorem 2 yields only a NEXPTIME upper bound. In this section, we prove that this can be improved to PSPACE.

First, we recall that the validity problem for LTL is PSPACE-complete [17, Theorem 4.1]. Thus to prove PSPACE-hardness of the GTL validity problem, it suffices to give a reduction from LTL validity to GTL validity. Consider the (negative) translation “ $(\cdot)^\bullet$ ” [18] defined as follows: 1) $p^\bullet = \neg\neg p$, with p a propositional variable and 2) homomorphic for the rest of operators.

In what follows we may assume that $T = \mathbb{N}$, equipped with the standard successor function. The next proposition may be checked by a standard structural induction on φ .

Given an LTL model $(\mathbb{N}, S, V)$, we associate a ‘crisp’ Gödel model $(\mathbb{N}, S, V')$ where $V'(p, t) = 1$ if $t \in V(p)$ and 0 otherwise. We can prove the following result by structural induction.

Proposition 3. *For any $\varphi \in \mathcal{L}$ and for all $t \in \mathbb{N}$, 1) if $(\mathbb{N}, S, V), t \models \varphi$ then $V'(\varphi, t) = 1$, and 2) if $(\mathbb{N}, S, V), t \not\models \varphi$ then $V'(\varphi, t) = 0$.*

Conversely, given a real Gödel temporal model $(\mathbb{N}, S, V')$, we associate the crisp Gödel temporal model $(\mathbb{N}, S, V)$ by fixing $V(p, t) = V'(\neg\neg p, t) \in \{0, 1\}$.

Proposition 4. *For any $\varphi \in \mathcal{L}$ and any $t \in \mathbb{N}$, we have $V'(\varphi^\bullet, t) = V(\varphi, t)$.*

Corollary 1. *For any $\varphi \in \mathcal{L}$, we have $\text{LTL} \models \varphi$ if and only if $\text{GTL} \models \varphi^\bullet$.*

Proof. For the left-to-right direction, assume by contraposition that $\text{GTL} \not\models \varphi^\bullet$. Therefore there exists a Gödel temporal model $(\mathbb{N}, S, V')$ and $t \in \mathbb{N}$ such that $V'(\varphi^\bullet, t) \neq 1$. By Proposition 4 there exists a crisp Gödel temporal model $(\mathbb{N}, S, V)$ such that $V(\varphi, t) \neq 1$. This latter model can be viewed as an LTL model with $(\mathbb{N}, S, V), t \not\models \varphi$, since the real-valued semantics coincide with classical truth definitions when values are in $\{0, 1\}$. Therefore, $\text{LTL} \not\models \varphi$.

Conversely, assume by contraposition that $\text{LTL} \not\models \varphi$. This means that there exists an LTL model $(\mathbb{N}, S, V)$ and $t \in \mathbb{N}$ such that $(\mathbb{N}, S, V), t \not\models \varphi$. Then for the crisp Gödel temporal model $(\mathbb{N}, S, V')$ as defined in Proposition 3 we have $V'(\varphi, t) = 0$. As a consequence $\text{GTL} \not\models \varphi^\bullet$.

For the membership of PSPACE, we adapt the proof of LTL satisfiability from [10] to the case of GTL. Say that an **ultimately periodic quasimodel** $\mathcal{Q} = (W, \leq, \ell, S)$ such that there is a flow (T, f) with $T = \{0, \dots, i+l\}$ with $f(k) = k+1$ for $k < i+l$, $f(i+l) = i$, and a ‘projection’ function $\pi: W \rightarrow T$ such that each $\pi^{-1}(t)$ is an entire linear component of W and $w S v$ implies that $\pi(v) = f(\pi(w))$.

In other words, $\mathcal{Q}$ has an underlying flow T consisting of an initial segment followed by a loop, and each $t \in T$ is assigned a linear order $\pi^{-1}(t)$, which we may also write as W_t . Every falsifiable formula is falsifiable in a quasimodel of this form.

Theorem 3 (ultimately periodic quasimodel property). *Every falsifiable $\mathcal{L}$ -formula is falsifiable in an ultimately periodic quasimodel of height bounded by $|\Sigma| + 1$.*

Proof. We sketch the construction. By Lemma 7, if φ is falsifiable, it is falsifiable on some quasimodel $\mathcal{Q}' = (W', \leq', \ell', S')$ of height at most $|\Sigma| + 1$. Choose $w_0 \in W'$ such that $\varphi \notin \ell_0(w_0)$, and let W_0 be the linear component of w_0 (i.e., $W_0 = \{v \in W \mid v \leq w_0 \text{ or } w_0 \leq v\}$) and $\leq_0$ be $\leq'$ restricted to W_0 . By a priority method similar to that of Sect. 5, we define an infinite sequence $(W_0, \leq_0), (W_1, \leq_1), \dots$, and sensible relations $S_k \subseteq W_k \times W_{k+1}$, such that $\mathcal{Q}^\infty = (W^\infty, \leq^\infty, \ell^\infty, S^\infty)$ is a quasimodel, where $W^\infty = \bigsqcup_{k < \omega} W_k$ ($\bigsqcup$ denotes a disjoint union), $\leq^\infty = \bigsqcup_{k < \omega} \leq_k$, and so on.

Note that there are at most $2^{|\Sigma|(|\Sigma|+1)}$ possible choices of W_k , since each W_k consists of at most $|\Sigma| + 1$ types, and there are at most $2^{|\Sigma|}$ types. This in particular implies that some W_i repeats infinitely often. Let l be such that $W_{i+l} = W_i$ and every defect of W_i has been realised before W_{i+l} ; such an l exists because W_i has finitely many $\diamond$ or $\square$ defects. We define $\mathcal{Q} = (W, \leq, \ell, S)$ to be the restriction of $\mathcal{Q}^\infty$ to $\bigcup_{k=0}^{i+l-1} W_k$, but with S redefined on W_{i+l-1} so that it maps to W_i .

It remains to check that $\mathcal{Q}$ is a quasimodel. We only check that it is ω -sensible, as the other properties are easy to check. Consider the case of $\diamond\psi \in \ell(w)$ (the case $\square\psi \in \Sigma \setminus \ell(w)$ is analogous). Then $w \in W_k$ for some k , which means that for some j (namely, $j = i + l - k$), there is $v \in W_i$ such that $w S^j v$. By construction, there are some k' and some u such that $v S^{k'} u$ and $\psi \in \ell(u)$. Then $w S^{k+k'} u$ and $\psi \in \ell(u)$, as needed.

Ultimately periodic models can be represented using *moments*.

Definition 10. A Σ -*moment* is a sequence of the form $\mathbf{m} = (\mathbf{m}_0, \dots, \mathbf{m}_m)$, where

1. each $\mathbf{m}_i$ is a Σ -type,
2. $\mathbf{m}_i \supsetneq \mathbf{m}_{i+1}$ for $i < n - 1$,
3. for every $\varphi \Rightarrow \psi \in \Sigma \setminus \mathbf{m}_i$ there is some $j \leq i$ with $\varphi \in \mathbf{m}_j$ but $\psi \notin \mathbf{m}_j$,
4. for every $\varphi \Leftarrow \psi \in \mathbf{m}_i$ there is some $j \geq i$ with $\varphi \in \mathbf{m}_j$ but $\psi \notin \mathbf{m}_j$.

We write $|\mathfrak{M}|$ for the set $\{\mathbf{m}_0, \dots, \mathbf{m}_m\}$. The set of Σ -moments is denoted M_Σ .

We define the labelled space $(\mathbf{m}_0, \dots, \mathbf{m}_m) + (\mathbf{n}_0, \dots, \mathbf{n}_n)$ to be the parallel sum of the two linear posets $(\mathbf{m}_0, \dots, \mathbf{m}_m)$ and $(\mathbf{n}_0, \dots, \mathbf{n}_n)$ with labelling given by the identity.

Definition 11. Say the moment $\mathbf{n}$ is a **temporal successor** of $\mathbf{m}$, denoted $\mathbf{m} S_\Sigma \mathbf{n}$, if there exists a fully confluent convex sensible relation $R \subseteq |\mathbf{m}| \times |\mathbf{n}|$ on the labelled space $\mathbf{m} + \mathbf{n}$.

Definition 12. We define $\mathbb{M}_\Sigma = (M_\Sigma, S_\Sigma)$.

Because of condition (2), if Σ is finite then so is $\mathbb{M}_\Sigma$.

Definition 13. A small falsifiability witness for an $\mathcal{L}$ -formula φ is a finite sequence of moments $\mathbf{m}^0, \dots, \mathbf{m}^i, \dots, \mathbf{m}^{i+l}$ of subsets of Σ with a distinguished position i and binary relations $S_j \subseteq |\mathbf{m}^j| \times |\mathbf{m}^{j+1}|$ for each $j < i + l$ such that

- (A) $\varphi \in \mathbf{m}_0^0$ and $\mathbf{m}^i = \mathbf{m}^{i+l}$,
- (B) S_j is a sensible relation on $\mathbf{m}^j + \mathbf{m}^{j+1}$,
- (C) If $\Diamond\psi \in \mathbf{m}_j^i$ then there are $l' < l$ and a sequence $(j_k)_{k \leq l'}$ with $j_0 = j$ such that $\mathbf{m}_{j_k}^{i+k} S_{i+k} \mathbf{m}_{j_{k+1}}^{i+k+1}$ if $k < l'$ and $\psi \in \mathbf{m}_{j_{l'}}^{i+l'}$,
- (D) If $\Box\psi \in \Sigma \setminus \mathbf{m}_j^i$ then there are $l' < l$ and a sequence $(j_k)_{k \leq l'}$ with $j_0 = j$ such that $\mathbf{m}_{j_k}^{i+k} S_{i+k} \mathbf{m}_{j_{k+1}}^{i+k+1}$ if $k < l'$ and $\psi \notin \mathbf{m}_{j_{l'}}^{i+l'}$.

Theorem 3 implies that if an $\mathcal{L}$ -formula is falsifiable then it has a small falsifiability witness. Moreover, the converse is also true. As a consequence, we obtain an equivalence between the existence of an infinite structure (a model of φ) and the existence of a finite structure (a small falsifiability witness) for a given $\mathcal{L}$ -formula φ .

Theorem 4. *An $\mathcal{L}$ -formula is falsifiable if and only if it has a small falsifiability witness.*

Proof. For the left-to-right direction, assume that the formula φ is falsifiable. By Theorem 3, there exists an ultimately periodic quasimodel $\mathcal{Q} = (W, \leq, \ell, S)$ such that $W = \bigcup_{k < i+l} W_k$ and $\varphi \in \ell(w)$ for some $w \in W_0$. For each W_k , let $W_k = \{v_0^k \dots v_{m_k}^k\}$ in increasing order, and let $\mathbf{m}^k = (\ell(v_0^k) \dots \ell(v_{m_k}^k))$ (deleting repeating types if needed). It is easy to check that the sequence $\mathbf{m}^0, \dots, \mathbf{m}^i, \dots, \mathbf{m}^{i+l}$ yields a small falsifiability witness.

Conversely, we will show that if a formula has a small falsifiability witness $\mathbf{m}^0, \dots, \mathbf{m}^i, \dots, \mathbf{m}^{i+l}$ then it is falsifiable. Write $\mathbf{m}^k = (\mathbf{m}_0^k, \dots, \mathbf{m}_{m_k}^k)$ and consider the labelled space $\mathcal{Q} = (W, \leq, \ell, S)$, where $W = \{(\mathbf{m}_s^k, k) \mid k < i+l \text{ and } s \leq m_k\}$ and $\leq, \ell, S$ are defined in the obvious way. It is not hard to check that $\mathcal{Q}$ is a quasimodel falsifying φ . Hence by Theorem 3, φ is falsifiable.

From this we may define Algorithm 1, which nondeterministically checks for falsifiability in PSPACE. According to Savitch's theorem [16], nondeterministic polynomial space is included in deterministic polynomial space; thus we obtain the following.

Theorem 5. *There exists a deterministic algorithm for falsifiability checking of an $\mathcal{L}$ -formula that is correct and works in space that is polynomial in the size of the input formula.*

With this we conclude that the validity problem is PSPACE complete.

Theorem 6. *The decision problem of testing falsifiability for GTL is PSPACE-complete.*

Algorithm 1. GTL falsifiability algorithm

```

input  $\varphi$ 
set  $\Sigma$  to be the set of subformulas of  $\varphi$ 
guess two moments,  $\mathbf{m}$  and  $\mathbf{m}_f$ , of heights  $m, n \leq |\Sigma| + 1$  such that  $\varphi \notin \mathbf{m}_m$ 
while  $\mathbf{m} \neq \mathbf{m}_f$  do
  guess a moment  $\mathbf{n}$  of height at most  $|\Sigma| + 1$ 
  if  $\mathbf{n}$  is a temporal successor of  $\mathbf{m}$  then  $\mathbf{m} \leftarrow \mathbf{n}$  else reject
end while
 $\Delta_\diamond \leftarrow \{(k, \diamond\psi) \mid \diamond\psi \in \mathbf{m}_k, 0 \leq k < n\}$ 
 $\Delta'_\diamond \leftarrow \{(k, \diamond\psi) \mid \psi \in \mathbf{m}_k, 0 \leq k < n\}$ 
 $\Delta_\square \leftarrow \{(k, \square\psi) \mid \square\psi \in \Sigma \setminus \mathbf{m}_k, 0 \leq k < n\}$ 
 $\Delta'_\square \leftarrow \{(k, \square\psi) \mid \psi \in \Sigma \setminus \mathbf{m}_k, 0 \leq k < n\}$ 
 $S^* \leftarrow \{(k, k) \mid 0 \leq k < n\}$ 
while  $\mathbf{m}_f \neq \mathbf{m}$  or  $\Delta_\diamond \not\subseteq \Delta'_\diamond$  or  $\Delta_\square \not\subseteq \Delta'_\square$  do
  guess a moment  $\mathbf{n}$  of height at most  $|\Sigma| + 1$ .
  if  $\mathbf{n}$  is a temporal successor of  $\mathbf{m}$  then  $\mathbf{m} \leftarrow \mathbf{n}$  else reject
   $S^* = \{(k, z) \mid (k, y) \in S^* \text{ and } (\mathbf{m}_y, \mathbf{n}_z) \in R\}$ 
   $\Delta'_\diamond \leftarrow \Delta'_\diamond \cup \{(k, \diamond\psi) \mid (k, x) \in S^* \text{ and } \psi \in \mathbf{n}_x\}$ 
   $\Delta'_\square \leftarrow \Delta'_\square \cup \{(k, \square\psi) \mid (k, x) \in S^* \text{ and } \psi \in \Sigma \setminus \mathbf{n}_x\}$ 
   $j \leftarrow j + 1$ 
end while
accept

```

8 Concluding Remarks

We have defined a natural version of linear temporal logic with a Gödel–Dummett base and shown that it may equivalently be characterized as a fuzzy logic or as a superintuitionistic logic using standard semantics in each case. Despite the lack of a finite model property for either of the two semantics, we have introduced a class of quasimodels for which GTL *does* satisfy a version of the finite model property, and moreover shown how these quasimodels can be used to adapt the classical proof of PSPACE-completeness for LTL.

This puts Gödel temporal logics in sharp contrast to other fuzzy logics, whose transitive modal versions are undecidable [19], or intuitionistic temporal logics, where systems are known to be decidable only with non-elementary upper bounds, if at all [5, 13]. This places Gödel–Dummett logic as *the* premier base for computational applications of sub-classical modal and temporal logics. In the sequel paper [3], we give a sound and complete calculus for this logic with a finite number of axioms and rules.

The techniques we have used are quite robust and should readily generalise to logics such as PDL or even the modal μ -calculus. This represents a milestone in the program pioneered by Caicedo et al. [9] of extending complexity results from classical modal and temporal logics to their Gödel counterparts.

Acknowledgments. This work has been partially supported by FWO-FWF grant G030620N/I4513N (J.P.A. and D.F.D.), FWO grant 3E017319 (J.P.A.), the projects

EL4HC and étoiles montantes CTASP at Région Pays de la Loire, France (M.D.), the COST action CA-17124 (M.D. and D.F.D.), and SNSF-FWO Lead Agency Grant 200021L_196176/G0E2121N (B.M. and D.F.D.).

References

1. Aguado, F., et al.: Linear-time temporal answer set programming. *Theory Pract. Log. Program.* 1–55 (2021). Advance Online Publication
2. Aguado, F., Cabalar, P., Diéguez, M., Pérez, G., Vidal, C.: Temporal equilibrium logic: a survey. *J. Appl. Non-Class. Log.* **23**(1–2), 2–24 (2013)
3. Aguilera, J.P., Diéguez, M., McLean, B., Fernández-Duque, D.: A Gödel calculus for linear temporal logic. In: 19th International Conference on Principles of Knowledge Representation and Reasoning (2022, to appear)
4. Baaz, M., Preining, N., Zach, R.: First-order Gödel logics. *Ann. Pure Appl. Log.* **147**, 23–47 (2007)
5. Balbiani, P., Boudou, J., Diéguez, M., Fernández-Duque, D.: Intuitionistic linear temporal logics. *ACM Trans. Comput. Log.* **21**(2), 14:1–14:32 (2020)
6. Balbiani, P., Diéguez, M.: Temporal here and there. In: Michael, L., Kakas, A. (eds.) *JELIA 2016. LNCS (LNAI)*, vol. 10021, pp. 81–96. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48758-8_6
7. Balbiani, P., Diéguez, M., Fernández-Duque, D.: Some constructive variants of S4 with the finite model property. In: 36th Annual ACM/IEEE Symposium on Logic in Computer Science, pp. 1–13. IEEE (2021)
8. Bílková, M., Frittella, S., Kozhemiachenko, D.: Constraint tableaux for two-dimensional fuzzy logics. In: Das, A., Negri, S. (eds.) *TABLEAUX 2021. LNCS (LNAI)*, vol. 12842, pp. 20–37. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86059-2_2
9. Caicedo, X., Metcalfe, G., Rodríguez, R.O., Rogger, J.: Decidability of order-based modal logics. *J. Comput. Syst. Sci.* **88**, 53–74 (2017)
10. Demri, S., Goranko, V., Lange, M.: *Temporal Logics in Computer Science: Finite-State Systems*. Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, Cambridge (2016)
11. Fernández-Duque, D.: Non-deterministic semantics for dynamic topological logic. *Ann. Pure Appl. Log.* **157**(2–3), 110–121 (2009)
12. Fernández-Duque, D.: A sound and complete axiomatization for dynamic topological logic. *J. Symb. Log.* **77**(3), 947–969 (2012)
13. Fernández-Duque, D.: The intuitionistic temporal logic of dynamical systems. *Log. Methods Comput. Sci.* **14**(3) (2018)
14. Rauszer, C.: *An Algebraic and Kripke-Style Approach to a Certain Extension of Intuitionistic Logic*. Instytut Matematyczny Polskiej Akademii Nauk, Warsaw (1980)
15. Reynolds, M., Zakharyashev, M.: On the Products of Linear Modal Logics. *J. Log. Comput.* **11**(6), 909–931 (2001)
16. Savitch, W.J.: Relationships between nondeterministic and deterministic tape complexities. *J. Comput. Syst. Sci.* **4**(2), 177–192 (1970)
17. Sistla, A.P., Clarke, E.M.: The complexity of propositional linear temporal logics. *J. ACM* **32**(3), 733–749 (1985)

18. Troelstra, A.S., van Dalen, D.: Constructivism in Mathematics: An Introduction Volume 1. Studies in Logic and the Foundations of Mathematics, vol. 121, North-Holland (1988) Constructivism in Mathematics: An Introduction Volume 1. Studies in Logic and the Foundations of Mathematics 121, North- Holland (1988)
19. Vidal, A.: On transitive modal many-valued logics. *Fuzzy Sets Syst.* **407**, 97–114 (2021)
20. Wolter, F.: On logics with coimplication. *J. Philos. Log.* **27**(4), 353–387 (1998). <https://doi.org/10.1023/A:1004218110879>



Fixed Point Logics and Definable Topological Properties

David Fernández-Duque^{1,2} and Quentin Gougeon^{3(✉)}

¹ Institute of Computer Science, Czech Academy of Sciences, Prague, Czech Republic
David.FernandezDuque@UGent.be

² Department of Mathematics WE16, Ghent University, Ghent, Belgium

³ CNRS-INPT-UT3, Toulouse University, Toulouse, France
quentin.gougeon@irit.fr

Abstract. Modal logic enjoys topological semantics that may be traced back to McKinsey and Tarski, and the classification of topological spaces via modal axioms is a lively area of research. In the past two decades, there has been interest in extending topological modal logic to the language of the μ -calculus, but previously no class of topological spaces was known to be μ -calculus definable that was not already modally definable. In this paper we show that the full μ -calculus is indeed more expressive than standard modal logic, in the sense that there are classes of topological spaces (and weakly transitive Kripke frames) which are μ -definable, but not modally definable. The classes we exhibit satisfy a modally definable property outside of their perfect core, and thus we dub them *imperfect spaces*. We show that the μ -calculus is sound and complete for these classes. Our examples are minimal in the sense that they use a single instance of a greatest fixed point.

Keywords: Mu-calculus · Expressivity · Topological semantics

1 Introduction

Topological semantics for modal logic originated with McKinsey and Tarski [17] in the 1940's, but saw a more recent revival due to the work of Esakia [9], Shehtman [20], and others. In what we call the *closure semantics*, the modal $\Diamond$ is interpreted as the topological closure, and $\Box$ as the interior. The logic of all topological spaces in this semantics is **S4**, and we refer to [4] for an overview of topological completeness of modal logics above **S4**. The more expressive [16] *derivational semantics* has gained traction in recent years, but was already considered by McKinsey and Tarski. It is obtained by interpreting the modal $\Diamond$ as the Cantor derivative.¹ Esakia [8, 9] showed that the derivative logic of all topological spaces is the modal logic **wK4** = **K** + $(\Diamond\Diamond p \rightarrow p \vee \Diamond p)$. This is also the modal logic of all *weakly transitive* frames, i.e., those for which the reflexive

¹ Recall that the derivative $\mathbf{d}(A)$ of a set A consists of all limit points of A .

closure of the accessibility relation is transitive. It is well-known that the modal logic of transitive frames is **K4** [6, 7], which moreover corresponds to a natural class of topological spaces denoted T_d . Many familiar topological spaces are T_d , such as Euclidean spaces.

Even more recently, topological semantics have been extended to the language of the μ -calculus [3, 10, 11, 13]. The relational μ -calculus is notoriously challenging from a theoretical perspective, with difficult completeness [21] and decidability [15] proofs (see also [1, 18, 19] for more recent work exhibiting various modifications to these results and their proofs). Since a transitive modality is already definable in the basic μ -calculus, Goldblatt and Hodkinson [14] obtained completeness and decidability as a corollary for transitive frames, and thus for T_d spaces. This does not work for weakly transitive frames, but surprisingly, Baltag et al. [3] showed that the combination of the μ -calculus with topological semantics is much more manageable than the original μ -calculus, with natural and transparent proofs of decidability and completeness involving only classical tools from modal logic (albeit intricately combined).

Thus the topological μ -calculus is decidable and complete, potentially placing it as a powerful yet technically manageable framework for reasoning about topologically-defined fixed points. The Achilles' heel of this proposal is that despite the sophisticated machinery, no class of topological spaces was formerly known to be μ -definable, but not modally definable. Our goal is to exhibit such classes of spaces. Here it is convenient to recall the notion of reducibility of formal languages, following Kudinov and Shehtman [16]. If $\mathcal{L}$ and $\mathcal{L}'$ are sub-languages of the μ -calculus, then $\mathcal{L}$ *reduces to* $\mathcal{L}'$ if every class of spaces definable in $\mathcal{L}$ is also definable in $\mathcal{L}'$ (see Sect. 2). If $\mathcal{L}$ reduces to $\mathcal{L}'$, we may also say that $\mathcal{L}'$ is *at least as expressive as* $\mathcal{L}$, and if moreover $\mathcal{L}'$ does not reduce to $\mathcal{L}$, we say that $\mathcal{L}'$ is *more expressive than* $\mathcal{L}$.²

More precisely, we manage to exhibit infinitely many topologically complete logics in the language of the μ -calculus whose classes of spaces are not modally definable. These axioms separate spaces into two parts, a perfect part (i.e., without isolated points), and a complement satisfying some property definable by a modal formula φ ; we call these spaces φ -*imperfect spaces*. The perfect part is defined via a greatest fixed point operator.

The paper is structured as follows: in Sect. 2, we present the relevant material regarding derivative spaces, the μ -calculus and axiomatic expressivity. In Sect. 3, we use greatest fixed points to construct classes of spaces that are not modally definable. Completeness results for some of these classes are then laid out in Sect. 4. We end with some concluding remarks in Sect. 5.

² Note that a stronger notion of expressivity is also considered in the literature: namely, $\mathcal{L}'$ is at least as expressive as $\mathcal{L}$ if for every $\varphi \in \mathcal{L}$ there is a logically equivalent $\varphi' \in \mathcal{L}'$. To avoid confusion we may call the latter *local expressivity*, and the notion we are concerned with *axiomatic expressivity*. With this terminology in mind, while it was known that μ -calculus is locally more expressive than the basic modal language over topological spaces (see e.g. [10]), here we will show that it is also axiomatically more expressive.

2 Background

In this section we review the syntax and semantics of the topological μ -calculus. Following [3, 12], we present our semantics in the general setting of *derivative spaces*, and work in a language with ν (rather than μ) as primitive.

Definition 1. We fix a countable set Prop of *atomic propositions* (also called *variables*). The language $\mathcal{L}_\mu$ of the modal μ -calculus is defined by the following grammar:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \varphi \mid \Diamond\varphi \mid \nu p.\varphi$$

where $p \in \text{Prop}$ and in the construct $\nu p.\varphi$, the formula φ is *positive in p* , that is, every occurrence of p lies under the scope of an even number of negations. The abbreviations $\varphi \vee \psi$, $\Box\varphi$, $\perp$ and $\top$ are defined as usual. We denote by $\varphi[\psi_1, \dots, \psi_n/p_1, \dots, p_n]$ the formula φ where each formula ψ_i is substituted for every free occurrence of the variable p_i . We then introduce the abbreviation $\mu p.\varphi := \neg\nu p.\neg\varphi[\neg p/p]$. Finally, the *basic modal language* $\mathcal{L}_\Diamond$ is the fragment of $\mathcal{L}_\mu$ without occurrences of ν .

Definition 2. A *derivative space* is a pair $\mathcal{X} = (X, \mathbf{d})$, where X is a set of *points* and $\mathbf{d}: \mathcal{P}(X) \rightarrow \mathcal{P}(X)$ is an operator on subsets of X , satisfying for all $A, B \subseteq X$:

- $\mathbf{d}(\emptyset) = \emptyset$,
- $\mathbf{d}(A \cup B) = \mathbf{d}(A) \cup \mathbf{d}(B)$,
- $\mathbf{d}(\mathbf{d}(A)) \subseteq A \cup \mathbf{d}(A)$.

A *derivative model* based on $\mathcal{X}$ is a tuple of the form $\mathfrak{M} = (X, \mathbf{d}, V)$ with $V: \text{Prop} \rightarrow \mathcal{P}(X)$ a *valuation*. Given $x \in X$ we then call $(\mathfrak{M}, x)$ a *pointed derivative model*. If $p \in \text{Prop}$ and $A \subseteq X$, we define the valuation $V[p := A]$ by

$$V[p := A](q) := \begin{cases} A & \text{if } p = q \\ V(q) & \text{otherwise} \end{cases}.$$

We then write $\mathfrak{M}[p := A] := (X, \mathbf{d}, V[p := A])$.

Definition 3. Given a derivative model $\mathfrak{M} = (X, \mathbf{d}, V)$, we define by induction on a formula $\varphi \in \mathcal{L}_\mu$ the *extension* $\llbracket \varphi \rrbracket_{\mathfrak{M}}$ of φ in $\mathfrak{M}$ as follows:

$$\begin{aligned} \llbracket p \rrbracket_{\mathfrak{M}} &:= V(p) & \llbracket \neg\varphi \rrbracket_{\mathfrak{M}} &:= X \setminus \llbracket \varphi \rrbracket_{\mathfrak{M}} \\ \llbracket \varphi \wedge \psi \rrbracket_{\mathfrak{M}} &:= \llbracket \varphi \rrbracket_{\mathfrak{M}} \cap \llbracket \psi \rrbracket_{\mathfrak{M}} & \llbracket \Diamond\varphi \rrbracket_{\mathfrak{M}} &:= \mathbf{d}(\llbracket \varphi \rrbracket_{\mathfrak{M}}) \\ \llbracket \nu p.\varphi \rrbracket_{\mathfrak{M}} &:= \bigcup \{A \subseteq W \mid A \subseteq \llbracket \varphi \rrbracket_{\mathfrak{M}[p := A]}\} \end{aligned}$$

We then write $\mathfrak{M}, x \models \varphi$ whenever $x \in \llbracket \varphi \rrbracket_{\mathfrak{M}}$ and we say that φ is *true* at the point x . If $\mathfrak{M}$ is based on $\mathcal{X}$ and $\mathfrak{M}, x \models \varphi$, we say that φ is *satisfiable* on $\mathfrak{M}$, or on $\mathcal{X}$, or on $\mathcal{X}, x$ (depending on what is deemed relevant).

If $\llbracket \varphi \rrbracket_{\mathfrak{M}} = X$, we write $\mathfrak{M} \models \varphi$. If $\mathfrak{M} \models \varphi$ for all models $\mathfrak{M}$ based on $\mathcal{X}$ we write $\mathcal{X} \models \varphi$ and we say that φ is *valid* on $\mathcal{X}$. We also have a notion of

pointwise validity, that is, if $\mathfrak{M}, x \models \varphi$ for every model $\mathfrak{M}$ based on $\mathcal{X}$, then we write $\mathcal{X}, x \models \varphi$. If $\mathcal{X} \models \varphi$ for all derivative spaces $\mathcal{X}$, we write $\models \varphi$. Given a class $\mathcal{C}$ of derivative spaces, we write $\mathcal{C} \models \varphi$ whenever $\mathcal{X} \models \varphi$ for all $\mathcal{X} \in \mathcal{C}$. If Γ is a set of formulas we write $\mathfrak{M}, x \models \Gamma$ whenever $\mathfrak{M}, x \models \varphi$ for all $\varphi \in \Gamma$, and all of the other notations are adapted accordingly.

In modal logic it is customary to study morphisms that preserve validity. In the context of derivative spaces, these are known as *d-morphisms* (see e.g. [16]).

Definition 4. Let $\mathcal{X} = (X, d)$ and $\mathcal{X}' = (X', d')$ be two derivative spaces. A map $f: X \rightarrow X'$ is called a *d-morphism* from $\mathcal{X}$ to $\mathcal{X}'$ if it satisfies $f^{-1}[d'(A')] = d(f^{-1}[A'])$ for all $A' \subseteq X'$.

Proposition 5. Let $\mathcal{X} = (X, d)$ and $\mathcal{X}' = (X', d')$ be two derivative spaces and $f: X \rightarrow X'$ a *d-morphism*. If $\varphi \in \mathcal{L}_\mu$ and $\mathcal{X} \models \varphi$, then $\mathcal{X}' \models \varphi$.

Presenting our semantics in terms of derivative spaces is useful, as both weakly transitive Kripke frames and topological spaces (either with the closure or the d operator) can be viewed as special cases of derivative spaces. While our ‘intended’ semantics is topological, Kripke semantics will be useful in establishing many of our main results.

Definition 6. A Kripke frame is a pair $\mathfrak{F} = (W, R)$, with W a set of *possible worlds* and $R \subseteq W^2$. We denote by $R^+ := R \cup \{(w, w) \mid w \in W\}$ the *reflexive closure* of R . The frame $\mathfrak{F}$ is said to be *rooted* in r if for all $w \in W$ we have rR^+w . We say that $\mathfrak{F}$ is *weakly transitive* if wRu and uRv implies wR^+v . In this case $\mathfrak{F}$ is also called a **wK4** frame, and it induces a derivative space (W, d) with d defined by $d(A) := \{w \mid wRu \text{ and } u \in A\}$.

Slightly abusing terminology, we will identify $\mathfrak{F}$ and (W, d) (since one can be constructed from the other). Then (pointed) derivative models based on **wK4** frames will be called (pointed) *Kripke models*, while *d-morphisms* between **wK4** frames will be called *bounded morphisms*.

Now we turn our attention to the ‘official’ semantics of the topological μ -calculus.

Definition 7. Let X be a set of *points*. A *topology* on X is a set $\tau \subseteq \mathcal{P}(X)$ containing $\emptyset$ and X , closed under arbitrary unions, and closed under finite intersections. The pair (X, τ) is then called a *topological space*. The elements of τ are called the *open* sets of X . The complement of an open set is called a *closed* set. If $x \in U \in \tau$ then U is called an *open neighbourhood* of x . Slightly abusing notation, we will often keep τ implicit and let X refer to the space (X, τ) .

Definition 8. Let X be a topological space, $A \subseteq X$ and $x \in X$. The point x is said to be a *limit point* of A if for all open neighbourhoods U of x , we have $U \cap A \setminus \{x\} \neq \emptyset$. We denote by $d(A)$ the set of all limit points of A and call it the *derived set* of A . The dual of d is defined by $\widehat{d}(A) := X \setminus d(X \setminus A)$.

Given a topological space X , it is easily observed that the pair (X, d) is a derivative space. Conversely, the topology τ can be recovered from d since for all $A \subseteq X$, the set A is closed if and only if $d(A) \subseteq A$. For this reason we choose,

again, to identify (X, τ) and $(X, \mathbf{d})$. Then (pointed) derivative models based on topological spaces will be called (pointed) *topological models*. Observe that the familiar *closure* and *interior* operators can be defined by $\text{Cl}(A) := A \cup \mathbf{d}(A)$ and $\text{Int}(A) := A \cap \widehat{\mathbf{d}}(A)$. Writing $\Box^+ \varphi := \varphi \wedge \Box \varphi$ and $\Diamond^+ \varphi := \varphi \vee \Diamond \varphi$, we then have $\llbracket \Box^+ \varphi \rrbracket_{\mathfrak{M}} = \text{Int}(\llbracket \varphi \rrbracket_{\mathfrak{M}})$ and $\llbracket \Diamond^+ \varphi \rrbracket_{\mathfrak{M}} = \text{Cl}(\llbracket \varphi \rrbracket_{\mathfrak{M}})$ for all topological models $\mathfrak{M}$. We recall some important classes of topological spaces that will be useful throughout the text.

Definition 9. Let X be a topological space. A point $x \in X$ is said to be *isolated* if $\{x\}$ is open. Given $x \in A \subseteq X$ we say that x is *isolated in A* if there exists U open such that $\{x\} = U \cap A$. The space X is called *dense-in-itself* if it contains no isolated point. The space X is called *scattered* if any subspace of X contains an isolated point. We say that X is T_d if every $x \in X$ is isolated in $\text{Cl}(\{x\})$. We say that X is *extremally disconnected* if $\text{Cl}(U)$ is open for every open set U , and *Aleksandroff* if arbitrary intersections of open sets are open.

Aleksandroff spaces are closely connected to Kripke frames, via the following construction.

Definition 10. Let $\mathfrak{F} := (W, R)$ be a **wK4** frame. A set $U \subseteq W$ is called an *upset* if $w \in U$ and wRu implies $u \in U$. The collection τ_R of all upsets over W is then a topology, and (W, τ_R) is called the topological space *induced* by $\mathfrak{F}$. If $\mathfrak{M} = (W, R, V)$ is a Kripke model based on $\mathfrak{F}$, then $((W, \tau_R), V)$ is the topological model *induced* by $\mathfrak{M}$.

It is not hard to check that a space of the form (W, τ_R) is always Aleksandroff (and, indeed, every Aleksandroff space is of this form [2]). In fact we will simply not distinguish a weakly transitive Kripke frame from the topological space induced by it. This is partly motivated by the following proposition.

Proposition 11. *Let $\mathfrak{M} = (W, R, V)$ be an irreflexive and weakly transitive Kripke model, and $\mathfrak{M}' := ((W, \tau_R), V)$ the space induced by it. For all $w \in W$ and $\varphi \in \mathcal{L}_\mu$ we have*

$$\mathfrak{M}, w \models \varphi \iff \mathfrak{M}', w \models \varphi.$$

The modal logic of all topological spaces is known as **wK4**, and consists of the following induction rules and axioms:

Name	Axiom/inference rule
	All propositional tautologies
Uniform substitution	From φ infer $\varphi[\psi_1, \dots, \psi_n/p_1, \dots, p_n]$
K (Distribution)	$\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$
Modus Ponens	From φ and $\varphi \rightarrow \psi$ infer ψ
Necessitation	From φ infer $\Box \varphi$
Weak transitivity	$\Diamond \Diamond p \rightarrow p \vee \Diamond p$

The axiomatic system **K4** is the extension of **wK4** with the axiom $4 := \Diamond p \rightarrow \Diamond \Diamond p$. The axiomatic system $\mu\mathbf{wK4}$ is the extension of **wK4** with the *fixed point axiom* $\nu p. \varphi \rightarrow \varphi[\nu p. \varphi / p]$ and the *induction rule*

$$\text{from } \varphi \rightarrow \psi[\varphi/p] \text{ infer } \varphi \rightarrow \nu p. \psi.$$

Definition 12. Let **L** be a logic in a sub-language of $\mathcal{L}_\mu$. If φ is a formula, the statement $\mathbf{L} \vdash \varphi$ says that φ is derivable in **L**. We say that **L** is *sound and complete* with respect to a class $\mathcal{C}$ of derivative spaces if for all formulas φ we have $\mathbf{L} \vdash \varphi$ iff $\mathcal{C} \models \varphi$. We call **L** *Kripke complete* if it is sound and complete with respect to some class of Kripke frames, and *topologically complete* if it is sound and complete with respect to some class of topological spaces.

Theorem 13 ([3]). *The logic $\mu\mathbf{wK4}$ is sound and complete with respect to the class of all **wK4** frames, with respect to the class of all topological spaces, and with respect to the class of all derivative spaces.*

In order to compare the expressivity of different languages, we need to introduce the notion of definable classes.

Definition 14. Given a formula φ , we let $\mathcal{C}(\varphi)$ be the class of derivative spaces $\mathcal{X}$ such that $\mathcal{X} \models \varphi$. Let $\mathcal{C}_0$ be a class of derivative spaces and $\mathcal{L} \subseteq \mathcal{L}_\mu$. We say that $\mathcal{C}$ is *$\mathcal{L}$ -definable within $\mathcal{C}_0$* if there exists $\varphi \in \mathcal{L}$ such that $\mathcal{C}(\varphi) \cap \mathcal{C}_0 = \mathcal{C} \cap \mathcal{C}_0$.

If $\mathcal{L}, \mathcal{L}' \subseteq \mathcal{L}_\mu$, we say that $\mathcal{L}'$ is *at least as expressive as $\mathcal{L}$ over $\mathcal{C}_0$* if every class definable in $\mathcal{L}$ within $\mathcal{C}_0$ is also definable in $\mathcal{L}'$ within $\mathcal{C}_0$. If $\mathcal{L}'$ is at least as expressive as $\mathcal{L}$ but $\mathcal{L}$ is not at least as expressive as $\mathcal{L}'$, we say that $\mathcal{L}'$ is *more expressive than $\mathcal{L}$ over $\mathcal{C}_0$* .

In particular, a $\mathcal{L}_\Diamond$ -definable class will be called *modally definable*, and a $\mathcal{L}_\mu$ -definable class will be called *μ -definable*. As discussed in Footnote, this notion of expressivity is also known as *reducibility* or *axiomatic expressivity* (see Footnote 2). The choice to compare expressivity relatively to a class of derivative spaces is convenient as it allows to derive all kinds of auxiliary results. We will consider the following classes of interest:

$$\begin{aligned} \mathcal{C}_{\text{all}} &:= \{\mathcal{X} \mid \mathcal{X} \text{ is a derivative space}\} \\ \mathcal{C}_{\text{fin}} &:= \{(X, \mathbf{d}) \in \mathcal{C}_{\text{all}} \mid X \text{ is finite}\} \\ \mathcal{C}_{\text{Kripke}} &:= \{\mathfrak{F} \mid \mathfrak{F} \text{ is a } \mathbf{wK4} \text{ frame}\} \\ \mathcal{C}_{\text{irrefl}} &:= \{\mathfrak{F} \in \mathcal{C}_{\text{Kripke}} \mid \mathfrak{F} \text{ is irreflexive}\} \\ \mathcal{C}_{\text{topo}} &:= \{X \mid X \text{ is a topological space}\} \\ \mathcal{C}_{\mathbf{K4}} &:= \{\mathcal{X} \in \mathcal{C}_{\text{all}} \mid \mathcal{X} \models \mathbf{K4}\} \end{aligned}$$

It is well established that $\mathcal{C}_{\text{Kripke}} \cap \mathcal{C}_{\mathbf{K4}}$ is the class of transitive Kripke frames [6], while $\mathcal{C}_{\text{topo}} \cap \mathcal{C}_{\mathbf{K4}}$ is the class of T_d spaces [4].

3 Classes Defined by Greatest Fixed Points

The goal of this section is to exhibit μ -definable classes that are not modally definable. It turns out that a whole family of formulas of the form $\theta \vee \nu p. \Diamond p$ will

yield the desired result. We easily see that given a pointed Kripke model $(\mathfrak{M}, x)$, we have $\mathfrak{M}, x \models \nu p.\Diamond p$ if and only if there exists an infinite path starting from x . Topologically, $\nu p.\Diamond p$ holds in the *perfect core* of X , the largest dense-in-itself subset of X . While the existence of an infinite path is not in general modally definable, it is not hard to check that $\mathcal{C}(\nu p.\Diamond p) = \mathcal{C}(\Diamond \top)$, as this is just the class of dense-in-themselves spaces. However, the story becomes more complicated if we only require certain points in the space to satisfy $\nu p.\Diamond p$. In this case, the following can be applied to exhibit many modally undefinable classes of spaces.

Theorem 15. *Let $\theta \in \mathcal{L}_\mu$ and suppose that for all $n \in \mathbb{N}$ there exists a **wK4** frame $\mathfrak{F}_n = (W_n, R_n)$ and $r_n \in W_n$ such that:*

1. $\mathfrak{F}_n$ is rooted in r_n and $\mathfrak{F}_n, r_n \not\models \theta \vee \nu p.\Diamond p$;
2. $\mathfrak{F}_n$ contains a path of length n ;
3. for all $w \in W_n \setminus \{r_n\}$ we have $\mathfrak{F}_n, w \models \theta$.

Then $\mathcal{C}(\theta \vee \nu p.\Diamond p)$ is not modally definable within $\mathcal{C}_{\text{irrefl}} \cap \mathcal{C}_{\mathbf{K4}}$. If in addition every $\mathfrak{F}_n$ is finite, then $\mathcal{C}(\theta \vee \nu p.\Diamond p)$ is not modally definable within $\mathcal{C}_{\text{irrefl}} \cap \mathcal{C}_{\text{fin}}$ and $\mathcal{C}_{\text{Kripke}} \cap \mathcal{C}_{\text{fin}} \cap \mathcal{C}_{\mathbf{K4}}$.

Remark 16. We recall that both Kripke frames and topological spaces are identified with their respective derivative spaces, so $\mathcal{C}_{\text{irrefl}} \cap \mathcal{C}_{\mathbf{K4}}$ can equivalently be regarded as the class of all T_d Aleksandroff spaces, and $\mathcal{C}_{\text{irrefl}} \cap \mathcal{C}_{\text{fin}}$ as the class of finite topological spaces. Thus Theorem 15 applies to classes of topological spaces, as well as Kripke frames.

Remark 17. It is easily observed that if $\mathcal{C}$ is not modally definable within $\mathcal{C}_0$ and $\mathcal{C}_0 \subseteq \mathcal{C}_1$, then $\mathcal{C}$ is not modally definable within $\mathcal{C}_1$ as well. This allows us to draw interesting consequences from Theorem 15, as $\mathcal{C}_{\text{irrefl}} \cap \mathcal{C}_{\mathbf{K4}}$ is a subclass of $\mathcal{C}_{\text{all}}$, $\mathcal{C}_{\text{Kripke}}$, $\mathcal{C}_{\text{topo}}$, $\mathcal{C}_{\text{topo}} \cap \mathcal{C}_{\mathbf{K4}}$ and many other relevant classes.

From now on, we fix a formula θ and a family of frames $(\mathfrak{F}_n)_{n \in \mathbb{N}}$ satisfying the assumptions of Theorem 15. For all $n \in \mathbb{N}$, we assume that $W_n \cap \omega = \emptyset$. We start with an elementary observation.

Claim 18. *For all $n \in \mathbb{N}$, the frame $\mathfrak{F}_n$ is irreflexive and transitive.*

Proof. First assume that $\mathfrak{F}_n$ is not irreflexive, so that there is w with $wR_n w$. Then, $(r_n, w, w, \dots)$ is an infinite path beginning on r_n , contradicting $\mathfrak{F}_n, r_n \not\models \nu p.\Diamond p$. If instead $\mathfrak{F}_n$ is not transitive, then since $\mathfrak{F}_n$ is weakly transitive, this can only occur if there exist $w, u \in W_n$ such that $wR_n u$, $uR_n w$ and not $wR_n w$. Then, $(r_n, w, u, w, u, \dots)$ is an infinite path beginning on r_n (or else $(w, u, w, u, \dots)$ in case $w = r_n$). $\square$

Given a world $w \in W_n$, we define the **wK4** frames $\mathfrak{F}_{n,w}^{\text{point}} = (W^0, R^0)$, $\mathfrak{F}_{n,w}^{\text{cycle}} = (W^1, R^1)$ and $\mathfrak{F}_{n,w}^{\text{spine}} = (W^2, R^2)$ by:

$$W^0 := W_n \cup \{0\}$$

$$R^0 := R_n \cup \{(r_n, 0), (0, 0)\} \cup \{(0, u) \mid wR_n^+ u\}$$

$$W^1 := W_n \cup \{0, 1\}$$

$$R^1 := R_n \cup \{(r_n, 0), (r_n, 1), (0, 1), (1, 0)\} \cup \{(k, u) \mid k \in \{0, 1\} \text{ and } wR_n^+ u\}$$

$$W^2 := W_n \cup \omega$$

$$R^2 := R_n \cup \{(r_n, k) \mid k \in \omega\} \cup \{(m, k) \mid m < k < \omega\} \cup \{(k, u) \mid k \in \omega, wR_n^+ u\}$$

In words, $\mathfrak{F}_{n,w}^{\text{point}}$ is the frame $\mathfrak{F}_n$ endowed with a reflexive point reachable from the root, and which sees all the successors of w (as well as w itself). The frames $\mathfrak{F}_{n,w}^{\text{cycle}}$ and $\mathfrak{F}_{n,w}^{\text{spine}}$ are constructed similarly, but with respectively a two-element loop and an infinite branch, instead of a reflexive point. The three frames are depicted in Fig. 1.

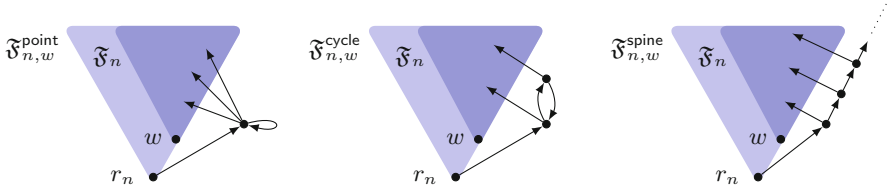


Fig. 1. The frames $\mathfrak{F}_{n,w}^{\text{point}}$, $\mathfrak{F}_{n,w}^{\text{cycle}}$ and $\mathfrak{F}_{n,w}^{\text{spine}}$

If some modal formula ψ defines the same class of spaces as $\theta \vee \nu p. \Diamond p$, then by construction ψ should be refuted at $\mathfrak{F}_n, r_n$ for all n , but not at $\mathfrak{F}_{n,w}^{\text{spine}}, r_n$ or $\mathfrak{F}_{n,w}^{\text{cycle}}, r_n$ or $\mathfrak{F}_{n,w}^{\text{point}}, r_n$ since in all three of them there is an infinite path starting from the root. Yet we will prove that if n is big enough and $\neg\psi$ is satisfiable on $\mathfrak{F}_n, r_n$, then it is also satisfiable on $\mathfrak{F}_{n,w}^{\text{point}}, r_n$ for some w , leading to a contradiction.³ The proof is rather technical, but we can sketch the main lines of our strategy. First, it is clear that transferring the satisfiability of a diamond formula (i.e., of the form $\Diamond\varphi$) or a Boolean formula from $\mathfrak{F}_n, r_n$ to $\mathfrak{F}_{n,w}^{\text{point}}, r_n$ is immediate, so the challenge really comes from box formulas (of the form $\Box\varphi$). The central argument is that since n may be arbitrarily large, we can select some $\mathfrak{F}_n$ with an arbitrarily long path. By means of a pigeonhole argument, we will then manage to show that on some point w of this path, if $\Box\varphi$ is satisfied, then so is $\Box^+\varphi$ (when $\Box\varphi$ is any subformula of $\neg\psi$). Then, transferring the truth of $\Box\varphi$ to the reflexive point of $\mathfrak{F}_{n,w}^{\text{point}}$ will be straightforward.

First, we recall that the *negative normal form* (or NNF for short) for modal logic is the syntax generated by the following grammar:

$$\varphi ::= p \mid \neg p \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \Box\varphi \mid \Diamond\varphi.$$

It is well known that for all modal formulas, there exists an equivalent formula in NNF. We also introduce a notion of *type* of a possible world.

³ Later we will see that the same result with $\mathfrak{F}_{n,w}^{\text{spine}}, r_n$ and $\mathfrak{F}_{n,w}^{\text{cycle}}, r_n$ follows for free.

Definition 19. Let φ be a modal formula. We write $\psi \leq \varphi$ whenever ψ is a subformula of φ . We also call the *box size* $|\varphi|_{\Box}$ of φ the number of subformulas of φ of the form $\Box\psi$. If $\mathfrak{M}$ is a derivative model and w a world in $\mathfrak{M}$, we define the *box type* of w relative to φ as the set $t_{\mathfrak{M}}^{\varphi}(w) := \{\Box\psi \mid \Box\psi \leq \varphi \text{ and } \mathfrak{M}, w \models \Box\psi\}$.

As explained above, the following result allows to transfer the satisfiability of box formulas as soon as the parameter n is large enough.

Claim 20. *Let φ be a modal formula in NNF and $n > 2^{|\varphi|_{\Box}}$. Suppose that there exists a valuation V over $\mathfrak{F}_n$ such that $\mathfrak{F}_n, V, r_n \models \Box\varphi$. Then there exists a world $w \in W_n$ and a valuation V' over $\mathfrak{F}_{n,w}^{\text{point}}$ such that $\mathfrak{F}_{n,w}^{\text{point}}, V', r_n \models \Box\varphi$, and V and V' coincide over $\mathfrak{F}_n$.*

Proof. First, we know that $\mathfrak{F}_n$ contains a path $(w_i)_{i \in [1,n]}$ of length n . By construction there are $2^{|\varphi|_{\Box}}$ different box types relative to φ . Thus, by the pigeonhole principle, there exists $i, j \in \mathbb{N}$ such that $1 \leq i < j \leq n$ and $t_{\mathfrak{M}}^{\varphi}(w_i) = t_{\mathfrak{M}}^{\varphi}(w_j)$. We then define a valuation V' over $\mathfrak{F}_{n,w_j}^{\text{point}}$ by setting, for all $p \in \text{Prop}$:

$$V'(p) := \begin{cases} V(p) \cup \{0\} & \text{if } w_j \in V(p) \\ V(p) & \text{otherwise} \end{cases}.$$

So V and V' coincide over $\mathfrak{F}_n$, and V' is defined over 0 so that this point satisfies the same atomic propositions as w_j . We then prove by induction on $\psi \leq \varphi$ that $\mathfrak{F}_n, V, w_j \models \psi$ implies $\mathfrak{F}_{n,w_j}^{\text{point}}, V', 0 \models \psi$:

- If ψ is of the form $\psi = p$ or $\psi = \neg p$ with $p \in \text{Prop}$ this is just true by construction.
- If ψ is of the form $\psi = \psi_1 \wedge \psi_2$, then $\mathfrak{F}_n, V, w_j \models \psi_1 \wedge \psi_2$ implies $\mathfrak{F}_n, V, w_j \models \psi_1$ and $\mathfrak{F}_n, V, w_j \models \psi_2$ and it suffices to apply the induction hypothesis. If ψ is of the form $\psi = \psi_1 \vee \psi_2$, then $\mathfrak{F}_n, V, w_j \models \psi_1 \vee \psi_2$ implies $\mathfrak{F}_n, V, w_j \models \psi_1$ or $\mathfrak{F}_n, V, w_j \models \psi_2$ and the result follows in the same way.
- Suppose that ψ is of the form $\psi = \Diamond\psi_0$ and $\mathfrak{F}_n, V, w_j \models \psi$. Then since V and V' coincide over $\mathfrak{F}_n$, we have $\mathfrak{F}_{n,w_j}^{\text{point}}, V', w_j \models \psi$ as well. By transitivity it follows that $\mathfrak{F}_{n,w_j}^{\text{point}}, V', 0 \models \psi$.
- Suppose that ψ is of the form $\psi = \Box\psi_0$ and that $\mathfrak{F}_n, V, w_j \models \psi$. Then since $t_{\mathfrak{M}}^{\varphi}(w_i) = t_{\mathfrak{M}}^{\varphi}(w_j)$, we have $\mathfrak{F}_n, V, w_i \models \psi$ as well. Since $w_i R_n w_j$ it follows $\mathfrak{F}_n, V, w_j \models \psi_0$, and then $\mathfrak{F}_{n,w_j}^{\text{point}}, V', 0 \models \psi_0$ by the induction hypothesis. Since V and V' coincide over $\mathfrak{F}_n$ we also have $\mathfrak{F}_{n,w_j}^{\text{point}}, V', w_j \models \Box^+ \psi_0$. All in all we obtain $\mathfrak{F}_{n,w_j}^{\text{point}}, V', 0 \models \Box\psi_0$ as desired.

Now observe that since $w_i R_n w_j$ we must have $w_j \neq r_n$, otherwise we would obtain $r_n R_n r_n$ by transitivity. Thus $r_n R_n w_j$ and from $\mathfrak{F}_n, V, r_n \models \Box\varphi$ we obtain $\mathfrak{F}_n, V, w_j \models \varphi$, and then $\mathfrak{F}_{n,w_j}^{\text{point}}, V', 0 \models \varphi$. Since V and V' coincide over $\mathfrak{F}_n$, we conclude that $\mathfrak{F}_{n,w_j}^{\text{point}}, V', r_n \models \Box\varphi$. $\square$

We can then extend the result to any modal formula.

Claim 21. *Let φ be a modal formula. There exists $n \in \mathbb{N}$ such that if φ is satisfiable on $\mathfrak{F}_n, r_n$, then there exists a world $w \in W_n$ such that φ is satisfiable on $\mathfrak{F}_{n,w}^{\text{spine}}$ and $\mathfrak{F}_{n,w}^{\text{cycle}}$ and $\mathfrak{F}_{n,w}^{\text{point}}$.*

Proof. Applying the theorem of disjunctive normal form for propositional logic, and using the fact that $\Box$ and $\wedge$ commute, we can assume that φ is of the form $\varphi = \bigvee_{i=1}^m \sigma_i$ with $\sigma_i = \rho_i \wedge \Box\psi_i \wedge \bigwedge_{j=1}^{m_i} \Diamond\theta_{i,j}$ for all $i \in [1, m]$, where ρ_i is a propositional formula. Note that since $\Box\top$ is a tautology, we can always assume the presence of $\Box\psi_i$. We also suppose that ψ_i is presented in NNF. We then define

$$n := 1 + \max \{2^{|\psi_i|_{\Box}} \mid 1 \leq i \leq m\}$$

and assume that there exists a valuation V such that $\mathfrak{F}_n, V, r_n \models \varphi$. Then there exists $i \in [1, m]$ such that $\mathfrak{F}_n, V, r_n \models \sigma_i$. It follows that $\mathfrak{F}_n, V, r_n \models \Box\psi_i$ with $n > 2^{|\psi_i|_{\Box}}$, so by Claim 20 there exists $w \in W_n$ and a valuation V' over $\mathfrak{F}_{n,w}^{\text{point}}$ such that $\mathfrak{F}_{n,w}^{\text{point}}, V', r_n \models \Box\psi_i$, and V and V' coincide over $\mathfrak{F}_n$. It is then clear that $\mathfrak{F}_{n,w}^{\text{point}}, V', r_n \models \sigma_i$, and thus $\mathfrak{F}_{n,w}^{\text{point}}, V', r_n \models \varphi$.

This proves that φ is satisfiable on $\mathfrak{F}_{n,w}^{\text{point}}$. Now consider the function f which maps every $n \in \omega$ (resp. $n \in \{0, 1\}$) to 0, and every $w \in W_n$ to w itself. Then f defines a bounded morphism from $\mathfrak{F}_{n,\mathfrak{F}}^{\text{spine}}$ (resp. $\mathfrak{F}_{n,w}^{\text{cycle}}$) to $\mathfrak{F}_{n,w}^{\text{point}}$, and we conclude that φ is satisfiable on $\mathfrak{F}_{n,w}^{\text{spine}}$ and $\mathfrak{F}_{n,w}^{\text{cycle}}$. $\square$

We are now ready to prove Theorem 15:

Proof. Suppose toward a contradiction that there is a formula $\psi \in \mathcal{L}_{\Diamond}$ defining the same class as $\theta \vee \nu p. \Diamond p$ within $\mathcal{C}_{\text{irrefl}} \cap \mathcal{C}_{\mathbf{K4}}$. Let n be the integer obtained by applying Claim 21 to $\neg\psi$. By Claim 18, the frame $\mathfrak{F}_n$ is irreflexive and transitive, and we also have $\mathfrak{F}_n \not\models \theta \vee \nu p. \Diamond p$ by assumption, so $\mathfrak{F}_n \not\models \psi$ as well.

Thus $\neg\psi$ is satisfiable on $\mathfrak{F}_n, v$ for some $v \in W_n$. If $v \neq r_n$, we denote by $\mathfrak{F}$ the subframe of $\mathfrak{F}_n$ generated by v . Then $\mathfrak{F}$ does not contain r_n , for otherwise we would have $vR_n r_n R_n v$ and thus $vR_n v$, a contradiction. The assumption on $\mathfrak{F}_n$ yields $\mathfrak{F} \models \theta$, so $\mathfrak{F} \models \theta \vee \nu p. \Diamond p$ and thus $\mathfrak{F} \models \psi$. Therefore $\mathfrak{F}_n, v \models \psi$, a contradiction. We conclude that $r_n = v$. Then by Claim 21 there exists $w \in W_n$ such that $\neg\psi$ is satisfiable on $\mathfrak{F}_{n,w}^{\text{spine}}$. Yet $\mathfrak{F}_{n,w}^{\text{spine}} \in \mathcal{C}_{\text{irrefl}} \cap \mathcal{C}_{\mathbf{K4}}$ and $\mathfrak{F}_{n,w}^{\text{spine}} \models \theta \vee \nu p. \Diamond p$, so $\mathfrak{F}_{n,w}^{\text{spine}} \models \psi$, a contradiction.

Now suppose that every $\mathfrak{F}_n$ is finite. By the same reasoning, we can show that $\mathcal{C}(\theta \vee \nu p. \Diamond p)$ is not modally definable within $\mathcal{C}_{\text{irrefl}} \cap \mathcal{C}_{\text{fin}}$ and $\mathcal{C}_{\text{Kripke}} \cap \mathcal{C}_{\text{fin}} \cap \mathcal{C}_{\mathbf{K4}}$. To that end it suffices to replace $\mathfrak{F}_{n,w}^{\text{spine}}$ by respectively $\mathfrak{F}_{n,w}^{\text{cycle}}$, which is irreflexive and finite, and $\mathfrak{F}_{n,w}^{\text{point}}$, which is transitive and finite. $\square$

Theorem 15 remains a very general statement, and it is worth instantiating it with examples. The following result shows the existence of infinitely many non-modally definable classes of spaces.

Proposition 22. *Given $m \in \mathbb{N}$ we define $.2_m^+ := (\Diamond^+ \Box^+ q \rightarrow \Box^+ \Diamond^+ q) \vee \Box^m \perp$ and $\text{IP}.2_m^+ := .2_m^+ \vee \nu p. \Diamond p$. Then the class of topological spaces X such that $X \models \text{IP}.2_m^+$ is not modally definable. In addition, whenever $m, k \geq 1$ and $m \neq k$ we have $\mu\mathbf{wK4} + \text{IP}.2_m^+ \neq \mu\mathbf{wK4} + \text{IP}.2_k^+$.*

Proof. It suffices to prove that the assumptions of Theorem 15 are satisfied for $\theta := .2_m^+$. In $\Diamond^+ \Box^+ q \rightarrow \Box^+ \Diamond^+ q$ we recognize a variant of the axiom .2 [7], but relative to the reflexive closure R^+ ; we call it $.2^+$, and this also explains the name $.2_m^+$. Thus, given a frame $\mathfrak{F} = (W, R)$ we have $\mathfrak{F} \models \text{IP}.2_m^+$ iff for all $w \in W$ one of the following holds:

- for all $u, v \in W$ such that wR^+u and wR^+v , there exists $t \in W$ such that uR^+t and vR^+t ;
- there exists no path of length $m + 1$ starting from w ;
- there exists an infinite path starting from w .

Consider, for all $n \in \mathbb{N}$, the frame $\mathfrak{F}_n^m := (W_n^m, R_n^m)$ depicted in Fig. 2. We can see that the $\mathfrak{F}_n^m$'s fulfil all the conditions of Theorem 15, so we are done (see Remark 17 for why the result applies to topological spaces). Finally, if $1 \leq m < k$ we can see that $\mathfrak{F}_1^{m-1} \models \text{IP}.2_m^+$ whereas $\mathfrak{F}_1^{m-1} \not\models \text{IP}.2_k^+$, and this proves that $\mu\mathbf{wK4} + \text{IP}.2_m^+ \neq \mu\mathbf{wK4} + \text{IP}.2_k^+$. $\square$

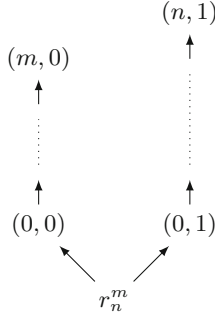


Fig. 2. The fork-like frame $\mathfrak{F}_n^m$

In Sect. 5 we will analyze these axioms further to see that they are well-behaved, but we find it appropriate to end this section by presenting an intuitive topological interpretation of the axiom $\text{IP}.2_0^+$, which reduces to $.2^+ \vee \nu p. \Diamond p$. Given a formula θ and a space X , we say that X is θ -imperfect if there exist two disjoint subspaces Y and Z of X such that $X = Y \cup Z$, $Y \models \theta$ and Z is dense-in-itself.

Proposition 23. *Let $\theta \in \mathcal{L}_\mu$ and X a topological space. Then $X \models \theta \vee \nu p. \Diamond p$ if and only X is θ -imperfect.*

Proof. From left to right, assume that $X \models \theta \vee \nu p. \Diamond p$. We set $Z := \{x \in X \mid X, x \models \nu p. \Diamond p\}$ and $Y := X \setminus Z$. The fixed point equation immediately gives $Z = \text{d}(Z)$, so Z is dense-in-itself. From $\text{d}(Z) \subseteq Z$ we also obtain that Z is closed and Y is open. Now, let $x \in Y$ and V be a valuation over Y . We have $X, V, x \models \theta \vee \nu p. \Diamond p$ and by construction $X, V, x \not\models \nu p. \Diamond p$, so $X, V, x \models \theta$. Since Y is open we obtain $Y, V, x \models \theta$. Therefore $Y \models \theta$.

From right to left, suppose that such a decomposition $X = Y \cup Z$ exists. Let $x \in X$ and V be a valuation over X . Suppose that $x \in Z$. Since Z is dense-in-itself we have $Z \subseteq d(Z) = \llbracket \Diamond p \rrbracket_{X,V[p:=Z]}$ so $Z \subseteq \llbracket \nu p. \Diamond p \rrbracket_{X,V}$. Therefore $X, V, x \models \nu p. \Diamond p$. Otherwise, we have $x \in Y$. If $x \notin \text{Int}(Y)$, then $x \in \text{Cl}(Z)$ and since $x \notin Z$ it follows that $x \in d(Z)$. We have seen that $X, V, z \models \nu p. \Diamond p$ for all $z \in Z$, so $X, V, x \models \Diamond \nu p. \Diamond p$, and then the fixed point equation gives $X, V, x \models \nu p. \Diamond p$. Otherwise we have $x \in \text{Int}(Y)$. Since $Y \models \theta$ and $\text{Int}(Y)$ is open in Y , we have $\text{Int}(Y) \models \theta$. Then $\text{Int}(Y), V, x \models \theta$ and since $\text{Int}(Y)$ is open, we finally get $X, V, x \models \theta$. In all cases we obtain $X, V, x \models \theta \vee \nu p. \Diamond p$ as desired. $\square$

Remark 24. By inspection of the proof for the left-to-right implication, we can also assume that Y is scattered and Z is perfect (i.e., closed and dense-in-itself).

In our example, the axiom $.2^+$ is known to define the class of extremally disconnected spaces [4] (see Definition 9). We thus obtain the following result:

Corollary 25. *The class of spaces that can be written as the disjoint union of an extremally disconnected subspace and a perfect subspace is not modally definable.*

4 Completeness for Imperfect Spaces

We have shown that there are μ -definable classes that are not modally definable, including infinitely many classes of imperfect spaces. We can make these examples even stronger by showing that the logics we have exhibited are complete for these classes. To this end, we construct the canonical model and use the technique of the final model applied by Fine and Zakharyashev to modal logic (see [5, 7]) and by Baltag et al. [3] to the μ -calculus. Central will be the notion of cofinal subframe logic.

Definition 26. Let $\mathfrak{F} = (W, R)$ be a Kripke frame. A subframe $\mathfrak{F}' = (W', R')$ of $\mathfrak{F}$ is called a *cofinal subframe* of $\mathfrak{F}$ if $w' \in W'$ and $w'Rw$ implies the existence of $u' \in W'$ such that wR^+u' . Given $\mathfrak{M}$ based on $\mathfrak{F}$ and $\mathfrak{M}'$ a submodel of $\mathfrak{M}$, we call $\mathfrak{M}'$ a *cofinal submodel* of $\mathfrak{M}$ if it is based on a cofinal subframe $\mathfrak{F}'$ of $\mathfrak{F}$.

Definition 27. Let $\mathbf{L}$ be an extension of $\mathbf{K}$. The logic $\mathbf{L}$ is called *cofinal subframe* if whenever $\mathfrak{F} \models \mathbf{L}$ and $\mathfrak{F}'$ is a cofinal subframe of $\mathfrak{F}$, we have $\mathfrak{F}' \models \mathbf{L}$.

Definition 28. Let $\mathbf{L}$ be an extension of $\mathbf{K}$. The *canonical model* of $\mathbf{L}$ is the model $\mathfrak{M} := (\Omega, R, V)$ with:

- Ω the set of maximal $\mathbf{L}$ -consistent subsets of $\mathcal{L}_\Diamond$;
- $R := \{(\Gamma, \Delta) \mid \Box\varphi \in \Gamma \implies \varphi \in \Delta\}$;
- $V(p) := \{\Gamma \in \Omega \mid p \in \Gamma\}$ for all $p \in \text{Prop}$.

The so-called *Truth Lemma* then establishes an equivalence between truth and membership at the worlds of $\mathfrak{M}$, i.e., $\mathfrak{M}, \Gamma \models \varphi$ if and only if $\varphi \in \Gamma$. Combined with the Lindenbaum's Lemma, this yields completeness of $\mathbf{L}$ with respect to its canonical model [6, Sect. 4.2]. If $\mathbf{L}$ is an extension of $\mu\mathbf{wK4}$, the canonical

model is defined in the same way, but the Truth Lemma then fails to hold. The technique designed in [3] consists in restricting oneself to an appropriate cofinal submodel of $\mathfrak{M}$. First, given a $\mathbf{L}$ -consistent formula ψ , one can construct a finite set of formulas Σ containing ψ , closed under subformulas, and closed (up to logical equivalence in $\mathbf{L}$) under negation and $\Diamond^+$. We then define the so-called Σ -final model as follows.

Definition 29. A world $\Gamma \in \Omega$ is called Σ -final if there exists $\varphi \in \Sigma \cap \Gamma$ such that whenever $\Gamma R \Delta$ and $\varphi \in \Delta$, we have $\Delta R \Gamma$. The Σ -final model is then the submodel $\mathfrak{M}_\Sigma$ of $\mathfrak{M}$ induced by $\Omega_\Sigma := \{\Gamma \in \Omega \mid \Gamma \text{ is } \Sigma\text{-final}\}$.

Under the right assumptions it can be proven that (1) $\mathfrak{M}_\Sigma$ is a cofinal submodel of $\mathfrak{M}$, (2) ψ belongs to some Σ -final world, (3) the Truth Lemma holds in $\mathfrak{M}_\Sigma$ for the formulas in Σ . This yields Kripke completeness of $\mu\mathbf{wK4}$ and, in fact, of any logic of the form $\mu\mathbf{wK4} + \theta$ where $\theta \in \mathcal{L}_\Diamond$ and $\mathbf{wK4} + \theta$ is a canonical and cofinal subframe logic. Note that this result is limited to extensions of $\mu\mathbf{wK4}$ with *basic* modal axioms. This is to be contrasted with the work of the present section, which offers completeness results for axioms with fixed points. First, we need a technical lemma.

Lemma 30. *If $\mu\mathbf{wK4} + \theta \vdash \varphi$, then $\mu\mathbf{wK4} + (\theta \vee \nu p.\Diamond p) \vdash \varphi \vee \nu p.\Diamond p$.*

Proof. We write $\mathbf{L}_0 := \mu\mathbf{wK4} + \theta$ and $\mathbf{L} := \mu\mathbf{wK4} + (\theta \vee \nu p.\Diamond p)$. We proceed by induction on the length of a proof.

- If φ is an axiom of $\mu\mathbf{wK4}$ or θ itself, then this is clear.
- Suppose that this holds for φ , and that $\mathbf{L}_0 \vdash \varphi[\psi_1, \dots, \psi_n/p_1, \dots, p_n]$ is obtained from $\mathbf{L}_0 \vdash \varphi$. By the induction hypothesis we have $\mathbf{L} \vdash \varphi \vee \nu p.\Diamond p$ and by substitution it follows that $\mathbf{L} \vdash \varphi[\psi_1, \dots, \psi_n/p_1, \dots, p_n] \vee \nu p.\Diamond p$.
- Suppose that this holds for φ and $\varphi \rightarrow \psi$, and that $\mathbf{L}_0 \vdash \psi$ is obtained from $\mathbf{L}_0 \vdash \varphi$ and $\mathbf{L}_0 \vdash \varphi \rightarrow \psi$. By the induction hypothesis we have $\mathbf{L} \vdash \varphi \vee \nu p.\Diamond p$ and $\mathbf{L} \vdash (\varphi \rightarrow \psi) \vee \nu p.\Diamond p$, and we deduce $\mathbf{L} \vdash \psi \vee \nu p.\Diamond p$.
- Suppose that this holds for φ , and that $\mathbf{L}_0 \vdash \Box\varphi$ is obtained from $\mathbf{L}_0 \vdash \varphi$. By the induction hypothesis we have $\mathbf{L} \vdash \varphi \vee \nu p.\Diamond p$, and by necessitation we obtain $\mathbf{L} \vdash \Box(\varphi \vee \nu p.\Diamond p)$ or equivalently $\mathbf{L} \vdash \Box(\neg\nu p.\Diamond p \rightarrow \varphi)$. By distribution we obtain $\mathbf{L} \vdash \Box\neg\nu p.\Diamond p \rightarrow \Box\varphi$, and the implication $\neg\nu p.\Diamond p \rightarrow \Box\neg\nu p.\Diamond p$ can also be derived in $\mu\mathbf{wK4}$. Therefore $\mathbf{L} \vdash \neg\nu p.\Diamond p \rightarrow \Box\varphi$, or equivalently $\mathbf{L} \vdash \Box\varphi \vee \nu p.\Diamond p$.
- Suppose that this holds for $\varphi \rightarrow \psi[\varphi/p]$ and that $\mathbf{L}_0 \vdash \varphi \rightarrow \nu p.\psi$ is obtained from $\mathbf{L}_0 \vdash \varphi \rightarrow \psi[\varphi/p]$. By the induction hypothesis we have

$$\mathbf{L} \vdash \nu p.\Diamond p \vee (\varphi \rightarrow \psi[\varphi/p])$$

and we prove that

$$\mu\mathbf{wK4} \vdash \psi[\varphi/p] \wedge \neg\nu p.\Diamond p \rightarrow \psi[\varphi \wedge \neg\nu p.\Diamond p/p].$$

Indeed, consider a $\mathbf{wK4}$ frame $\mathfrak{M}$ rooted in w and assume that $\mathfrak{M}, w \models \psi[\varphi/p] \wedge \neg\nu p.\Diamond p$. From $\models \neg\nu p.\Diamond p \rightarrow \Box\neg\nu p.\Diamond p$ we obtain $\mathfrak{M} \models \neg\nu p.\Diamond p$, so

$\mathfrak{M} \models \varphi \leftrightarrow (\varphi \wedge \neg \nu p. \Diamond p)$ and thus $\mathfrak{M} \models \psi[\varphi/p] \leftrightarrow \psi[\varphi \wedge \neg \nu p. \Diamond p/p]$. Therefore $\mathfrak{M}, w \models \psi[\varphi \wedge \neg \nu p. \Diamond p/p]$ and the result follows by Theorem 13. We then obtain

$$\mathbf{L} \vdash \varphi \wedge \neg \nu p. \Diamond p \rightarrow \psi[\varphi \wedge \neg \nu p. \Diamond p/p]$$

and by the induction rule we derive $\mathbf{L} \vdash \varphi \wedge \neg \nu p. \Diamond p \rightarrow \nu p. \psi$, or equivalently $\mathbf{L} \vdash \nu p. \Diamond p \vee (\varphi \rightarrow \nu p. \psi)$.

□

Theorem 31. *Let θ be a modal formula such that $\mathbf{wK4} + \theta$ is cofinal subframe and canonical. Then $\mu\mathbf{wK4} + \theta \vee \nu p. \Diamond p$ is Kripke complete.*

Proof. We write $\mathbf{L} := \mu\mathbf{wK4} + \theta \vee \nu p. \Diamond p$ and $\mathbf{L}_0 := \mu\mathbf{wK4} + \theta$. Suppose that $\mathbf{L} \not\models \neg \psi$ and let Σ be a finite set of formulas containing ψ and $\nu p. \Diamond p$, and with the closure properties enumerated above. We introduce:

- $\mathfrak{M} = (\Omega, R, V)$ the canonical model of $\mathbf{L}$, based on $\mathfrak{F} = (\Omega, R)$;
- $\mathfrak{M}_\Sigma = (\Omega_\Sigma, R_\Sigma, V_\Sigma)$ the Σ -final submodel of $\mathfrak{M}$, based on $\mathfrak{F}_\Sigma = (\Omega_\Sigma, R_\Sigma)$;
- $\mathfrak{M}_0 = (\Omega_0, R_0, V_0)$ the canonical model of $\mathbf{L}_0$, based on $\mathfrak{F}_0 = (\Omega_0, R_0)$ (see Fig. 3).

We know that $\mathfrak{F}_\Sigma$ is a cofinal subframe of $\mathfrak{F}$. In addition we have $\mathbf{L} \subseteq \mathbf{L}_0$, so for all maximal consistent sets Γ such that $\mathbf{L}_0 \subseteq \Gamma$ we also have $\mathbf{L} \subseteq \Gamma$; it is also clear that R and R_0 coincide over Ω_0 , so $\mathfrak{F}_0$ is a subframe of $\mathfrak{F}$. We then introduce

$$\Omega' := \{\Gamma \in \Omega_\Sigma \mid \mathfrak{M}_\Sigma, \Gamma \models \neg \nu p. \Diamond p\}$$

which induces a generated subframe $\mathfrak{F}' = (\Omega', R')$ of $\mathfrak{F}$. Indeed, if $\Gamma \in \Omega'$ and $\Gamma R_\Sigma^+ \Delta$, then since $\mathfrak{M}_\Sigma, \Gamma \models \neg \nu p. \Diamond p$ we have $\mathfrak{M}_\Sigma, \Delta \models \neg \nu p. \Diamond p$ too and thus $\Delta \in \Omega'$. Also, given $\Gamma \in \Omega'$ we have $\mathfrak{M}_\Sigma, \Gamma \models \neg \nu p. \Diamond p$, and we obtain $\neg \nu p. \Diamond p \in \Gamma$ by the Truth Lemma. If $\mathbf{L}_0 \vdash \varphi$, then $\mathbf{L} \vdash \varphi \vee \nu p. \Diamond p$ by Lemma 30, and from $\varphi \vee \nu p. \Diamond p \in \Gamma$ and $\neg \nu p. \Diamond p \in \Gamma$ we deduce $\varphi \in \Gamma$. Therefore $\mathbf{L}_0 \subseteq \Gamma$, and we obtain $\Gamma \in \Omega_0$. This proves that $\mathfrak{F}'$ is a subframe of $\mathfrak{F}_0$.

Now, suppose $\Gamma \in \Omega'$, $\Delta \in \Omega_0$ and $\Gamma R \Delta$. Since $\mathfrak{F}_\Sigma$ is cofinal in $\mathfrak{F}$, there exists $\Lambda \in \Omega_\Sigma$ such that $\Delta R^+ \Lambda$. By weak transitivity it follows that $\Gamma R^+ \Lambda$, and since $\mathfrak{F}'$ is a generated subframe of $\mathfrak{F}_\Sigma$ it follows that $\Lambda \in \Omega'$. Therefore $\mathfrak{F}'$ is a cofinal subframe of $\mathfrak{F}_0$. As observed in [3], that $\mathbf{wK4} + \theta$ is canonical implies that $\mu\mathbf{wK4} + \theta$ is canonical too, so $\mathfrak{F}_0 \models \theta$. Since $\mathbf{L}_0$ is cofinal subframe it follows that $\mathfrak{F}' \models \theta$ as well.

Now let $V_\bullet$ be a valuation over Ω_Σ and $\Gamma \in \Omega_\Sigma$. If $\Gamma \in \Omega'$, let $(\Omega', R', V'_\bullet)$ be the submodel of $(\Omega_\Sigma, R_\Sigma, V_\bullet)$ induced by Ω' . We know that $(\Omega', R', V'_\bullet), \Gamma \models \theta$, and since $\mathfrak{F}'$ is a generated subframe of $\mathfrak{F}_\Sigma$, it follows that $(\Omega_\Sigma, R_\Sigma, V_\bullet), \Gamma \models \theta$. Otherwise we have $\mathfrak{M}_\Sigma, \Gamma \models \nu p. \Diamond p$, but since $\nu p. \Diamond p$ contains no free variable, its truth value does not depend on the valuation V_Σ , and thus $(\Omega_\Sigma, R_\Sigma, V_\bullet), \Gamma \models \nu p. \Diamond p$. Therefore $\mathfrak{F}_\Sigma \models \theta \vee \nu p. \Diamond p$. As mentioned earlier, ψ is satisfiable on $\mathfrak{M}_\Sigma$ and this proves Kripke completeness. □

In order to prove topological completeness, we apply the technique used in [3] to turn a $\mathbf{wK4}$ frame into an appropriate topological space. The construction

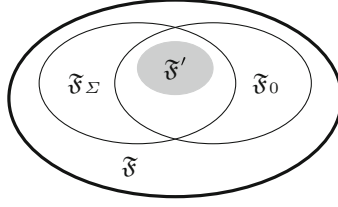


Fig. 3. The canonical frame of $\mathbf{L}$ and its subframes

essentially consists of replacing every reflexive point w of a frame by countably many copies of w , and to arrange them all into a dense-in-itself subspace, so as to mimic the reflexivity of w in a topological manner.

Definition 32. Let $\mathfrak{F} = (W, R)$ be a $\mathbf{wK4}$ frame. We denote by W^r the set of reflexive worlds of $\mathfrak{F}$, and by W^i the set of irreflexive worlds of $\mathfrak{F}$. We then introduce the *unfolding* of $\mathfrak{F}$ as the space $X_{\mathfrak{F}} := (W^r \times \omega) \cup (W^i \times \{\omega\})$ endowed with the topology $\tau_{\mathfrak{F}}$ of all sets U such that for all $(w, \alpha) \in U$:

1. there exists $n_{w,\alpha}^U < \omega$ such that for all $(u, \beta) \in X_{\mathfrak{F}}$, if wRu , uRw and $\beta \geq n_{w,\alpha}^U$ then $(u, \beta) \in U$;
2. if $(u, \beta) \in X_{\mathfrak{F}}$, wRu and not uRw then $(u, \beta) \in U$.

Proposition 33 ([3]). *The pair $(X_{\mathfrak{F}}, \tau_{\mathfrak{F}})$ is a topological space and the map $\pi: X_{\mathfrak{F}} \rightarrow W$ defined by $\pi(w, \alpha) := w$ is a surjective d-morphism.*

Theorem 34. *Let θ be a modal formula such that $\mathbf{wK4} + \theta$ is cofinal subframe and canonical. Then $\mu\mathbf{wK4} + \theta \vee \nu p.\Diamond p$ is topologically complete.*

Proof. Suppose that ψ is consistent in $\mu\mathbf{wK4} + \theta \vee \nu p.\Diamond p$. We keep the notations of the proof of Theorem 31. We introduce the spaces $X := X_{\mathfrak{F}_\Sigma}$, $Y := \pi^{-1}[\Omega']$ and $Z := X \setminus Y$. We prove that Y and Z satisfy the conditions of Proposition 23. First, we know that $\mathfrak{F}'$ is a generated subframe of $\mathfrak{F}_\Sigma$, so Ω' is open, and thus so is $\pi^{-1}[\Omega'] = Y$. In addition, since $\mathfrak{F}' \models \neg \nu p.\Diamond p$, the frame $\mathfrak{F}'$ is irreflexive, so $Y = \Omega' \times \{\omega\}$ and $\pi|_Y$ is injective. Since π is a d-morphism, the maps π and π^{-1} are continuous, and since Y is open, so are $\pi|_Y$ and $\pi|_Y^{-1}$. Therefore $\pi|_Y$ is a homeomorphism between Y and $\mathfrak{F}'$. From $\mathfrak{F}' \models \theta$ and Proposition 11, we conclude that $Y \models \theta$.

We then prove that Z is dense-in-itself. Let $(\Gamma, \alpha) \in Z$ and U be an open neighbourhood of (Γ, α) . From $(\Gamma, \alpha) \in Z$ we know that $\Gamma \notin \Omega'$, that is, $\mathfrak{M}_\Sigma, \Gamma \models \nu p.\Diamond p$. If $\alpha \neq \omega$, then Γ is reflexive. We select some $\beta \geq n_{w,\alpha}^U$ such that $\beta \neq \alpha$, and by definition of $n_{w,\alpha}^U$ we obtain $(\Gamma, \beta) \in U$. We also have $(\Gamma, \beta) \in Z$. Otherwise we have $\alpha = \omega$, and then Γ is irreflexive. From this and $\mathfrak{M}_\Sigma, \Gamma \models \nu p.\Diamond p$ we obtain the existence of $\Delta \neq \Gamma$ such that $\Gamma R \Delta$ and $\mathfrak{M}_\Sigma, \Delta \models \nu p.\Diamond p$. We set $\beta := n_{w,\alpha}^U$ if Δ is reflexive, and $\beta := \omega$ otherwise; we then have $(\Delta, \beta) \in Z$ by definition. Depending on whether $\Delta R \Gamma$ or not, we apply either item 1 or item 2 of Definition 32, and in both cases we obtain $(\Delta, \beta) \in U$. Both cases bring the existence of some element in $U \cap Z$ different from (Γ, α) , and we are done.

It follows that $X \models \theta \vee \nu p. \Diamond p$. We know that ψ is satisfiable on $\mathfrak{F}_\Sigma$, and since π is a d-morphism it follows by Proposition 5 that ψ is satisfiable on X as well. This concludes the proof. $\square$

In the following corollary, we finally apply these results to our examples.

Corollary 35. *For all $m \in \mathbb{N}$, the logic $\mu\mathbf{wK4} + \text{IP}.2_m^+$ is Kripke and topologically complete.*

Proof. Since $(\Diamond\Diamond p \rightarrow p \vee \Diamond p) \wedge .2_m^+$ is a Sahlqvist formula, the logic $\mathbf{L}_0 := \mathbf{wK4} + .2_m^+$ is canonical [6, Sect. 4.3]. In order to apply Theorem 31 and Theorem 34 we prove that $\mathbf{L}_0$ is cofinal subframe. Let $\mathfrak{F} = (W, R)$ be a $\mathbf{wK4}$ frame such that $\mathfrak{F} \models \mathbf{L}_0$, and let $\mathfrak{F}' = (W', R')$ be a cofinal subframe of $\mathfrak{F}$.

Let $w \in W'$. First, suppose that $\mathfrak{F}, w \models .2^+$. Then if wR^+u and wR^+v with $u, v \in W'$, we have by assumption uR^+t and vR^+t for some $t \in W$. Then since $\mathfrak{F}'$ is cofinal in $\mathfrak{F}$ we have tR^+t' for some $t' \in W'$, and thus uR^+t' and vR^+t' . This proves that $\mathfrak{F}', w \models .2^+$. Otherwise there exists a valuation V such that $\mathfrak{F}, V, w \not\models .2^+$, and since $\mathfrak{F}, V, w \models .2_m^+$ it follows that $\mathfrak{F}, V, w \models \Box^m \perp$. From this we deduce $\mathfrak{F}', w \models \Box^m \perp$. In both cases we obtain $\mathfrak{F}', w \models .2_m^+$. Therefore $\mathfrak{F}' \models \mathbf{L}_0$ and this proves the claim. $\square$

5 Conclusion

We have established some fundamental results regarding the expressivity of the topological μ -calculus as opposed to basic modal logic. We have shown that the latter is indeed more expressive axiomatically than the former, a fact that was surprisingly difficult to prove. Accordingly, the examples we have exhibited are optimal in the sense that they involve topologically complete logics, which we have argued correspond to natural classes of spaces. In particular, they are related to the perfect core of a space, equivalent to the unary version of the tangled derivative, perhaps the most fundamental topological fixed point. This suggests that we are only scratching the surface of the jungle of spatial μ -logics, and their classification could be a bold new direction in the study of topological modal logics.

Acknowledgements. We are grateful to Nick Bezhanishvili for his involvement in this project as a co-supervisor. We are also indebted to a number of anonymous referees who provided us with helpful feedback on an earlier version of this paper. DFD was partially supported by the FWO-FWF Lead Agency Grant G030620N.

References

1. Afshari, B., Leigh, G.: Cut-free completeness for modal μ -calculus. In: 32nd Annual ACM/IEEE Symposium on Logic in Computer Science LICS, pp. 1–12. IEEE Press (2017)
2. Aleksandroff, P.: Diskrete räume. *Matematicheskii Sbornik* **2**, 501–518 (1937)
3. Baltag, A., Bezhanishvili, N., Fernández-Duque, D.: The topological μ -calculus: completeness and decidability. In: 36th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2021, Rome, Italy, 29 June–2 July 2021, pp. 1–13. IEEE (2021). <https://doi.org/10.1109/LICS52264.2021.9470560>
4. van Benthem, J., Bezhanishvili, G.: Modal logics of space. In: Aiello, M., Pratt-Hartmann, I., Van Benthem, J. (eds.) *Handbook of Spatial Logics*, pp. 217–298. Springer, Dordrecht (2007). https://doi.org/10.1007/978-1-4020-5587-4_5
5. Bezhanishvili, G., Ghilardi, S., Jibladze, M.: An algebraic approach to subframe logics. Modal case. *Notre Dame J. Formal Logic* **52**(2), 187–202 (2011)
6. Blackburn, P., de Rijke, M., Venema, Y.: *Modal Logic*. Cambridge Tracts in Theoretical Computer Science, Cambridge University Press (2001)
7. Chagrov, A.V., Zakharyashev, M.: *Modal Logic*, Oxford Logic Guides, vol. 35. Clarendon Press, Consecutive, Oxford (1997)
8. Esakia, L.: Intuitionistic logic and modality via topology. *Ann. Pure Appl. Log.* **127**(1–3), 155–170 (2004). Provinces of logic determined
9. Esakia, L.: Weak transitivity-a restitution. *Logical Invest.* **8**, 244–245 (2001)
10. Fernández-Duque, D.: On the modal definability of simulability by finite transitive models. *Stud. Logica.* **98**(3), 347–373 (2011). <https://doi.org/10.1007/s11225-011-9339-x>
11. Fernández-Duque, D.: Tangled modal logic for spatial reasoning. In: Twenty-Second International Joint Conference on Artificial Intelligence (2011)
12. Fernández-Duque, D., Iliev, P.: Succinctness in subsystems of the spatial μ -calculus. *FLAP* **5**(4), 827–874 (2018). <https://www.collegepublications.co.uk/downloads/ifcolog00024.pdf>
13. Goldblatt, R., Hodkinson, I.: Spatial logic of tangled closure operators and modal μ -calculus. *Ann. Pure Appl. Log.* **168**(5), 1032–1090 (2017)
14. Goldblatt, R., Hodkinson, I.: The finite model property for logics with the tangle modality. *Stud. Logica.* **106**(1), 131–166 (2017). <https://doi.org/10.1007/s11225-017-9732-1>
15. Kozen, D.: Results on the propositional μ -calculus. *Theor. Comput. Sci.* **27**(3), 333–354 (1983)
16. Kudinov, A., Shehtman, V.: Derivational modal logics with the difference modality. In: Bezhanishvili, G. (ed.) *Leo Esakia on Duality in Modal and Intuitionistic Logics*. OCL, vol. 4, pp. 291–334. Springer, Dordrecht (2014). https://doi.org/10.1007/978-94-017-8860-1_11
17. McKinsey, J.C.C., Tarski, A.: The algebra of topology. *Ann. Math.* 141–191 (1944)
18. Santocanale, L., Venema, Y.: Completeness for flat modal fixpoint logics. *Ann. Pure Appl. Log.* **162**(1), 55–82 (2010)
19. Santocanale, L.: Completions of μ -algebras. *Ann. Pure Appl. Log.* **154**(1), 27–50 (2008)
20. Shehtman, V.: “Everywhere” and “here”. *J. Appl. Non Class. Logics* **9**(2–3), 369–379 (1999). <https://doi.org/10.1080/11663081.1999.10510972>
21. Walukiewicz, I.: Completeness of Kozen’s axiomatisation of the propositional μ -calculus. *Inf. Comput.* **157**(1–2), 142–182 (2000). <https://doi.org/10.1006/inco.1999.2836>



Correspondence Theory for Generalized Modal Algebras

Zhiguang Zhao^(✉) 

Taishan University, Tai'an 271000, China
zhaozhiguang23@gmail.com

Abstract. In the present paper, we give a systematic study of the correspondence theory of generalized modal algebras and generalized modal spaces, in the spirit of [3, 6]. The special feature of the present paper is that in the proof of the (right-handed) topological Ackermann lemma, the admissible valuations are not the clopen valuations anymore, but values in the set $\mathcal{D}_K(X)$ which are only closed and satisfy additional properties, not necessarily open. This situation is significantly different from existing settings using Stone/Priestley-like dualities, where all admissible valuations are clopen valuations.

Keywords: generalized modal algebra · generalized modal space · duality theory · correspondence theory

1 Introduction

Generalized Boolean algebras are the $(\rightarrow, \wedge, \vee, \top)$ -subreducts of Boolean algebras, i.e. its logic counterpart is the $(\neg, \perp)$ -free fragment of classical propositional logic. It is first studied by Stone [12]. Our treatment of generalized Boolean algebras follow the spirit of Tarski algebras [1–7], which are the $(\rightarrow, \vee, \top)$ -subreducts of Boolean algebras, therefore our treatment can be seen as Tarski algebras with a conjunction.

In [3, 6], some formulas defined in the language of modal Tarski algebras and subordination Tarski algebras are characterized in terms of first-order conditions on the dual modal Tarski spaces or subordination Tarski spaces. We follow their approach to study the correspondence theory for generalized modal algebras (which are the $(\rightarrow, \wedge, \vee, \top, \Box)$ -subreducts of modal algebras) and generalized modal spaces.

The present paper aims at obtaining similar but more general results in the setting of generalized modal algebras, using algorithmic correspondence theory [8, 9], which uses an algorithm ALBA to transform an input formula/inequality in the language of certain algebras into its first-order correspondent on the dual

The research of the author is supported by the Chinese National Social Science Fund of China (Project no. 21CZX049) and the Taishan Young Scholars Program of the Government of Shandong Province, China (No. tsqn201909151).

topological spaces. The special feature of the present paper is that in the proof of the right-handed topological Ackermann lemma, the admissible valuations are not the clopen valuations anymore, but values in the set $\mathcal{D}_{\mathcal{K}}(X)$ which are only closed and satisfy additional properties, not necessarily open. This situation is significantly different from the Stone/Priestley-like duality settings, where all admissible valuations are clopen valuations. The present work can also be taken as a step towards correspondence theory for logics which are not based on bounded algebras.

The paper is organized as follows: Sect. 2 gives the preliminaries on generalized modal algebras, generalized modal spaces and their duality on the object level. Section 3 gives the language to describe generalized modal algebras and generalized modal spaces, as well as its semantics. Section 4 gives the expanded language that the algorithm ALBA manipulate as well as the first-order correspondence language. Section 5 defines the inductive inequalities for our language. Section 6 describes the algorithm ALBA that computes the first-order correspondents of the modal formulas. Section 7 shows that ALBA succeeds on inductive inequalities. Section 8 proves that ALBA is sound with respect to admissible valuations. Section 9 gives the proof of the right-handed topological Ackermann lemma. Section 10 gives an example how ALBA executes. Section 11 gives some concluding remarks.

2 Preliminaries

In the present section, we give preliminaries on generalized Boolean algebras, generalized Stone spaces, generalized modal algebras, generalized modal spaces and their object-level duality. For more details, see [12].

2.1 Generalized Boolean Algebras

Definition 1. A generalized Boolean algebra is a tuple $A = (A, \rightarrow, \wedge, \vee, \top)$ such that $(A, \rightarrow, \wedge, \vee, \top)$ is a Brouwerian algebra, and $\vee$ satisfies $a \vee b = (a \rightarrow b) \rightarrow b$.

It is clear that a generalized Boolean algebra satisfies all the classical propositional logic laws that do not involve $\perp$ or $\neg$. In a generalized Boolean algebra we can define an order relation $\leq$ by $a \leq b$ iff $a = a \wedge b$. A non-empty subset F of A is called a *filter* if $\top \in F$ and F is upward-closed (i.e. if $a \in F$ and $a \leq b$ then $b \in F$) and for any $a, b \in A$, if $a, b \in F$, then $a \wedge b \in F$. A proper filter $F \subsetneq A$ is *maximal* if for any filter H such that $F \subseteq H$, we have $F = H$ or $H = A$. The set of all maximal filters of A is denoted by $Ul(A)$. Given a full set X , we use Y^c to denote the complement of Y relative to X , i.e. $X - Y$.

2.2 Generalized Stone Spaces

In what follows, a topological space $(X, \tau_{\mathcal{K}})$ with basis $\mathcal{K}$ will be denoted by $(X, \mathcal{K})$.

Definition 2. A generalized Stone space is a topological space $(X, \mathcal{K})$ such that:

- For every $x, y \in X$, if $x \neq y$, then there exists $U \in \mathcal{K}$ such that $x \in U$ and $y \notin U$;
- $\mathcal{K}$ is a basis of compact subsets for the topology $\tau_{\mathcal{K}}$;
- For any $A, B \in \mathcal{K}$, $A \cap B^c \in \mathcal{K}$;
- For any $A, B \in \mathcal{K}$, $A \cup B \in \mathcal{K}$;
- For any closed subset Y of X , and for any downward-directed subfamily $\mathcal{I}$ of $\mathcal{K}$, if $Y \cap U \neq \emptyset$ for each $U \in \mathcal{I}$, then $Y \cap \bigcap \mathcal{I} \neq \emptyset$.

Given a generalized Stone space $(X, \mathcal{K})$, the family $\mathcal{D}_{\mathcal{K}}(X) = \{U \subseteq X : U^c \in \mathcal{K}\}$ with the implication defined by $U \Rightarrow V = U^c \cup V$, conjunction defined by intersection, disjunction defined by union, for each $U, V \in \mathcal{D}_{\mathcal{K}}(X)$, is a generalized Boolean algebra called the dual generalized Boolean algebra of $(X, \mathcal{K})$.

Condition 5. of Definition 2 can be expressed in the following way (see [6]):

- For any closed subset Y of X , and for any directed subfamily $\mathcal{F}$ of $\mathcal{D}_{\mathcal{K}}(X)$, if $Y \subseteq \bigcup \{U : U \in \mathcal{F}\}$, then $Y \subseteq U$, for some $U \in \mathcal{F}$.

2.3 Topological Duality Between Generalized Boolean Algebras and Generalized Stone Spaces

If A is a generalized Boolean algebra, then the family $\mathcal{K}_A = \{(\sigma(a))^c : a \in A\}$ is a basis for a topology $\tau_{\mathcal{K}_A}$ defined on $Ul(A)$, where $\sigma(a) = \{P \in Ul(A) : a \in P\}$. Generalizing the results given in [7] we have that the pair $(Ul(A), \mathcal{K}_A)$ is a generalized Stone space and the map $\sigma : A \rightarrow \mathcal{D}_{\mathcal{K}_A}(Ul(A))$ is an algebra isomorphism.

Let $(X, \mathcal{K})$ be a generalized Stone space. Then the map $\varepsilon : X \rightarrow Ul(\mathcal{D}_{\mathcal{K}}(X))$ given by $\varepsilon(x) = \{U \in \mathcal{D}_{\mathcal{K}}(X) : x \in U\}$ is a homeomorphism between the topological spaces $(X, \mathcal{K})$ and $(Ul(\mathcal{D}_{\mathcal{K}}(X)), \mathcal{K}_{\mathcal{D}_{\mathcal{K}}(X)})$.

2.4 Adding Modality $\Box$

Definition 3. A generalized modal algebra is a tuple $A = (A, \rightarrow, \wedge, \vee, \top, \Box)$ such that $(A, \rightarrow, \wedge, \vee, \top)$ is a generalized Boolean algebra, $\Box \top = \top$ and $\Box(a \wedge b) = \Box a \wedge \Box b$ for all $a, b \in A$.

In what follows, we use R to denote a binary relation on the set X , $R(x)$ to denote $\{y \in X \mid Rxy\}$, $R[Y]$ to denote $\{y \in X \mid (\exists x \in Y)(Rxy)\}$ and $\Box U$ to denote $\{x \in X : R(x) \subseteq U\}$.

Definition 4. A generalized modal space is a structure $(X, \mathcal{K}, R)$ such that $(X, \mathcal{K})$ is a generalized Stone space, and

- For each $x \in X$, $R(x)$ is closed;
- $\Box U \in \mathcal{D}_{\mathcal{K}}(X)$, for each $U \in \mathcal{D}_{\mathcal{K}}(X)$.

The duality between generalized Boolean algebras and generalized Stone spaces can be naturally extended to generalized modal algebras and generalized modal spaces.

2.5 Some Useful Propositions

The following propositions will be useful in the proof of the topological Ackermann lemma in Sect. 9.2.

Lemma 1 (Generalizing Lemma 2.7 in [6]). *Let $(X, \mathcal{K})$ be a generalized Stone space. Let $\{U_i : i \in I\}$ and $\{V_j : j \in J\}$ be non-empty families of $\mathcal{D}_{\mathcal{K}}(X)$ such that $\bigcap \{U_i : i \in I\} \subseteq \bigcup \{V_j : j \in J\}$, then there exist $U_1, \dots, U_n$ and $V_1, \dots, V_k$ such that $U_1 \cap \dots \cap U_n \subseteq V_1 \cup \dots \cup V_k$.*

Proposition 1 (Generalizing Proposition 4.4 in [6]).

Let $(X, \mathcal{K}, R)$ be a generalized modal space. Then $R[Y]$ is a closed subset for each closed subset Y of X .

3 Syntax and Semantics

In the present section, we give the syntax and semantics of the logic formulas for generalized modal algebras and generalized modal spaces. We follow the presentation of [13].

3.1 Language and Syntax

Definition 5. *Given a countable set Prop of propositional variables, the generalized modal language $\mathcal{L}$ is defined as follows:*

$$\varphi ::= p \mid \top \mid \varphi \rightarrow \varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \Box \varphi,$$

where $p \in \text{Prop}$. We use the notation $\bar{p}$ to denote a list of propositional variables and use $\varphi(\bar{p})$ to indicate that the propositional variables occur in φ are all in $\bar{p}$. We call a formula pure if it does not contain propositional variables. We use the notation $\bar{\theta}$ to indicate a finite list of formulas. We use the notation $\theta(\eta/p)$ to indicate uniformly substituting p by η .

We will find it convenient to use inequalities of the form $\varphi \leq \psi$ and quasi-inequalities of the form $\varphi_1 \leq \psi_1 \ \& \ \dots \ \& \ \varphi_n \leq \psi_n \Rightarrow \varphi \leq \psi$ in the algorithm. Intuitively, $\varphi \leq \psi$ expresses the model-level truth of the implicative formula $\varphi \rightarrow \psi$.

3.2 Semantics

We interpret formulas on the dual generalized modal spaces, with two kinds of valuations, namely *admissible valuations* which interpret propositional variables as elements in $\mathcal{D}_{\mathcal{K}}(X)$ (i.e. interpret them as elements of the dual generalized modal algebras), and *arbitrary valuations* which interpret propositional variables as arbitrary subsets of the space. Notice that the admissible valuations are not necessarily clopen in the topology (they are only closed and satisfy additional conditions), in contrast to existing settings in algorithmic correspondence theory [9, 10] where Stone/Priestley-like dualities are used.

Definition 6. In a generalized modal space $(X, \mathcal{K}, R)$, (we abuse notation to use X to denote the space), we call X the domain of the space.

- An admissible model is a pair $M = (X, V)$ where $V : \text{Prop} \rightarrow \mathcal{D}_{\mathcal{K}}(X)$ is an admissible valuation on X .
- An arbitrary model is a pair $M = (X, V)$ where $V : \text{Prop} \rightarrow P(X)$ is an arbitrary valuation on X such that for all propositional variables p , $V(p)$ is an arbitrary subset of X .

Given a valuation V , a propositional variable $p \in \text{Prop}$, a subset $A \subseteq X$, we can define V_A^p , the p -variant of V as follows: $V_A^p(q) = V(q)$ for all $q \neq p$ and $V_A^p(p) = A$.

Now the satisfaction relation can be defined as follows: given any generalized modal space $(X, \mathcal{K}, R)$, any valuation V on X , any $w \in X$,

$$\begin{aligned}
 X, V, w \Vdash p & \quad \text{iff } w \in V(p) \\
 X, V, w \Vdash \top & \quad : \text{ always} \\
 X, V, w \Vdash \varphi \rightarrow \psi & \quad \text{iff } X, V, w \not\Vdash \varphi \text{ or } X, V, w \Vdash \psi \\
 X, V, w \Vdash \varphi \wedge \psi & \quad \text{iff } X, V, w \Vdash \varphi \text{ and } X, V, w \Vdash \psi \\
 X, V, w \Vdash \varphi \vee \psi & \quad \text{iff } X, V, w \Vdash \varphi \text{ or } X, V, w \Vdash \psi \\
 X, V, w \Vdash \Box \varphi & \quad \text{iff } \forall v (Rwv \Rightarrow X, V, v \Vdash \varphi)
 \end{aligned}$$

For any formula φ , we let $V(\varphi) = \{w \in X \mid X, V, w \Vdash \varphi\}$ denote the truth set of φ in (X, V) .

- The formula φ is globally true on (X, V) (notation: $X, V \Vdash \varphi$) if $X, V, w \Vdash \varphi$ for every $w \in W$.
- We say that φ is admissibly valid on a generalized modal space X (notation: $X \Vdash_{\mathcal{D}_{\mathcal{K}}} \varphi$) if φ is globally true on (X, V) for every admissible valuation V .
- We say that φ is valid on a generalized modal space X (notation: $X \Vdash_P \varphi$) if φ is globally true on (X, V) for every arbitrary valuation V .

For the semantics of inequalities and quasi-inequalities,

- $X, V \Vdash \varphi \leq \psi$ iff $V(\varphi) \subseteq V(\psi)$;
- $X, V \Vdash \varphi_1 \leq \psi_1 \ \& \ \dots \ \& \ \varphi_n \leq \psi_n \Rightarrow \varphi \leq \psi$ iff $X, V \Vdash \varphi \leq \psi$ holds whenever $X, V \Vdash \varphi_i \leq_i \psi_i$ for all $i = 1, \dots, n$.

The definitions of validity are similar to formulas.

4 Preliminaries on Algorithmic Correspondence

In this section, we give preliminaries on the correspondence algorithm ALBA in the style of [9, 13]. The algorithm ALBA transforms the input inequality $\varphi \leq \psi$ into an equivalent pure quasi-inequality which contains no propositional variable, and therefore can be translated into the first-order correspondence language via the standard translation of the expanded language (see page 6).

In the remainder of the paper, we will define an expanded language which the algorithm will manipulate (Sect. 4.1), define the first-order correspondence language of the expanded language and the standard translation (Sect. 4.2). We give the definition of inductive inequalities (Sect. 5), define a version of the algorithm ALBA (Sect. 6), and show its success on inductive inequalities (Sect. 7) and soundness with respect to admissible valuations (Sect. 8).

4.1 The Expanded Language

In the present subsection, we give the definition of the expanded language, which will be used in ALBA:

$$\varphi ::= p \mid \mathbf{i} \mid \mathbf{m} \mid \perp \mid \top \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \rightarrow \varphi \mid \Box \varphi \mid \Diamond \varphi$$

where $\mathbf{i} \in \text{Nom}$ is called a *nominal*, and $\mathbf{m} \in \text{CoNom}$ is called a *conominal*. For $\mathbf{i}$, it is interpreted as a singleton set, and $\mathbf{m}$ is interpreted as the complement of a singleton set. For $\Diamond$, it is interpreted as the diamond modality on the inverse relation R^{-1} .

For the semantics of the expanded language, the valuation V is extended to $\text{Prop} \cup \text{Nom} \cup \text{CoNom}$ such that $V(\mathbf{i})$ is a singleton for each $\mathbf{i} \in \text{Nom}$ and $V(\mathbf{m})$ is the complement of a singleton for each $\mathbf{m} \in \text{CoNom}$.¹ The additional semantic clauses can be given as follows:

$$\begin{aligned} X, V, w \Vdash \perp & : \text{ never} \\ X, V, w \Vdash \mathbf{i} & \text{ iff } V(\mathbf{i}) = \{w\} \\ X, V, w \nVdash \mathbf{m} & \text{ iff } V(\mathbf{m}) = \{w\}^c \\ X, V, w \Vdash \Diamond \varphi & \text{ iff } \exists v (Rvw \text{ and } X, V, v \Vdash \varphi) \end{aligned}$$

4.2 The First-Order Correspondence Language and the Standard Translation

In the first-order correspondence language, we have a binary predicate symbol R corresponding to the binary relation in the generalized modal space, a set of unary predicate symbols P corresponding to each propositional variable p , a set of individual symbols i corresponding to each nominal $\mathbf{i}$, and a set of individual symbols m corresponding to each conominal $\mathbf{m}$.

Definition 7. *The standard translation of the expanded language is defined as follows:*

$$\begin{aligned} ST_x(p) &:= Px & ST_x(\varphi \wedge \psi) &:= ST_x(\varphi) \wedge ST_x(\psi) \\ ST_x(\perp) &:= \perp & ST_x(\varphi \vee \psi) &:= ST_x(\varphi) \vee ST_x(\psi) \\ ST_x(\top) &:= \top & ST_x(\varphi \rightarrow \psi) &:= ST_x(\varphi) \rightarrow ST_x(\psi) \\ ST_x(\mathbf{i}) &:= x = i & ST_x(\Box \varphi) &:= \forall y (Rxy \rightarrow ST_y(\varphi)) \\ ST_x(\mathbf{m}) &:= x \neq m & ST_x(\Diamond \varphi) &:= \exists y (Ryx \wedge ST_y(\varphi)) \end{aligned}$$

¹ Notice that we allow admissible valuations to interpret nominals as singletons, even if singletons might not be in $\mathcal{D}_K(X)$. The admissibility restrictions are only for the propositional variables.

$$ST(\varphi \leq \psi) := \forall x(ST_x(\varphi) \rightarrow ST_x(\psi))$$

$$ST(\varphi_1 \leq \psi_1 \ \& \ \dots \ \& \ \varphi_n \leq \psi_n \Rightarrow \varphi \leq \psi) := ST(\varphi_1 \leq \psi_1) \wedge \dots \wedge ST(\varphi_n \leq \psi_n) \rightarrow ST(\varphi \leq \psi)$$

It is easy to see that this translation is correct:

Proposition 2. *For any generalized modal space X , any valuation V on X , any $w \in X$ and any expanded language formula φ ,*

$$X, V, w \Vdash \varphi \text{ iff } X, V \models ST_x(\varphi)[w].$$

Proposition 3. *For any generalized modal space X , any valuation V on X , and inequality Ineq , quasi-inequality Quasi ,*

$$X, V \Vdash \text{Ineq} \text{ iff } X, V \models ST(\text{Ineq});$$

$$X, V \Vdash \text{Quasi} \text{ iff } X, V \models ST(\text{Quasi}).$$

5 Inductive Inequalities for Generalized Modal Algebras

In this section, we define inductive inequalities for generalized modal algebras and generalized modal spaces.

We first define positive formulas with propositional variables in $A \subseteq \text{Prop}$ as follows:

$$\text{POS}_A ::= p \mid \top \mid \Box \text{POS}_A \mid \text{POS}_A \wedge \text{POS}_A \mid \text{POS}_A \vee \text{POS}_A$$

where $p \in A \subseteq \text{Prop}$.

Then we define the dependence order on propositional variables as any irreflexive and transitive binary relation $<_\Omega$ on them. Then we define the PIA formulas² with main variable p as follows:

$$\text{PIA}_p ::= p \mid \top \mid \Box \text{PIA}_p \mid \text{POS}_{A_p} \rightarrow \text{PIA}_p$$

where $A_p = \{q \in \text{Prop} \mid q <_\Omega p\}$ in PIA_p . Then we define the inductive antecedent as follows:

$$\text{Ant} ::= \text{PIA}_p \mid \text{Ant} \wedge \text{Ant} \mid \text{Ant} \vee \text{Ant}$$

where $p \in \text{Prop}$. Then we define the inductive succedent as follows:

$$\text{Suc} ::= p \mid \top \mid \text{PIA}_q \rightarrow \text{Suc} \mid \Box \text{Suc} \mid \text{Suc} \wedge \text{Suc} \mid \text{Suc} \vee \text{Suc}$$

where $p, q \in \text{Prop}$.

² For the meaning of PIA formulas, we refer the reader to [11, Remark 3.24].

Finally, an Ω -inductive inequality is an inequality of the form $\text{Ant} \leq \text{Suc}$ where each propositional variable in the inequality both occur positively and negatively³. An inductive inequality is an Ω -inductive inequality for some $<_{\Omega}$.

Remark 1. The definition of inductive formulas here can be seen as the special case in general settings where all propositional variables are of order-type 1 (see [13, Definition 4.1]), which means that we will compute their minimal valuations rather than maximal valuations and use only the right-handed Ackermann rule.

6 Algorithm

In the present section, we define the algorithm ALBA which compute the first-order correspondence of the input inequality in the style of [9]. The algorithm ALBA proceeds in three stages. Firstly, ALBA receives an inequality $\text{Ant} \leq \text{Suc}$ as input.

1. Preprocessing and First approximation:

- (a) We apply the following distribution rules exhaustively:
 - In Ant , rewrite every subformula of the former form into the latter form:
 - $\alpha \wedge (\beta \vee \gamma), (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$
 - $(\beta \vee \gamma) \wedge \alpha, (\beta \wedge \alpha) \vee (\gamma \wedge \alpha)$
 - In Suc , rewrite every subformula of the former form into the latter form:
 - $\alpha \rightarrow \beta \wedge \gamma, (\alpha \rightarrow \beta) \wedge (\alpha \rightarrow \gamma)$
 - $\Box(\alpha \wedge \beta), \Box\alpha \wedge \Box\beta$
 - $\alpha \vee (\beta \wedge \gamma), (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$
 - $(\beta \wedge \gamma) \vee \alpha, (\beta \vee \alpha) \wedge (\gamma \vee \alpha)$
- (b) Apply the splitting rules:

$$\frac{\alpha \leq \beta \wedge \gamma}{\alpha \leq \beta \quad \alpha \leq \gamma} \quad \frac{\alpha \vee \beta \leq \gamma}{\alpha \leq \gamma \quad \beta \leq \gamma}$$

Now for each obtained inequality $\varphi_i \leq \psi_i$, We apply the following first-approximation rule:

$$\frac{\varphi_i \leq \psi_i}{\mathbf{i}_0 \leq \varphi_i \ \& \ \psi_i \leq \mathbf{m}_0 \Rightarrow \mathbf{i}_0 \leq \mathbf{m}_0}$$

Now we focus on each set of the antecedent inequalities $\{\mathbf{i}_0 \leq \varphi_i, \psi_i \leq \mathbf{m}_0\}$, which we call a *system*.

³ We say that an occurrence of a propositional variable p in a formula φ is *positive*, if it is in the scope of an even number of the left-handside of implications, and *negative* if it is in the scope of an odd number of the left-handside of implications. For example, in $(p \rightarrow q) \rightarrow r$, p is positive because it is in the scope of 2 of the left-handside of implications, and r is also positive (in the scope of 0), and q is negative (in the scope of 1).

2. The reduction-elimination cycle:

In this stage, for each $\{\mathbf{i}_0 \leq \varphi_i, \psi_i \leq \mathbf{m}_0\}$, we apply the following rules to eliminate all the propositional variables:

(a) Splitting rules:

$$\frac{\alpha \leq \beta \wedge \gamma}{\alpha \leq \beta \quad \alpha \leq \gamma} \quad \frac{\alpha \vee \beta \leq \gamma}{\alpha \leq \gamma \quad \beta \leq \gamma}$$

(b) Residuation rules:

$$\frac{\alpha \leq \Box \beta}{\Diamond \alpha \leq \beta} \quad \frac{\alpha \leq \beta \rightarrow \gamma}{\alpha \wedge \beta \leq \gamma}$$

(c) Approximation rules:

$$\frac{\Box \alpha \leq \mathbf{m}}{\alpha \leq \mathbf{n} \quad \Box \mathbf{n} \leq \mathbf{m}} \quad \frac{\alpha \rightarrow \beta \leq \mathbf{m}}{\mathbf{j} \leq \alpha \quad \beta \leq \mathbf{n} \quad \mathbf{j} \rightarrow \mathbf{n} \leq \mathbf{m}}$$

The nominals and conominals introduced by the approximation rules must not occur in the system before applying the rule.

(d) Deleting rule: delete inequalities of the form $\alpha \leq \top$.

(e) The right-handed Ackermann rule.⁴ This rule eliminates propositional variables and is the core of the algorithm, the other rules are aimed at reaching a shape in which the rule can be applied. Notice that an important feature of this rule is that it is executed on the whole set of inequalities, and not on a single inequality.

$$\begin{array}{l} \text{The system} \left\{ \begin{array}{l} \theta_1 \leq p \\ \vdots \\ \theta_n \leq p \\ \eta_1 \leq \iota_1 \\ \vdots \\ \eta_m \leq \iota_m \end{array} \right. \\ \text{is replaced by} \left\{ \begin{array}{l} \eta_1((\theta_1 \vee \dots \vee \theta_n)/p) \leq \iota_1((\theta_1 \vee \dots \vee \theta_n)/p) \\ \vdots \\ \eta_m((\theta_1 \vee \dots \vee \theta_n)/p) \leq \iota_m((\theta_1 \vee \dots \vee \theta_n)/p) \end{array} \right. \end{array}$$

where:

- i. p does not occur in $\theta_1, \dots, \theta_n$;
- ii. Each η_i is positive, and each ι_i negative in p , for $1 \leq i \leq m$.

3. **Output:** If in the previous stage, for some systems, the algorithm gets stuck, i.e. some propositional variables cannot be eliminated, then the algorithm halts and output “failure”. Otherwise, each initial system after the first

⁴ Here we only have the right-handed Ackermann rule, because we could only guarantee the right-handed topological Ackermann lemma, due to the fact that admissible valuations are not clopen anymore, but only closed.

approximation has been reduced to a set of pure inequalities $\text{Reduce}(\mathbf{i}_0 \leq \varphi_i, \psi_i \leq \mathbf{m}_0)$, and then the output is a set of quasi-inequalities $\{\&\text{Reduce}(\mathbf{i}_0 \leq \varphi_i, \psi_i \leq \mathbf{m}_0) \Rightarrow \mathbf{i}_0 \leq \mathbf{m}_0\}_{i \in I}$. Then we can use the conjunction of the standard translations of the quasi-inequalities to obtain the first-order correspondence (notice that in the standard translation of each quasi-inequality, we need to universally quantify over all the individual variables).

7 Success of ALBA

In the present section, we show the success of ALBA on any inductive inequality $\varphi \leq \psi$.

Theorem 1. *ALBA succeeds on any inductive inequality $\varphi \leq \psi$ and outputs a pure quasi-inequality and a first-order formula.*

Proof. We check the shape of the inequality or system in each stage, for the input inequality $\text{Ant} \leq \text{Suc}$:

Stage 1. After applying the distribution rules, it is easy to see that Ant becomes the form $\bigvee \bigwedge \text{PIA}_p$, and Suc becomes of the form $\bigwedge \text{Suc}'$, where

$$\text{Suc}' ::= p \mid \top \mid \text{PIA}_q \rightarrow \text{Suc}' \mid \Box \text{Suc}' \mid \text{Suc}' \vee \text{Suc}'.$$

Then by applying the splitting rules, we get a set of inequalities of the form $\bigwedge \text{PIA}_p \leq \text{Suc}'$.

After the first approximation rule, each system is of the form $\{\mathbf{i}_0 \leq \bigwedge \text{PIA}_p, \text{Suc}' \leq \mathbf{m}_0\}$.

Stage 2. In this stage, we deal with each system $\{\mathbf{i}_0 \leq \bigwedge \text{PIA}_p, \text{Suc}' \leq \mathbf{m}_0\}$.

For the inequality $\mathbf{i}_0 \leq \bigwedge \text{PIA}_p$, by first applying the splitting rule for $\bigwedge$ and then exhaustively applying the residuation rules for $\Box$ and $\rightarrow$, we get inequalities of the form $\text{MinVal}_p \leq p$ or $\text{MinVal}_p \leq \top$, where

$$\text{MinVal}_p ::= \mathbf{i}_0 \mid \blacklozenge \text{MinVal}_p \mid \text{MinVal}_p \wedge \text{POS}_{A_p},$$

where $A_p = \{q \in \text{Prop} \mid q <_{\Omega} p\}$.

For the inequality $\text{Suc}' \leq \mathbf{m}_0$, by exhaustively applying the approximation rules for $\Box$ and $\rightarrow$ and the splitting rule for $\vee$, we get a set of inequalities, each of which is in the following form:

- $\mathbf{j} \leq \text{PIA}'_p$;
- $p \leq \mathbf{m}$;
- $\mathbf{i} \rightarrow \mathbf{n} \leq \mathbf{m}$;
- $\Box \mathbf{n} \leq \mathbf{m}$;

Now by applying the same strategy of the $\mathbf{i}_0 \leq \bigwedge \text{PIA}_p$ case to $\mathbf{j} \leq \text{PIA}'_p$, we get a set of inequalities of the following form (it is easy to see that by the positivity and negativity requirement of propositional variables in inductive inequalities, the first kind of inequality always exists for each propositional variable in the inequality):

- $\text{MinVal}_p \leq p$;
- $\text{MinVal}_p \leq \top$ (they are deleted by the deleting rule);
- $p \leq \mathbf{m}$;
- $\mathbf{i} \rightarrow \mathbf{n} \leq \mathbf{m}$;
- $\Box \mathbf{n} \leq \mathbf{m}$;

Now we are ready to apply the right-handed Ackermann rule to an Ω -minimal variable q to eliminate it. Then since there are only finitely many propositional variables, we can always find another Ω -minimal variable to eliminate. Finally we eliminate all propositional variables and get a pure quasi-inequality and its standard translation. Notice that during last phase of the algorithm applying the right-handed Ackermann rule, the right-hand side of an inequality is either p or $\mathbf{m}$ where p is only appearing in the inequalities with minimal valuations.

8 Soundness of ALBA

In this section we show the soundness of the algorithm with respect to the admissible valuations. The soundness proof follows the style of [9]. For most of the rules, the soundness proofs are the same to existing literature and hence are omitted, so we only give details for the proofs which are different, namely the right-handed Ackermann rule.

Theorem 2 (Soundness). *If ALBA runs according to the success proof on an input inductive inequality $\varphi \leq \psi$ and outputs a first-order formula $\text{FO}(\varphi \leq \psi)$, then for any generalized modal space $(X, \mathcal{K}, R)$,*

$$X \Vdash_{\mathcal{D}_K} \varphi \leq \psi \text{ iff } X \models \text{FO}(\varphi \leq \psi).$$

Proof. The proof goes similarly to [9, Theorem 8.1]. Let $\varphi \leq \psi$ denote the input inequality, let $\{\mathbf{i}_0 \leq \varphi_i \ \& \ \psi_i \leq \mathbf{m}_0 \Rightarrow \mathbf{i}_0 \leq \mathbf{m}_0\}_{i \in I}$ denote the set of quasi-inequalities after the first-approximation rule, let $\{\&\text{Reduce}(\mathbf{i}_0 \leq \varphi, \psi \leq \mathbf{m}_0) \Rightarrow \mathbf{i}_0 \leq \mathbf{m}_0\}_{i \in I}$ denote the set of quasi-inequalities after Stage 2, let $\text{FO}(\varphi \leq \psi)$ denote the standard translation of the quasi-inequalities in Stage 3 into first-order formulas, then it suffices to show the equivalence from (1) to (4) given below:

$$X \Vdash_{\mathcal{D}_K} \varphi \leq \psi \tag{1}$$

$$X \Vdash_{\mathcal{D}_K} \mathbf{i}_0 \leq \varphi_i \ \& \ \psi_i \leq \mathbf{m}_0 \Rightarrow \mathbf{i}_0 \leq \mathbf{m}_0, \text{ for all } i \in I \tag{2}$$

$$X \Vdash_{\mathcal{D}_K} \&\text{Reduce}(\mathbf{i}_0 \leq \varphi, \psi \leq \mathbf{m}_0) \Rightarrow \mathbf{i}_0 \leq \mathbf{m}_0, \text{ for all } i \in I \tag{3}$$

$$X \models \text{FO}(\varphi \leq \psi) \tag{4}$$

The equivalence between (1) and (2) follows from Proposition 4;

The equivalence between (2) and (3) follows from Propositions 5, 6;

The equivalence between (3) and (4) follows from Proposition 3.

In the remainder of this section, we prove the soundness of the rules in each stage.

Proposition 4. *The distribution rules, the splitting rules are sound in X , and the first-approximation rule is sound in X , i.e. (1) and (2) are equivalent.*

Proof. See [13, Proposition 6.2, 6.3].

The next step is to show the soundness of each rule of Stage 2. For each rule, before the application of this rule we have a system S , after applying the rule we get a system S' , the soundness of Stage 2 is then the equivalence of the following:

- $X \Vdash_{\mathcal{D}_K} \& S \Rightarrow \mathbf{i}_0 \leq \mathbf{m}_0$
- $X \Vdash_{\mathcal{D}_K} \& S' \Rightarrow \mathbf{i}_0 \leq \mathbf{m}_0$

where $\& S$ denote the meta-conjunction of inequalities of S . It suffices to show the following property:

- For any X , any admissible valuation V , if $X, V \Vdash S$, then there is an admissible valuation V' such that $V'(\mathbf{i}_0) = V(\mathbf{i}_0)$, $V'(\mathbf{m}_0) = V(\mathbf{m}_0)$ and $X, V' \Vdash S'$;
- For any X , any admissible valuation V' , if $X, V' \Vdash S'$, then there is an admissible valuation V such that $V(\mathbf{i}_0) = V'(\mathbf{i}_0)$, $V(\mathbf{m}_0) = V'(\mathbf{m}_0)$ and $X, V \Vdash S$.

Proposition 5. *The splitting rules, the approximation rules, the residuation rules and the deleting rule in Stage 2 are sound in both directions in X .*

Proof. The soundness proofs for the splitting rules, the approximation rules and the residuation rules are the same to the soundness of the same rules in [13, Lemma 6.5-6.7, 6.12]. The soundness of the deleting rule is trivial.

Proposition 6. *The right-handed Ackermann rule applied in the success proof is sound in X .*

This rule is the most complicated rule in the soundness proof of the algorithm, and indeed this proof is different from existing settings due to that the admissible valuations are not clopen anymore. We devote the next section to this proof.

9 Right-Handed Topological Ackermann Lemma

9.1 Analysis of the Right-Handed Ackermann Rule

We consider the application of the right-handed Ackermann rule. Before the application, each inequality in the system is of the following shape, where p is the current Ω -minimal propositional variable:

- $\text{MinVal}'_{p,1} \leq p, \dots, \text{MinVal}'_{p,n} \leq p$
- inequalities of the form $\text{MinVal}'_q \leq q$, where $q \neq p$
- inequalities of the form $p \leq \mathbf{m}_p$
- inequalities of the form $q \leq \mathbf{m}_q$, where $q \neq p$

- pure inequalities

here $\text{MinVal}'_r \in C_r$ for $r = p, q$, where C_r is defined as follows:

$$C_r ::= \mathbf{i} \mid \top \mid s \mid \blacklozenge C_r \mid \Box C_r \mid C_r \wedge C_r \mid C_r \vee C_r$$

where s is a propositional variable of dependence order below r . It is easy to see that each $\text{MinVal}'_{p,i}$ is pure (since all propositional variables below p are already eliminated), and MinVal'_q may or may not contain p .

Now denote $\bigvee_i \text{MinVal}'_{p,i}$ as V_p . After the application of the right-handed Ackermann rule, the system is of the following shape:

- inequalities of the form $\text{MinVal}'_q(V_p/p) \leq q$
- inequalities of the form $V_p \leq \mathbf{m}_p$
- inequalities of the form $q \leq \mathbf{m}_q$
- pure inequalities

It is easy to see that in the system, in each non-pure inequality, they are of the form $\text{MinVal}'_q(V_p/p) \leq q$ or $q \leq \mathbf{m}_q$, which still fall in the categories described as before the application of the right-handed Ackermann rule. Also, the first application of the right-handed Ackermann rule is a special case of the situation described above.

Therefore, it suffices to show that for the system

- $\text{MinVal}'_{p,1} \leq p, \dots, \text{MinVal}'_{p,n} \leq p$
- inequalities of the form $\text{MinVal}'_q \leq q$, where $q \neq p$
- inequalities of the form $p \leq \mathbf{m}_p$
- inequalities of the form $q \leq \mathbf{m}_q$, where $q \neq p$
- pure inequalities,

the application of the right-handed Ackermann rule on variable p is sound with respect to admissible valuations. Indeed, we can consider the system as follows (without loss of generality we can take the join of the minimal valuations for p to make them V_p):

- $V_p \leq p$
- inequalities of the form $\text{MinVal}'_q(p) \leq q$, where $q \neq p$ and $\text{MinVal}'_q(p)$ contains positive occurrences of p
- inequalities of the form $p \leq \mathbf{m}_p$
- inequalities that does not contain p .

9.2 Proof of Topological Ackermann Lemma

In what follows we denote $\mathcal{D}_K(X)$ as A , which means that we identify the generalized Boolean algebra $\mathcal{D}_K(X)$ on X and the dual generalized Boolean algebra A , as well as their elements. We use $C(X)$ to denote the set of closed elements in X . We also denote $\blacklozenge Y := R[Y]$ and $\Box Y := (R^{-1}(Y^c))^c$. Now we prove the following lemmas:

- Lemma 2.** 1. For any $Y \in C(X)$, Y is an intersection of a downward-directed collection of elements in A , i.e. $\forall Y \in C(X)$, $Y = \bigcap_i X_i$ for some downward-directed $\{X_i\}_{i \in I} \subseteq A$.
2. $A \subseteq C(X)$.
3. $\{x\} \in C(X)$ for any $x \in X$.
4. If $Y \in C(X)$, $\Diamond Y \in C(X)$.
5. $\Box \bigcap_i X_i = \bigcap \Box X_i$, for any $X_i \in P(X)$.
6. If $Y \in C(X)$, then $\Box Y \in C(X)$.

Proof. 1. By the fact that $\mathcal{K}$ is a basis of X , C is an intersection of elements in A , i.e. $\forall Y \in C(X)$, $Y = \bigcap_i Y_i$ for some $\{Y_j\}_{j \in J} \subseteq A$. Now define $\{X_i\}_{i \in I}$ to be the set of all finite intersections of elements in $\{Y_j\}_{j \in J}$. Since A is closed under taking finite intersections, it is easy to see that $\{X_i\}_{i \in I}$ is a subset of A and is downward-directed.

2. Trivial.
3. By Definition 2, singletons are closed.
4. By Proposition 1.
5. By the fact that $\Box$ is completely intersection preserving.
6. An easy corollary of items 1 and 5.

Lemma 3. Let $(X, \mathcal{K})$ be a generalized Stone space. Let $\{U_i : i \in I\}$ be non-empty downward-directed family in $C(X)$ and $V \in A$ such that $\bigcap \{U_i : i \in I\} \subseteq V$, then there exists an $i \in I$ such that $U_i \subseteq V$.

Proof. For each $i \in I$, consider $U_i \in C(X)$, there is a collection $\{X_{i,j}\}_{j \in J_i} \subseteq A$ such that $U_i = \bigcap_j X_{i,j}$, therefore we have $\bigcap_{i,j} X_{i,j} \subseteq V$, by Lemma 1, there are $X_{i_1, j_1}, \dots, X_{i_n, j_n}$ such that $X_{i_1, j_1} \cap \dots \cap X_{i_n, j_n} \subseteq V$, therefore $U_{i_1} \cap \dots \cap U_{i_n} \subseteq V$, by downward-directedness, there is an $i \in I$ such that $U_i \subseteq V$.

Lemma 4. $\Diamond \bigcap_i X_i = \bigcap \Diamond X_i$ for any non-empty downward-directed $\{X_i\}_{i \in I} \subseteq C(X)$.

Proof. The direction $\Diamond \bigcap_i X_i \subseteq \bigcap \Diamond X_i$ is easy. For the other direction, suppose there is a closed set $Y \subseteq X$ such that $\Diamond \bigcap_i X_i \subseteq Y$. Then there is a collection $\{Z_j\}_{j \in J} \in A$ such that $Y = \bigcap_j Z_j$. Therefore, $\Diamond \bigcap_i X_i \subseteq Z_j$ for all j . Thus $\bigcap_i X_i \subseteq \Box Z_j$ for all j . By Lemma 3 and the downward-directedness of X_i , we have that for all j and some k_j depending on j , $X_{k_j} \subseteq \Box Z_j$, i.e., $\Diamond X_{k_j} \subseteq Z_j$, so $\bigcap_i \Diamond X_i \subseteq Z_j$ for all j , so $\bigcap_i \Diamond X_i \subseteq \bigcap_j Z_j = Y$. Now take $Y = \Diamond \bigcap_i X_i$, we have $\bigcap_i \Diamond X_i \subseteq \Diamond \bigcap_i X_i$.

Lemma 5. For any formula $\alpha(\bar{q}, \bar{\mathbf{i}}, p)$ built up from nominals, $\top$, propositional variables using $\Diamond, \Box, \wedge, \vee$, consider the following valuation: $V(\bar{q}) = \bar{Y}$, where $\bar{Y} \in A$, $V(\bar{\mathbf{i}}) = \{x\}$, where $x \in X$, $V(p) = Z$, where $Z \in A$, then $V(\alpha(\bar{q}, \bar{\mathbf{i}}, p)) \in C(X)$.

Proof. By induction on the complexity of $\alpha(\bar{q}, \bar{\mathbf{i}}, p)$.

Lemma 6. *For any formula $\alpha(\bar{q}, \bar{\mathbf{i}}, p)$ built up from nominals, $\top$, propositional variables using $\Diamond, \Box, \wedge, \vee$, for any $\bar{Y} \in A$ corresponding to $\bar{q}$, any $\{x\}$ (where $x \in X$) corresponding to $\bar{\mathbf{i}}$, any downward-directed $\{Z_i\}_{i \in I} \subseteq C(X)$ corresponding to p , we have $\bigcap_i \beta(\bar{Y}, \{x\}, Z_i) = \beta(\bar{Y}, \{x\}, \bigcap_i Z_i)$.⁵*

Proof. By induction on the complexity of $\alpha(\bar{q}, \bar{\mathbf{i}}, p)$.

Lemma 7 (Right-handed topological Ackermann lemma). *Let θ be V_p , $\eta_i(p) \leq \iota_i$ be $\text{MinVal}'_q(p) \leq q$ or $p \leq \mathbf{m}_p$ as described on page 13 for $1 \leq i \leq m$. Then for any admissible valuation V , the following are equivalent:*

1. $V(\eta_i(\theta/p)) \subseteq V(\iota_i)$ for $1 \leq i \leq m$,
2. *there exists $Z \in A$ such that $V'(\theta) \subseteq Z$ and $V'(\eta_i(p)) \subseteq V'(\iota_i)$ for $1 \leq i \leq m$, where V' is the same as V except that $V'(p) = Z$.*

Proof. 2. to 1. is easy by monotonicity.

For 1. to 2., since p is the current Ω -minimal propositional variable, θ is pure. By Lemma 5 and the shape of θ , $V(\theta) = V'(\theta) \in C(X)$. Now we denote $V(q)$ as Q , then $Q \in A$, $V(\mathbf{m}_p) = \{m\}^c$ for some $m \in X$.

Therefore, from $V(\text{MinVal}'_q(\theta/p)) \subseteq V(q)$, we have that $\text{MinVal}'_q(V(\theta)) \subseteq Q$ ⁶, since $V(\theta) \in C(X)$, there is a downward-directed collection $\{X_i\}_{i \in I} \subseteq A$ such that $V(\theta) = \bigcap_i X_i$, so $\text{MinVal}'_q(\bigcap_i X_i) \subseteq Q$. Now by Lemma 6, $\text{MinVal}'_q(\bigcap_i X_i) = \bigcap_i \text{MinVal}'_q(X_i)$, so $\bigcap_i \text{MinVal}'_q(X_i) \subseteq Q$. Now by Lemma 5, it is easy to see that $\{\text{MinVal}'_q(X_i) \mid i \in I\}$ is a downward-directed set of elements in $C(X)$, so by Lemma 3, there is an $i \in I$ such that $\text{MinVal}'_q(X_i) \subseteq Q$.

For $p \leq \mathbf{m}_p$, we have that $V(\theta) \subseteq \{m\}^c$. Since $V(\theta) \in C(X)$, there are downward-directed $Y_j \in A$ ($j \in J$) such that $V(\theta) = \bigcap_j Y_j$, so $\bigcap_j Y_j \subseteq \{m\}^c$. So $m \notin \bigcap_j Y_j$, therefore there is a $j \in J$ such that $m \notin Y_j$, so $Y_j \subseteq \{m\}^c$.

Now take the intersection of all X_i s and Y_j s picked up for each $\eta_i(p) \leq \iota_i$, we get the Z as desired (this is possible since A is closed under taking finite intersections).

Now from the right-handed topological Ackermann lemma, it is easy to see the soundness of the right-handed Ackermann rule when executed as described in the success proof.

10 Example

Here we show the execution of an example from [6, Proposition 6.1]:

⁵ Here by $\beta(\bar{Y}, \{x\}, Z_i)$ we mean $V(\beta)$ under the valuation where $V(\bar{q}) = \bar{Y}$, $V(\bar{\mathbf{i}}) = \{x\}$, $V(p) = Z_i$.

⁶ Here by $\text{MinVal}'_q(V(\theta))$ we mean $V(\text{MinVal}'_q)$ where $V(p) = V(\theta)$.

Example 1. $\forall p(\top \leq \Box(\Box p \rightarrow p))$

$$\forall p \forall i_0 \forall m_0 (i_0 \leq \top \ \& \ \Box(\Box p \rightarrow p) \leq m_0 \Rightarrow i_0 \leq m_0)$$

$$\forall p \forall i_0 \forall m_0 \forall m_1 (i_0 \leq \top \ \& \ \Box m_1 \leq m_0 \ \& \ \Box p \rightarrow p \leq m_1 \Rightarrow i_0 \leq m_0)$$

$$\forall p \forall i_0 \forall m_0 \forall m_1 (\Box m_1 \leq m_0 \ \& \ \Box p \rightarrow p \leq m_1 \Rightarrow i_0 \leq m_0)$$

$$\forall p \forall i_0 \forall i_1 \forall m_0 \forall m_1 \forall m_2 (\Box m_1 \leq m_0 \ \& \ i_1 \rightarrow m_2 \leq m_1 \ \& \ i_1 \leq \Box p \ \& \ p \leq m_2 \Rightarrow i_0 \leq m_0)$$

$$\forall p \forall i_0 \forall i_1 \forall m_0 \forall m_1 \forall m_2 (\Box m_1 \leq m_0 \ \& \ i_1 \rightarrow m_2 \leq m_1 \ \& \ \blacklozenge i_1 \leq p \ \& \ p \leq m_2 \Rightarrow i_0 \leq m_0)$$

$$\forall i_0 \forall i_1 \forall m_0 \forall m_1 \forall m_2 (\Box m_1 \leq m_0 \ \& \ i_1 \rightarrow m_2 \leq m_1 \ \& \ \blacklozenge i_1 \leq m_2 \Rightarrow i_0 \leq m_0)$$

$$\forall i_0 \forall i_1 \forall m_0 \forall m_1 \forall m_2 (\Box(\{m_1\}^c) \subseteq \{m_0\}^c \wedge \{i_1\}^c \cup \{m_2\}^c \subseteq \{m_1\}^c \wedge \blacklozenge \{i_1\} \subseteq \{m_2\}^c \rightarrow \{i_0\} \subseteq \{m_0\}^c)^7$$

$$\forall i_0 \forall i_1 \forall m_0 \forall m_1 \forall m_2 (Rm_0 m_1 \wedge i_1 = m_2 = m_1 \wedge \neg Ri_1 m_2 \rightarrow i_0 \neq m_0)$$

$$\forall i_0 \forall i_1 (Ri_0 i_1 \rightarrow Ri_1 i_1)$$

11 Concluding Remarks

In the present paper, we give a systematic study of the correspondence theory of generalized modal algebras and generalized modal spaces, in the spirit of [3, 6]. The major difference of the present paper with existing literature can be summarized as follows:

- The kind of logic we study is not based on algebras that are bounded lattice expansions, but without the bottom. This difference makes it more difficult to give a duality, as given in [7].
- The duality we use is a spectral-like duality, where the spaces have a basis of compact open sets rather than clopen sets as in Stone/Priestley-type dualities, as used in existing literature on correspondence theory. As far as the author knows, this is the first paper to treat Sahlqvist-type correspondence theory between algebras and spectral-like spaces.
- The use of spectral-like spaces makes the admissible valuations different: they are not clopen valuations anymore, but the complement of certain compact open sets, i.e. certain closed sets with additional conditions. Therefore, in the proof of the right-handed topological Ackermann lemma, we have to prove it for the specific setting rather than adapting existing proofs of topological Ackermann lemmas immediately.

⁷ Here this is the set-theoretic way to identify the truth set of each formula in the expanded language, which is indeed an abbreviation of its first-order translation. We use this kind of notation here because the first-order translation is too long.

References

1. Abbott, J.C.: Implicational algebras. *Bulletin mathématique de la Société des Sciences Mathématiques de la République Socialiste de Roumanie* **11**(59)(1), 3–23 (1967)
2. Abbott, J.C.: Semi-Boolean algebra. *Matematiki Vesnik* **4**(19)(40), 177–198 (1967)
3. Celani, S.A.: Modal Tarski algebras. *Rep. Math. Logic* **39**, 113–126 (2005)
4. Celani, S.A.: Complete and atomic Tarski algebras. *Arch. Math. Log.* **58**(7–8), 899–914 (2019)
5. Celani, S.A.: Subordination Tarski algebras. *J. Appl. Non-Classical Logics* **29**(3), 288–306 (2019)
6. Celani, S.A.: Relational representation of quasi-modal Tarski algebras (2022, in preparation)
7. Celani, S.A., Cabrer, L.M.: Topological duality for Tarski algebras. *Algebra Universalis* **58**(1), 73–94 (2008)
8. Conradie, W., Ghilardi, S., Palmigiano, A.: Unified correspondence. In: Baltag, A., Smets, S. (eds.) *Johan van Benthem on Logic and Information Dynamics*. OCL, vol. 5, pp. 933–975. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06025-5_36
9. Conradie, W., Palmigiano, A.: Algorithmic correspondence and canonicity for distributive modal logic. *Ann. Pure Appl. Logic* **163**(3), 338–376 (2012)
10. Conradie, W., Palmigiano, A.: Algorithmic correspondence and canonicity for non-distributive logics. *Ann. Pure Appl. Logic* **170**(9), 923–974 (2019)
11. Palmigiano, A., Sourabh, S., Zhao, Z.: Sahlqvist theory for impossible worlds. *J. Log. Comput.* **27**(3), 775–816 (2017)
12. Stone, M.H.: Postulates for Boolean algebras and generalized Boolean algebras. *Am. J. Math.* **57**(4), 703–732 (1935)
13. Zhao, Z.: Algorithmic correspondence for hybrid logic with binder. *Logic J. IGPL* (2021). jzab029



Tense Logics over Lattices

Xiaoyang Wang  and Yanjing Wang  

Department of Philosophy, Peking University, Beijing, China
{wangxiaoyang, y.wang}@pku.edu.cn

Abstract. Lattice theory has close connections with modal logic via algebraic semantics and lattices of modal logics. However, one less explored direction is to view lattices as relational structures based on partial orders, and study the modal logic over them. In this paper, following the earlier steps of Burgess and van Benthem in the 1980s, we use the basic tense logic and its nominal extensions with binary modalities of infimum and supremum to talk about lattices via standard Kripke semantics. As the main results, we obtain a series of complete axiomatizations of lattices, (un)bounded lattices over partial orders or strict preorders. In particular, we solve an axiomatization problem left open by Burgess (1984).

Keywords: tense logic · lattice · completeness · step-by-step · hybrid logic · modal logic · polyadic modal logic

1 Introduction

In the literature, there have been deep and fruitful connections between lattice theory and logic. For example, the algebraic semantics bridges various classical and non-classical logics with their corresponding lattices, starting from the days of Boole [4, 9]; Many-valued logics are often based on lattices of possible truth values [19]; Families of modal logics can form interesting kinds of lattices [8]; Specific logic-induced lattices are also studied in lattice theory [17].

However, there is yet another very natural connection between logic and lattices, which seems to be underdeveloped compared to the above-mentioned directions. Note that there are two equivalent views on lattices: the algebraic one and the order-theoretic one. From the latter, a lattice can be viewed as a partially ordered set where every pair of elements has a unique *supremum* and a unique *infimum*. This may ring a bell to modal logicians who are keen on capturing the properties of relational structures. Indeed, variants of modal logic, such as tense logic, have been successfully applied to capturing temporal structures based on posets [18]. Now, what is the tense logic over lattices as relational structures?

Not much has been done along this line, except the pioneering work by Burgess [10], where besides demonstrating the now well-known step-by-step method, Burgess also obtained a complete tense logic over strict preorders with upper and lower bounds for each pair of elements (not necessarily supremums

and infimums), but without greatest or minimal elements. Burgess also claimed the same logic is complete for unbounded lattices over strict preorders, but left the tense logic over *arbitrary* lattices over strict preorders open.

Note that studying modal logic over lattices is not merely for mathematical interests. In fact, lattices are drawing more and more attention in philosophical discussions, e.g., the (*exact*) *truthmakers* are assumed to have some structures as (semi-)lattices [11], and distributive lattices play an important role in mereology and the structure of sets [14, 16]. Moreover, the Medvedev frames, which can be seen as the special finite meet semilattices, can be captured by several modal languages [15]. More generally, lattices can be used to organize information [2], described by modal and non-classical logics. Moreover, the two tense modalities are viewed as adjoint operators in algebraic logic with lattice structures (cf. e.g., [20]).

In this paper, we continue this line of work on modal logic over lattices. After laying out the basics in Sect. 2, we first give, in Sects. 3 and 4, various complete axiomatizations of tense logics over lattices with or without bounds, over strict or partial orders. In particular, we answer a question left open in [10] by giving a complete axiomatization of arbitrary lattices over strict preorders. Moreover, in Sect. 5, by using the binary modal operators $\langle \text{sup} \rangle$ and $\langle \text{inf} \rangle$ proposed by van Benthem intending to capture the supremums and infimums of information states [1–3], and the nominals in tense logic [7, 15], we give a complete axiomatization of lattices with a powerful extension of the tense logic. We also discovered that without the nominals, the tense logic with $\langle \text{sup} \rangle$ and $\langle \text{inf} \rangle$ is still not capable of defining supremums and infimums precisely.

2 Preliminaries

2.1 Tense Logic

First, let us recall the basics of tense logic [6]. Given a set of proposition letters $\mathbf{P}$, the language of basic tense logic **TL** is defined by the following BNF grammar:

$$\varphi ::= p \in \mathbf{P} \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid G\varphi \mid H\varphi$$

In the temporal context, $G\varphi$ reads φ *is always going to be the case in the future*, and $H\varphi$ says φ *has always been the case in the past*. We define $\perp, \vee, \rightarrow$ as usual, and denote the dual of G as F and the dual of H as P . $F\varphi$ means φ will hold at *some* point in the future and $P\varphi$ says that φ held at *some* point in the past.

TL is interpreted on Kripke models $\mathcal{M} = (W, R, V)$ where W is a non-empty set of possible worlds; R is a binary relation over W , and $V : \mathbf{P} \rightarrow \mathcal{P}(W)$ is the valuation function. The Kripke semantics is defined as usual with the following clauses for the modalities:

$\mathcal{M}, s \models G\varphi \iff$	for all t such that sRt we have $\mathcal{M}, t \models \varphi$
$\mathcal{M}, s \models H\varphi \iff$	for all t such that tRs we have $\mathcal{M}, t \models \varphi$

The *normal tense logics* are defined as extensions of the following logic $\mathbb{K}_t$:

Logic $\mathbb{K}_t$ **Axioms**

TAUT Propositional tautologies

Sym $(p \rightarrow \text{GP}p) \wedge (p \rightarrow \text{HF}p)$ $\text{K}_G \quad G(p \rightarrow q) \rightarrow (Gp \rightarrow Gq)$ $\text{K}_H \quad H(p \rightarrow q) \rightarrow (Hp \rightarrow Hq)$ **Rules:**

MP	$\frac{\psi, \psi \rightarrow \varphi}{\varphi}$	NEC_G	$\frac{\vdash \varphi}{\vdash G\varphi}$	NEC_H	$\frac{\vdash \varphi}{\vdash H\varphi}$	$\text{USUB} \quad \frac{\vdash \varphi(p)}{\vdash \varphi[\psi/p]}$

The completeness can be proved based on the canonical model.

Definition 1. The canonical model for a normal tense modal logic Λ is the structure $\mathcal{M}_\Lambda^c = (W^c, R^c, V^c)$ where:

1. W^c is the set of all Λ -maximal consistent sets (Λ -MCSs);
2. for all $w, u \in W^c$, wR^cu iff for all formulas φ , $G\varphi \in w$ implies $\varphi \in u$;
3. The canonical valuation $V^c(p) = \{w \in W^c \mid p \in w\}$.

wR^cu can be equivalently defined by: for all φ , $H\varphi \in u$ implies $\varphi \in w$.

We can then show the following truth lemma by a routine proof [6]:

Lemma 1 (Truth lemma). For all $w \in W^c$ and $\varphi \in \mathbf{TL}$, $\varphi \in w \iff \mathcal{M}_\Lambda^c, w \models \varphi$.

2.2 Step-by-Step Method

In [10], Burgess uses the step-by-step method to show the completeness of tense logics over various classes of structures. Here, we review the basic ideas following the exposition in [6]. The method constructs a model step by step by using the information in the canonical model, where the maximal consistent sets are taken as building blocks to be attached to the worlds in the model under construction. Some good properties are preserved under each step, and all the undesired defects can be eventually fixed at the limit of the finitary constructions, i.e., the union of all the “networks” constructed monotonically in each step.

Definition 2 (Network). A network for a tense logic Λ is a triple $\mathcal{N} = (N, R, \mu)$ such that N is non-empty set, R is a binary relation over N , and μ is a labeling function mapping each point in N to a Λ -MCS.

Definition 3 (Coherence and Saturation). A network $\mathcal{N} = (N, R, \mu)$ for Λ is coherent if it satisfies:

- $\mu(s)R^c\mu(t)$ for all $s, t \in N$ such that sRt , where R^c is the canonical relation in the canonical model for Λ .

A network $\mathcal{N} = (N, R, \mu)$ for Λ is saturated if:

- (F): if $F\varphi \in \mu(s)$ for some $s \in N$, then there exists $t \in N$ s.t. sRt and $\varphi \in \mu(t)$,
- (P): if $P\varphi \in \mu(s)$ for some $s \in N$, then there exists $t \in N$ s.t. tRs and $\varphi \in \mu(t)$.

A network is perfect if it is both coherent and saturated.

Definition 4. Let $\mathcal{N} = (N, R, \mu)$ be a network for Λ . $\mathcal{F}_{\mathcal{N}}$ is the underlying frame of $\mathcal{N}$. The induced valuation $V_{\mathcal{N}}$ on $\mathcal{F}_{\mathcal{N}}$ is defined by $V_{\mathcal{N}}(p) = \{s \in N \mid p \in \mu(s)\}$. The structure $\mathcal{M}_{\mathcal{N}} = (\mathcal{F}_{\mathcal{N}}, V_{\mathcal{N}})$ is the induced model.

It is a routine proof to show the following [6]:

Lemma 2 (Truth lemma of induced model). If Λ is a normal tense logic, let $\mathcal{N}$ be a perfect network for Λ . Then for all formulas φ , and all nodes s in N , $\mathcal{M}_{\mathcal{N}}, s \models \varphi$ iff $\varphi \in \mu(s)$.

We usually fix a countably infinite set $X = \{x_i \mid i \in \omega\}$ and first build a singleton coherent network. Then we extend the network by adding more and more fresh elements in X . We need to show that every step preserves coherence and eventually, the union of all the intermediate networks is a perfect network.

2.3 Lattices

As we mentioned, lattices can be defined as posets with special properties, or algebraic structures with two binary operators $\wedge, \vee$. To make a difference between the logical connectives and algebraic operations of *meet* and *join*, in the sequel, we use $\mathbb{A}$ and $\mathbb{V}$ to denote the latter. We briefly review the formal definitions.

Definition 5 (Lattice [5]). A relational structure $\mathcal{L}_{\leq} = (L, \leq)$ is called a lattice iff it satisfies the following axioms:

$$\begin{aligned} \text{FORef} : \quad & \forall x (x \leq x) & \text{FOASym} : \quad & \forall x \forall y (x \leq y \wedge y \leq x \rightarrow x = y) \\ \text{FOTrans} : \quad & \forall x \forall y \forall z (x \leq y \wedge y \leq z \rightarrow x \leq z) \\ \text{FOSup} : \quad & \forall x \forall y \exists t (x \leq t \wedge y \leq t \wedge \forall z (x \leq z \wedge y \leq z \rightarrow t \leq z)) \\ \text{FOInf} : \quad & \forall x \forall y \exists t (t \leq x \wedge t \leq y \wedge \forall z (z \leq x \wedge z \leq y \rightarrow z \leq t)) \end{aligned}$$

Axioms **FORef**, **FOTrans**, and **FOASym** say that $\leq$ is a partial order (reflexive, transitive and anti-symmetric), and axioms **FOSup** and **FOInf** make sure any two elements have a least upper bound (*supremum*) and a greatest lower bound (*infimum*). It is not hard to check that the supremum and the infimum of two elements are unique given the anti-symmetry property of $\leq$, so we can define:

$$x \mathbb{A} y := \text{the infimum of } \{x, y\} \quad x \mathbb{V} y := \text{the supremum of } \{x, y\}$$

Following [10], we will also consider tense logic over lattices w.r.t. strict orders. Note that in first-order logic, $\leq$ can be defined by the strict order $<$ and equality, thus we can define lattices w.r.t. $<$ as follows:

Definition 6 (So-lattice [10]). A relational structure $\mathcal{L}_{<} = (L, <)$ is called an so-lattice iff it satisfies the axioms **FOIrr** : $\forall x \neg(x < x)$, **FOTrans** : $\forall x (x < y \wedge y < z \rightarrow x < z)$, and **FOSup** and **FOInf** (where $x \leq y$ is viewed as abbreviation $x < y \vee x = y$).

In [10], instead of **FOIrr**, strict antisymmetry **FOSASym** : $\forall x \forall y \neg(x < y \wedge y < x)$ is used.¹ Under the presence of **FOTrans**, **FOSASym** is equivalent to **FOIrr**. These

¹ Burgess calls the strict order satisfying **FOASym** and **FOTrans** *partial order*.

four axioms can also guarantee the supremum and infimum are unique, thus we can define $\vee$ and $\wedge$ as before.

Finally, let us recall the algebraic definition of lattices.

Definition 7 ([5]). *An algebraic structure $\mathcal{L} = (L, \wedge, \vee)$ is called a lattice iff it satisfies the following axioms:*

1. $\forall x((x \vee x = x) \wedge (x \wedge x = x))$
2. $\forall x \forall y \forall z((x \wedge y) \wedge z = x \wedge (y \wedge z)) \wedge ((x \vee y) \vee z = x \vee (y \vee z))$
3. $\forall x \forall y(x \wedge y = y \wedge x) \wedge (x \vee y = y \vee x)$
4. $\forall x \forall y((x \wedge y) \vee y = y) \wedge ((x \vee y) \wedge y = y)$

Form this algebraic definition, we can recover the partial order by defining $x \leq y :=(x \wedge y = x)$. These three definitions are equivalent to each other [5].

3 TL over Lattices w.r.t. $<$

In this section, we give several complete axiomatizations of tense logic over lattices and related structures. Let us start with the existing result in [10], where instead of lattices, Burgess first considered some unbounded *lattice-like structures* w.r.t. strict preorder $<$ such that every two elements have upper and lower bounds.² Consider the following axioms:

$$\begin{aligned} 4_G : Gp \rightarrow GGp \quad CR_G : FGp \rightarrow GFp \quad CR_H : PHp \rightarrow HPp. \\ D_G : Gp \rightarrow Fp \quad D_H : Hp \rightarrow Pp \end{aligned}$$

Axiom 4_G captures the transitivity of the relation in the model, axioms D_H and D_G capture the seriality of the relation, and axioms CR_G and CR_H capture the Church-Rosser properties (see below). In fact, they are canonical for their *first-order correspondents* (FOCs) since they are Sahlqvist formulas [6]:³

- Proposition 1.** 1. 4_G is canonical for its FOC $\forall x \forall y \forall z(xRy \wedge yRz \rightarrow xRz)$;
 2. CR_G is canonical for its FOC $\forall x \forall y \forall z(xRy \wedge xRz \rightarrow \exists t(yRt \wedge zRt))$;
 3. CR_H is canonical for its FOC $\forall x \forall y \forall z(yRx \wedge zRx \rightarrow \exists t(tRy \wedge tRz))$.
 4. D_G and D_H are canonical for their FOCs $\forall x \exists y xRy$ and $\forall x \exists y yRx$ respectively.

The two Church-Rosser properties for R and its converse actually express if two elements have a lower bound, then they have an upper bound and vice versa.

Now, let $USL : = K_t + 4_G + CR_G + CR_H + D_G + D_H$. To handle the irreflexivity of the strict order and the existence of upper and lower bounds for *any* two elements, Burgess used a step-by-step construction to show the completeness of USL .

Theorem 1 ([10]). *USL is sound and strongly complete for unbounded lattice-like structures. It is also sound and strongly complete for unbounded so-lattices.*

² Here “unbounded” means there are no greatest and least elements.

³ Note that the canonicity is based on the canonical model of normal tense logics.

Note that the second part of the theorem is only claimed without proof in [10, pp. 113], based on the same step-by-step construction for the first part, which builds an unbounded structure. It also means that the language **TL** cannot tell the existence of upper and lower bounds from the existence of the (unique) supremum and infimum for any pair of elements in the unbounded case.

In the rest of this paper, we will follow Burgess' steps below but consider lattices w.r.t. both $<$ and $\leq$ with or without bounds.

3.1 Arbitrary So-Lattices

Let $\mathcal{L}_<$ be the class of *arbitrary* so-lattices (cf. Definition 6). One may think the tense logic over $\mathcal{L}_<$ is simply **USL** without the axioms D_G and D_H . However, this is *not* the case. Note that CR_G and CR_H are no longer valid over $\mathcal{L}_<$ since the first-order properties that they define imply the so-lattice is unbounded, unless it is a singleton lattice.⁴ In [10, pp. 113], Burgess conjectured that a complete tense logic over $\mathcal{L}_<$ can be obtained by weakening the axioms of CR_G and CR_H but left the problem open.⁵ We will first solve this problem in this section.

We need to replace CR_G and CR_H by the following two axioms capturing the possible cases of the upper and lower bounds.

$$\begin{aligned} WCR_G : Fp \wedge Fq &\rightarrow F((Pp \wedge Pq) \vee (p \wedge Pq) \vee (Pp \wedge q) \vee (p \wedge q)) \\ WCR_H : Pp \wedge Pq &\rightarrow P((Fp \wedge Fq) \vee (p \wedge Fq) \vee (Fp \wedge q) \vee (p \wedge q)) \end{aligned}$$

Note that WCR_G and WCR_H are also Sahlqvist formulas, thus they are canonical for their FOCs (for normal tense logics).

Proposition 2. *WCR_G is canonical for its FOC $\forall x \forall y \forall z (xRy \wedge xRz \rightarrow (\exists t (xRt \wedge yRt \wedge zRt) \vee yRz \vee zRy \vee y = z))$ and WCR_H is canonical for its FOC $\forall x \forall y \forall z (yRx \wedge zRx \rightarrow (\exists t (tRx \wedge tRy \wedge tRz) \vee yRz \vee zRy \vee y = z))$.*

Let $\mathbb{SL} := \mathbb{K}_t + 4_G + WCR_G + WCR_H$, it is clear that $\mathbb{SL}$ is sound over $\mathcal{L}_<$. To show $\mathbb{SL}$ is also strongly complete, we first strengthen the coherence and saturation for networks in Definition 3 with the requirement of being so-lattices.

Definition 8. *An $\mathbb{SL}$ -coherent ($\mathbb{SL}$ -saturated, $\mathbb{SL}$ -perfect) network is a coherent (saturated, perfect) network for $\mathbb{SL}$ that is also a so-lattice.*

Notation. Since we are going to build strict orders, we will simply use the symbol $<$ for the relation R in the networks in this section for better readability.

Definition 9. *Let $\mathcal{N} = (N, <, \mu)$ be a network for $\mathbb{SL}$. An F -defect in $\mathcal{N}$ is a pair $\langle s, F\varphi \rangle$ where $s \in N$ and $F\varphi \in \mu(s)$ such that there is no t in N satisfying $s < t$ and $\varphi \in \mu(t)$. Similarly, a P -defect in $\mathcal{N}$ is a pair $\langle s, P\varphi \rangle$ such that there is no t in N satisfying $t < s$ and $\varphi \in \mu(t)$.*

⁴ E.g., by Proposition 1 (2), a frame validating CR_G also has the property $\forall x \forall y (xRy \rightarrow \exists t (yRt))$ (obtained by taking $y = z$ in the original corresponding first-order property).

⁵ Two formulas $Fp \wedge Fq \rightarrow F(Pp \wedge Pq)$, $Pp \wedge Pq \rightarrow P(Fp \wedge Fq)$ are mentioned in [10] as alternatives for CR_G and CR_H , but they are also *not* valid over $\mathcal{L}_<$.

To fix all the defects to obtain a perfect network, we extend the networks step by step by only adding new nodes and relations relevant to them such that the structure of the previous network is preserved entirely in the new network. The key lemma states that any defect of a finite coherent network can be repaired.

Lemma 3 (Repair Lemma for $\mathbb{SL}$). *For any defect of a finite, $\mathbb{SL}$ -coherent network $\mathcal{N}$ there is a finite, $\mathbb{SL}$ -coherent $\mathcal{N}'$ lacking this defect and $\mathcal{N}'$ extends $\mathcal{N}$.*

PROOF. Let $\mathcal{N}=(N, <, \mu)$ be a finite, $\mathbb{SL}$ -coherent network with a defect, we need to show how to fix it by extending it.

We only treat the F-defects, as the case for P-defects follows by symmetry. Assume that there is a point $s \in N$ and a formula $F\varphi$ in $\mu(s)$ for which there is no t in N such that $s < t$ and $\varphi \in \mu(t)$. In the following, let $w_1 = \mu(s) \in \mathcal{M}_{\mathbb{SL}}^c$.

By Lemma 1, since $F\varphi \in w_1$, there is a $w_2 \in \mathcal{M}_{\mathbb{SL}}^c$ such that $w_1 R^c w_2$. So we can pick a fresh node $y \notin N$, label it with $w_2 \in \mathcal{M}_{\mathbb{SL}}^c$ and connect s to y . However, it is not that simple. We first illustrate the ideas before the formal definition of our extended network $\mathcal{N}' = (N', <', \mu')$. Note that we should guarantee $\mathcal{N}'$ is indeed an so-lattice. According to Definition 6, that is to show in $\mathcal{N}'$:

- (1) $<'$ is irreflexive; (2) $<'$ is transitive;
- (3) the supremum of any x, x' exists; (4) the infimum of any x, x' exists.

(1) is not hard to show if we only add fresh nodes without reflexive arrows. For (2), we can add $\{(v, y) \mid v < s\}$ to $<'$. We can do that and preserve coherence (as in Definition 3) because of Proposition 1 (1). As for (3) and (4), we use Proposition 2. Since $(N, <)$ is a *finite* so-lattice, it has a greatest element u . Let $\mu(s)=w_1, \mu(y)=w_2, \mu(u)=w_3$ where s could be u as well. Note that $w_1 R^c w_3$ (when $s \neq u$) for $\mathcal{N}$ is coherent, and $w_1 R^c w_2$ by our selection of w_2 . When $s \neq u$, consider the possible relation between w_2 and w_3 in $\mathcal{M}^c$, we illustrate the intuitive idea first:

- $w_2 = w_3$: it is impossible because it means $\langle s, F\varphi \rangle$ is not an F-defects in $\mathcal{N}$.
- $w_3 R^c w_2$: we add $\{u, y\}$ to $<'$ and the greatest element in $\mathcal{N}'$ turns to y .
- $w_2 R^c w_3$ and *not* $w_3 R^c w_2$ ⁶ : we add $\{(s, y), (y, u)\}$ to $<'$ and the greatest element in $\mathcal{N}'$ is still u .
- w_2 and w_3 are incomparable: By Proposition 2, we have a w_4 in W^c such that $w_2 R^c w_4$ and $w_3 R^c w_4$. Note that $w_4 \neq w_2$ and $w_4 \neq w_3$ for w_2 and w_3 are incomparable. We add a *fresh* $z \notin (N \cup \{y\})$ to N' and let $\mu'(z) = w_4$.

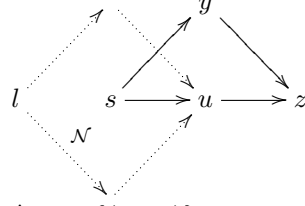
When $s = u$, we have $w_1 = w_3$, $w_1 R^c w_2$ and thus $w_3 R^c w_2$. We add $\{u, y\}$ to $<'$ as in the second case above.

Formally, we define $\mathcal{N}' = (N', <', \mu')$ by three cases:

⁶ This condition is to break the potential symmetry and reflexivity in $\mathcal{M}_{\mathbb{SL}}^c$.

(i) w_2 and w_3 are incomparable (thus $s \neq u$):

- $N' = N \cup \{y, z\}$.
- $<' = < \cup \{(s, y), (y, z)\} \cup \{(v, y) \mid v < s\} \cup \{(v, z) \mid v \in N\}$.
- $\mu' = \mu \cup \{(y, w_2), (z, w_4)\}$.



(ii) $w_3 R^c w_2$:

- $N' = N \cup \{y\}$, $<' = < \cup \{(v, y) \mid v \in N\}$, $\mu' = \mu \cup \{(y, w_2)\}$.

(iii) $w_2 R^c w_3$ and *not* $w_3 R^c w_2$ (thus $s \neq u$):

- $N' = N \cup \{y\}$, $<' = < \cup \{(s, y), (y, u)\} \cup \{(v, y) \mid v < s\}$, $\mu' = \mu \cup \{(y, w_2)\}$.

Now we prove that $\mathcal{N}'$ is a finite so-lattice that is coherent, if $\mathcal{N}$ is a finite, coherent network. Here we check (1)–(4) above for so-lattices:

(1) Irreflexivity is obvious by definition.

(2) For all $x, x', x'' \in N'$, if $x <' x', x' <' x''$, we show $x <' x''$. Note that cases (ii) and (iii) are easy to verify, we only consider case (i) (cf. the graph above).

- $x, x', x'' \in N$: Since $<$ is transitive, we have $x <' x''$ for $< \subseteq <'$.
- $x'' = z$: Since z is the greatest, thus $x <' z$ (note that x cannot be z).
- $x'' = y$: By (i), $x' \leq s$. Since $x <' x'$, $x < s$. Thus $x <' y$ by (i).
- x'' is neither y nor z : By (i), $x, x' \in N$ which reduces to the first item.

(3) We show supremum of any $x, x' \in N'$ always exists in $\mathcal{N}'$. Case (ii) is obvious since only a new greatest element is added. Consider (i) and (iii) below:

- We first show the existing supremums of the “old” elements in N do not change in $(N', <')$. Given any $x, x' \in N$, let $x'' = x \vee x'$ in so-lattice $(N, <)$, we show $x \vee x' = x''$ in $(N', <')$. Since $< \subseteq <'$, we only show for any $r \in N'$, $x \leq' r, x' \leq' r$ in $(N', <')$ implies $x'' \leq' r$ in $(N', <')$, where $\leq'$ is defined by $<'$ and $=$, similar for $\leq$ below.

- If $r \in N$ then we have $x'' \leq r$ thus $x'' \leq' r$.
- If $r = y$ then $x, x' \leq' y$ implies $x, x' \leq s$. Since $x \vee x' = x''$ in $\mathcal{N}$, we have $x'' \leq s$. By the definition of $<'$ we have $x'' \leq' y = r$.
- If $r = z$, we have $x'' \leq' z = r$ by the definition of $<'$.

Hence $x \vee x' = x''$ in $(N', <')$ (it is also true in case (ii)). Below we show the supremum exists for pairs involving new objects.

- Consider (i), we can verify that $z \vee x = z$ for any $x \in N'$; $y \vee x = y$ if $x \leq s$, and otherwise $y \vee x = z$ for $x \not\leq s$. Of course $y \vee y = y$.
- Consider (iii), we can verify that $y \vee x = y$ if $x \leq s$, and otherwise $y \vee x = u$ for $x \not\leq s$. Of course $y \vee y = y$.

It means that the infimum of any $x, x' \in N'$ always exists.

(4) We show the infimum of any $x, x' \in N'$ always exists. Again case (ii) is obvious. We only consider (i) and (iii) below.

- We can show the infimums of elements in N do not change in $(N', <')$ (also true for (ii)). To see it, note that (i) only adds extra elements beyond the previously greatest u . For case (iii), in $(N', <')$, y cannot be the infimum of any two elements in N since it is right below the greatest u . We only need to find an infimum for each pair with new elements.

- Consider (i): $x \wedge y$ is $x \wedge s$ in N for any $x \in N$. $x \wedge z = x$ for any $x \in N'$.
- Consider (iii): $x \wedge y$ is $x \wedge s$ in N for any $x \in N$, and $y \wedge y = y$.

In sum, we have shown that $(N', <')$ is a so-lattice. Since $(N, <)$ is finite and we add at most two points to $(N, <)$, we have $(N', <')$ is again a finite so-lattice.

Next, we will show that $\mathcal{N}'$ is coherent. Again, (ii) is the easy case based on the fact that R^c is transitive due to the canonicity of $\mathbf{4}_G$. We only consider (i) and (iii) below. For all $x, x' \in N'$ such that $x <' x'$, we need to show $\mu'(x)R^c\mu'(x')$.

- $x, x' \in N$: Note that $\mu'(x) = \mu(x)$ and $\mu'(x') = \mu(x')$ by definition of $\mathcal{N}'$. Now $\mu'(x)R^c\mu'(x')$ because $x <' x'$ implies $x < x'$ and $\mathcal{N}$ is coherent.
- $x \in N, x' = y$: By existence lemma of $\mathcal{M}_{\mathbb{SL}}^c$, we have $\mu'(s)R^c\mu'(y)$. By the definition of $<'$ in (i) and (iii), $x <' y$ implies $x < s$ or $x = s$. In the first case $\mu'(x)R^c\mu'(s)$, thus $\mu'(x)R^c\mu'(y)$ by the transitivity of R^c . When $x = s$ we have $\mu'(x)R^c\mu'(y)$ by the selection of $\mu'(y)$.
- $x \in N, x' = z$: It must be the case (i) such that $\mu'(u)R^c\mu'(z)$ by definition. Note that $x \leq u$ because u is the greatest element in $(N, <)$. If $x < u$ we have $\mu'(x)R^c\mu'(u)$, thus $\mu'(x)R^c\mu'(z)$ by transitivity of R^c . If $x = u$ we have $\mu'(x)R^c\mu'(z)$ by the selection of $\mu'(z)$.
- $x = y, x' = z$: $\mu'(y)R^c\mu'(z)$ by definition of case (i).

Hence $\mathcal{N}'$ is a finite, $\mathbb{SL}$ -coherent network extending $\mathcal{N}$ without that defect. $\square$

The construction in the above proof is inspired by Burgess' sketched construction in [10, pp. 112]. Note that Burgess's (simpler) construction always adds a new greatest/least element for each repairing step, by using $\mathbf{CR}_G$ or $\mathbf{CR}_H$, and then the final perfect network must be unbounded. In contrast, our construction, based on a careful case distinction, makes use of the weaker axioms $\mathbf{WCR}_G$ and $\mathbf{WCR}_H$ valid without assuming unboundedness. Note that, it may result in a *bounded* lattice, e.g., the construction (iii) does not add a new greatest element. As we have seen, the details of such a construction are quite involved.

With both the Truth Lemma 2 and the Repair Lemma 3 at our disposal, we can prove the desired strong completeness result. We start with a singleton network and extend it step by step to larger but finite networks by repeatedly using the (construction of) Repair Lemma. We obtain the required perfect network by taking the union of all these networks.

Theorem 2. $\mathbb{SL}$ is strongly complete for the class of so-lattices $\mathcal{L}_<$

PROOF. Fix a countably infinite set $X = \{x_i \mid i \in \omega\}$ as the building blocks of the networks, and enumerate the countable set of potential defects $X \times \{\mathbf{F}, \mathbf{P}\} \times \mathbf{TL}$. Given a consistent set of formulas Σ , expand it to an MCS Σ_0 by a usual Lindenbaum argument. Let $\mathcal{N}_0$ be the network $(\{x_0\}, \emptyset, \{(x_0, \Sigma_0)\})$. Trivially, $\mathcal{N}_0$ is a finite, $\mathbb{SL}$ -coherent network.

Let $0 \leq k$ and suppose $\mathcal{N}_k$ is a finite, $\mathbb{SL}$ -coherent network. Let D be the defect of $\mathcal{N}_k$ that is minimal in our enumeration. If such a D does not exist,⁷

⁷ Note that this can indeed happen, e.g., when $G\perp \wedge H\perp \in \Sigma_0$ thus no $\mathbf{F}\varphi$ or $\mathbf{P}\varphi$ is in Σ_0 .

then by Definition 8 and 3, $\mathcal{N}_k$ is already $\mathbb{SL}$ -perfect and we can set $\mathcal{N}_{k+1} = \mathcal{N}_k$. Otherwise, we extend $\mathcal{N}_k$ by repairing the defect D as described in the proof of the Repair Lemma to obtain $\mathcal{N}_{k+1}$, by using the minimal fresh nodes in $X \setminus \mathcal{N}_k$. Observe that D will not be a defect of any network extending $\mathcal{N}_k$. Finally, let

$$\mathcal{N} = (N, <, \mu) = \left(\bigcup_{k \in \omega} N_k, \bigcup_{k \in \omega} <_k, \bigcup_{k \in \omega} \mu_k \right).$$

Since $\mathbb{SL}$ -coherence is preserved by each step, we just need to check $\mathcal{N}$ is saturated and in fact a so-lattice. If $(N, <) = (N_k, <_k)$ for some $k \in \omega$, then by $\mathbb{SL}$ -coherence, $(N, <)$ is a so-lattice; if not, we should verify that the union of $(N_k, <_k)$ is a so-lattice. Now we need to check the four properties in Definition 6. **FOIrr** trivially holds since each $\mathcal{N}_k$ is a so-lattice. **FOTrans** also holds since any three elements in $\mathcal{N}$ will be contained in some $\mathcal{N}_k$ which is a so-lattice satisfying **FOTrans**. Now for **FOSup** and **FOInf** where $\leq$ is defined by $<$ and $=$, consider any $x, y \in N$, there is a large enough $k \in \omega$ such that $x, y \in N_k$ and hence $z = x \vee y, t = x \wedge y$ are in the so-lattice N_k . From the proof of Lemma 3, for any $j \geq k$, $z = x \vee y, t = x \wedge y$ still hold in the larger network N_j since we never alter the constructed supremums and infimums. Hence z and t are the supremum and infimum respectively for x and y in $(N, <)$. In sum, $(N, <)$ is a so-lattice.

Now towards a contradiction, suppose that $\mathcal{N}$ is not perfect, and let D be the minimal (according to our enumeration) defect of $\mathcal{N}$, say $D = D_k$. By our construction, there must be an $\mathcal{N}_i$ of $\mathcal{N}$ of which D is also a defect since it is only about a node and a formula. Note that D needs not be the minimal defect in $\mathcal{N}_i$. However, there can be at most k defects that are more “urgent”, so D will be repaired before stage $k + i$ of the construction, contradiction. Therefore $\mathcal{N}$ is a perfect network that is also a so-lattice satisfying Σ_0 at x_0 by Lemma 2. $\square$

Here we summarize the general method for proving the completeness theorem w.r.t. normal tense logic $\mathbb{K}_t\Gamma$ (normal extension of $\mathbb{K}_t$ with axioms in Γ):

1. Extending coherence and saturation with desired structural properties about the variants of lattices in concern;
2. Give a proper method to extend the finite network while fixing some defect to saturation while preserving coherence and the desired lattice structure. This will result in a Repair Lemma for $\mathbb{K}_t\Gamma$;
3. Construct coherent finite networks from a singleton network for a given $\mathbb{K}_t\Gamma$ -MCS Σ_0 step by step by repeated use of the above construction;
4. Prove the final constructed network is a perfect network which is also a desired lattice. This will give us the desired model for Σ_0 .

The constructive steps vary according to the specific $\mathbb{K}_t\Gamma$ as we will see below.

3.2 Bounded So-Lattices

We have seen the axiomatizations of unbounded so-lattices and arbitrary so-lattices. The natural remaining case is the bounded so-lattices, i.e., so-lattices

with a greatest element and a least element, to be axiomatized in this subsection. On top of $\mathbb{SL}$, we need the following extra axioms for the boundedness:

$$\text{Bd}_G : G\perp \vee FG\perp \quad \text{Bd}_H : H\perp \vee PH\perp$$

Intuitively, they express each point is either the greatest (least) or can see the greatest (least) element. As before, by Sahlqvist theorem, we can show:

Proposition 3. Bd_G is canonical for its FOC $\forall x(\forall y(\neg xRy) \vee \exists y(xRy \wedge \forall t(\neg yRt)))$; Bd_H is canonical for its FOC $\forall x(\forall y(\neg yRx) \vee \exists y(yRx \wedge \forall t(\neg tRy)))$.

Proposition 4. Suppose $\mathcal{F} = (W, R)$ validates WCR_G . For all $x, y, z \in W$, If xRy , $xRz, z \neq y$ and y is a G -dead-end (which means $G\perp \in y$), then zRy .

PROOF. From xRy, xRz and Proposition 2 we know one of the following must hold: 1) $y = z$, 2) yRz , 3) zRy , 4) there is a t such that zRt and yRt . 1) is impossible because of the assumption; 2) and 4) implies y is not a dead-end, which also contradicts to the assumption. Hence we have zRy . $\square$

Similarly we can state and prove the H-version of Proposition 4 w.r.t. WCR_H .

Let $\mathbb{BSL} := \mathbb{K}_t + 4_G + \text{WCR}_G + \text{WCR}_H + \text{Bd}_G + \text{Bd}_H$ and let $\mathfrak{L}_<^b$ be the class of bounded so-lattices, i.e., so-lattices with greatest and least elements.

Let $\mathcal{N}_0$ be a bounded network with the greatest element u and the least element l . When constructing the sequence of networks $\mathcal{N}_i$, we keep them as the greatest and least element. Then the union of such a sequence is still a bounded structure. The trick is to fix $x_1, x_2 \in X$ as the greatest and least elements for *all* networks *right from the start*. We give the formal definition of the $\mathbb{BSL}$ -coherent and saturated network:

Definition 10. A $\mathbb{BSL}$ -coherent(saturated, perfect) network is a coherent (saturated, perfect) network for $\mathbb{SL}$ that is also a bounded so-lattice with fixed greatest element x_1 and the least element x_2 .

Theorem 3. $\mathbb{BSL}$ is sound and strongly complete for $\mathfrak{L}_<^b$.

PROOF. Given a consistent set of formulas Σ , expand it to an MCS Σ_0 . Since Bd_G and Bd_H are in Σ_0 that is maximally consistent, there are four cases:

- $FG\perp \in \Sigma_0, PH\perp \in \Sigma_0$; then there are $\Sigma_0 R^c w_u$ such that $G\perp \in w_u$ and $w_l R^c \Sigma_0$ such that $H\perp \in w_l$. Let $\mathcal{N}_0$ be
 - $N_0 = \{x_0, x_1, x_2\}$; $<_0 = \{(x_0, x_1), (x_2, x_0), (x_2, x_1)\}$;
and $\mu_0 = \{(x_0, \Sigma_0), (x_1, w_u), (x_2, w_l)\}$.
- $G\perp \in \Sigma_0, PH\perp \in \Sigma_0$; then there is $w_l R^c \Sigma_0$ such that $H\perp \in w_l$. Let $\mathcal{N}_0$ be
 - $N_0 = \{x_1, x_2\}$; $<_0 = \{(x_2, x_1)\}$; $\mu_0 = \{(x_1, w_u), (x_2, w_l)\}$ where $w_u = \Sigma_0$.
- $FG\perp \in \Sigma_0, H\perp \in \Sigma_0$; then there is $\Sigma_0 R^c w_u$ such that $G\perp \in w_u$. Let $\mathcal{N}_0$ be
 - $N_0 = \{x_1, x_2\}$; $<_0 = \{(x_2, x_1)\}$; $\mu_0 = \{(x_1, w_u), (x_2, w_l)\}$ where $w_l = \Sigma_0$.
- $G\perp \in \Sigma_0, H\perp \in \Sigma_0$, then let $x_1 = x_2$ and let $\mathcal{N}_0 = (\{x_1\}, \emptyset, \{(x_1, \Sigma_0)\})$.

It is easy to check that $\mathcal{N}_0$ is a $\mathbb{BSL}$ -coherent network in each case. Note that in the last case, there cannot be any defect and the construction stops.

As for the Repair Lemma, consider the (F)-defect $\langle s, F\varphi \rangle$ (the (P)-defect is similar). Note that $s \neq x_1$, since $G\perp \in \mu_0(x_1)$. We then add a fresh point y and let $\mu'(y)$ be some w_2 such that $\mu(s)R^c w_2$. Moreover, since $\langle s, F\varphi \rangle$ is a defect, $w_2 \neq w_u$, and we have $\mu(s)R^c w_u$, and w_u is a dead-end in the canonical model. By Propositions 2 and 4, we have $w_2 R^c w_u$. So we can extend the network:

$$- N' = N \cup \{y\}, <' = < \cup \{(s, y), (y, x_1)\} \cup \{(v, y) \mid v < s\}, \mu' = \mu \cup \{(y, w_2)\}.$$

It is like the construction (iii) in the proof of Lemma 3, and coherence can be proved similarly. Now we start with one of the $\mathcal{N}_0$ defined above and then construct $\mathcal{N}$ step by step as in the proof of Theorem 2. Since each $\mathcal{N}_i$ is a bounded so-lattice with the greatest and least elements. It is easy to show that $\mathcal{N}_i$ is a bounded so-lattice and thus a $\mathbb{BSL}$ -perfect network. $\square$

4 TL over Lattices w.r.t. $\leq$

In this section, we give complete axiomatizations of tense logics over standard lattices over partial orders as in Definition 5, with or without bounds. Due to lack of space, we only list the results and sketch the proof strategies that are similar to the ones in the previous section.

Note that for lattices w.r.t. partial orders, we need the axiom T: $Gp \rightarrow p$, which is canonical for its FOC $\forall x xRx$, i.e., reflexivity. Also, we will write the relation in the networks as $\leq$ instead of $<$ in the previous section.

Let $\mathbb{L} = \mathbb{K}_t + 4_G + \mathbf{CR}_G + \mathbf{CR}_H + \mathbf{T}$ and let $\mathfrak{L}_{\leq}$ be the class of lattices as in Definition 5. In contrast with the case for so-lattice, $\mathbf{CR}_G$ and $\mathbf{CR}_H$ are valid on $\mathfrak{L}_{\leq}$.

Theorem 4. $\mathbb{L}$ is sound and strongly complete for $\mathfrak{L}_{\leq}$.

The construction for the Repair Lemma is very similar to the one for Lemma 3 where three cases are distinguished. Note that now $\mathbf{CR}_G$ and $\mathbf{CR}_H$ will replace the weaker axioms $\mathbf{WCR}_G$ and $\mathbf{WCR}_H$ to provide us the proper new greatest and least elements. Another difference is we have to do a reflexive, transitive closure after adding the new nodes to make sure the extended network is a lattice while keeping it coherent. Note that by the construction, we will not have two distinct nodes that can see each other, thus making sure the property of anti-symmetry.

Let $\mathfrak{L}_{\leq}^u$ be the class of unbounded lattices, it is interesting that $\mathbb{L}$ is actually also sound and complete over $\mathfrak{L}_{\leq}^u$, since we can always modify the construction and build an unbounded lattice. In fact, we cannot distinguish arbitrary lattices and unbounded lattices over partial orders, while we can do so over strict orders.

Theorem 5. $\mathbb{L}$ is sound and strongly complete for $\mathfrak{L}_{\leq}^u$.

The proof strategy is straightforward: If we get a bounded perfect network $\mathcal{N}$ in the proof of Theorem 4, suppose u and l are the upper bound and lower

bound respectively. We can simply make infinitely many copies of u and l . More precisely, we add those nodes and relations $\{(u, u_0), (l_0, l)\} \cup \{(u_i, u_{i+1}), (l_{i+1}, l_i) \mid i \in \omega\}$ to $\mathcal{N}$ and take the reflexive, transitive closure. As for the labeling function, let $\mu(u_i) = \mu(u), \mu(l_i) = \mu(l)$ for all $i \in \omega$. The resulting structure is an unbounded perfect network $\mathcal{N}^*$ without altering the truth value of formulas.

Finally, let $\mathcal{L}_{\leq}^b$ be the class of bounded lattices. The axioms Bd_G and Bd_H used to capture the boundedness over so-lattices do not work under the presence of axiom T. Instead, to axiomatize the logic over $\mathcal{L}_{\leq}^b$, we need the McKinsey axioms:

$$\mathbf{M}_G : \text{GF}p \rightarrow \text{FG}p \quad \mathbf{M}_H : \text{HP}p \rightarrow \text{PH}p$$

Let $\mathbb{BL}$ be $\mathbb{K}_t + 4_G + \text{CR}_G + \text{CR}_H + \text{T}_G + \mathbf{M}_G + \mathbf{M}_H$, we have:

Theorem 6. *$\mathbb{BL}$ is sound and strongly complete for $\mathcal{L}_{\leq}^b$.*

It is well known that the axiom 4 and the McKinsey formula together capture the property of atomicity: $\forall x \exists y (xRy \wedge \forall z (yRz \rightarrow y = z))$ [6]. Moreover, they are also canonical together [13]. In our setting, this means that in the canonical model $\mathcal{M}_{\mathbb{BL}}^c$, given an MCS, we can find a successor and a predecessor such that they are “final”. By using this observation, when building networks from a singleton one given an MCS, we can fix a greatest element x_1 and a least element x_2 for all latter coherent networks. The rest of the proof is similar to that of Theorem 3 except we have CR_G and CR_H to help us to connect the newly added points with the fixed x_1 and x_2 while keeping the coherence.

5 Nominal TL with $\langle \text{sup} \rangle$ and $\langle \text{inf} \rangle$

As demonstrated by Theorem 1, the tense language cannot really distinguish the existence of upper bounds/lower bounds and the existence of particular supremum/infimum respectively. To capture the latter, we extend our tense language with the two binary operators $\langle \text{sup} \rangle$ and $\langle \text{inf} \rangle$ introduced by van Benthem [1]:

Definition 11. *The language of tense logic with superemum and infimum $TL\text{SI}$:*

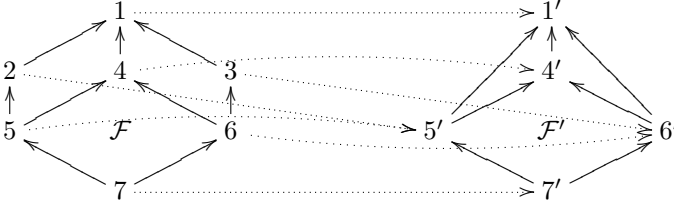
$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid G\varphi \mid H\varphi \mid \langle \text{sup} \rangle(\varphi, \varphi) \mid \langle \text{inf} \rangle(\varphi, \varphi)$$

Intuitively, $\langle \text{sup} \rangle(\varphi, \psi)$ says that the current element is the supremum of some φ element and some ψ element. Similarly for $\langle \text{inf} \rangle(\varphi, \psi)$. Let $\langle \text{ub} \rangle(\varphi, \psi) := (\text{P}\varphi \wedge \text{P}\psi)$ and $\langle \text{lb} \rangle(\varphi, \psi) := (\text{F}\varphi \wedge \text{F}\psi)$ to stand for the upper/lower bound operators.

The model is extended with two ternary relations $\mathcal{M} = (W, R, R_{\text{sup}}, R_{\text{inf}}, V)$ such that R_{sup} and R_{inf} are intended to capture, given further requirements, the ternary relations induced by relating z with x, y if z is the supremum or infimum of x, y respectively. The satisfaction relation is defined below:

$\mathcal{M}, s \models \langle \text{sup} \rangle(\varphi, \psi) \iff$	there exist $t, u \in W$ s.t. $R_{\text{sup}}stu, \mathcal{M}, t \models \varphi, \mathcal{M}, u \models \psi$
$\mathcal{M}, s \models \langle \text{inf} \rangle(\varphi, \psi) \iff$	there exist $t, u \in W$ s.t. $R_{\text{inf}}stu, \mathcal{M}, t \models \varphi, \mathcal{M}, u \models \psi$

Thus for any lattice $(L, \leq)$ with induced operators $\wedge, \vee$, we can define the corresponding induced **TL**SI-frame $(L, \leq, R_{\text{sup}}^{\vee}, R_{\text{inf}}^{\wedge})$ such that $R_{\text{sup}}^{\vee}stu$ iff $s = t \vee u$, and $R_{\text{inf}}^{\wedge}stu$ iff $s = t \wedge u$. Since $\wedge$ and $\vee$ are functions, it follows that $R_{\text{sup}}^{\vee}xyz$ and $R_{\text{sup}}^{\vee}x'yz$ implies $x = x'$. However, such (frame) properties are not definable in **TL**SI. Due to the space limitation, we only illustrate it by sketching the following example of two frames $\mathcal{F}$ and $\mathcal{F}'$, where only the binary partial orders are depicted below. Let the ternary relations be the induced relations in each frame, except that R'_{sup} in $\mathcal{F}'$ is defined as $R'_{\text{sup}} \cup \{(1', 5', 6')\}$ violating the uniqueness of supremum mentioned above. We can build a p-morphism from $\mathcal{F}$ to $\mathcal{F}'$ as the dotted lines show, to preserve the validity of formulas in the language of **TL**SI, thus showing the property is not definable in **TL**SI (cf. [6]).



In order to handle the uniqueness problem of supremum and infimum, we further add nominals as in *nominal tense logic* by Blackburn [7].

Definition 12. Given a countable $\mathbf{P}$ and a countable set of nominals $\mathbf{N}$, the language of nominal tense logic with supremum and infimum (**nTL**SI) is defined as:

$$\varphi ::= p \mid i \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid G\varphi \mid H\varphi \mid \langle \text{sup} \rangle(\varphi, \varphi) \mid \langle \text{inf} \rangle(\varphi, \varphi)$$

Definition 13. A named model for **nTL**SI is $\mathcal{M} = (W, R, R_{\text{sup}}, R_{\text{inf}}, V, \lambda)$ where $\lambda : \mathbf{N} \rightarrow W$ is a valuation of the nominals in $\mathbf{N}$. A named model is surjective if λ is surjective. The nominals i are interpreted by λ : $\mathcal{M}, s \models i$ iff $s = \lambda(i)$.

In the following, we will focus on *surjective named models* only. The following proposition shows the validity of an infinite derivation rule to be featured in our proof system, in order to prove completeness (cf. [12]).

Proposition 5. For any surjective named model $\mathcal{M}$ and any $w \in W$, $\mathcal{M}, w \models i \rightarrow \varphi$ for any $i \in \mathbf{N}$ implies $\mathcal{M}, w \models \varphi$.

By using **nTL**SI, we give the system **NL** extending $\mathbb{K}_t$ with the usual axioms and rules for $\langle \text{sup} \rangle$ and $\langle \text{inf} \rangle$ as normal binary operators and the following axioms:

Extra Axioms

$\mathbf{T} : Gp \rightarrow p$	$\text{sup}_G : \langle \text{sup} \rangle(p, q) \rightarrow \langle \text{ub} \rangle(p, q)$
$\mathbf{4}_G : Gp \rightarrow GGp$	$\text{inf}_H : \langle \text{inf} \rangle(p, q) \rightarrow \langle \text{lb} \rangle(p, q)$
$\text{CR}_G : FGp \rightarrow GFp$	$\text{sup}_E : \langle \text{ub} \rangle(p, q) \rightarrow P\langle \text{sup} \rangle(p, q)$
$\text{CR}_H : PHp \rightarrow HPp$	$\text{inf}_E : \langle \text{lb} \rangle(p, q) \rightarrow F\langle \text{inf} \rangle(p, q)$
$\text{NOM} : i \wedge PF(i \wedge p) \rightarrow p$	$\text{sup}_U : k \wedge \langle \text{sup} \rangle(i, j) \rightarrow HG(\langle \text{sup} \rangle(i, j) \rightarrow k)$
$\text{ASym} : i \rightarrow G(Fi \rightarrow i)$	$\text{inf}_U : k \wedge \langle \text{inf} \rangle(i, j) \rightarrow GH(\langle \text{inf} \rangle(i, j) \rightarrow k)$

Extra Rules:

$$\text{COV}_0 \frac{i \rightarrow \varphi, \text{ for all } i \in \mathbf{N}}{\varphi} \quad \text{USUB}_{\mathbf{N}} \frac{\vdash \varphi(p, i)}{\vdash \varphi[\psi/p, j/i]} \quad i, j \in \mathbf{N}, p \in \mathbf{P}.$$

ASym captures anti-symmetry and NOM specifies the uniqueness of names. sup_G says a supremum is an upper bound; sup_E captures the existence of supremum and it is smaller than any upper bound; sup_U captures the uniqueness of supremum. Similarly for inf_H , inf_E and inf_U .

Note that COV_0 is an infinite rule and we can essentially allow the length of a proof to be infinite. Following [12], let $NTh(\Gamma)$ be the *named theory* of $\mathbf{NL}$ derived by Γ , i.e., the closure of Γ plus the axioms under the applications of rules of $\mathbf{NL}$. By a transfinite induction, we can show that *deduction theorem* holds for such named theories: $\psi \in NTh(\Gamma \cup \{\varphi\})$ iff $\varphi \rightarrow \psi \in NTh(\Gamma)$.

We say that Γ is *surjectively consistent* (*s-consistent for short*) if $\perp \notin NTh(\Gamma)$. We say that $\mathbf{NL}$ is *strongly surjectively complete* over lattices if every surjective consistent set is satisfiable in a surjective named model whose frame is induced from a lattice.

To prove the completeness, we first show Lindenbaum-like lemma inspired by Gargov [12] by a transfinite induction.

Lemma 4. *Every s-consistent set Δ can be extended into a maximal consistent named theory with a nominal in it (MCNS for short).*

PROOF. We enumerate the formulas of **nTLSI**: $\varphi_0, \varphi_1, \dots$ and define successively a chain of s-consistent named $\mathbf{NL}$ -theories $T_0, T_1, \dots$ as follows: $T_0 = NTh(\Delta)$. If T_n is defined and s-consistent, then let $T_{n+1} = NTh(T_n \cup \{\varphi_n\})$ if the latter is s-consistent. Otherwise, we have $\varphi_n \rightarrow \perp \in T_n$ by deduction theorem, i.e., $\neg\varphi_n \in T_n$. Since T_n is s-consistent, by COV_0 there exists a nominal $i \in \mathbf{N}$ such that $i \rightarrow \varphi_n$ does not belong to T_n . Then we can put $\neg(i \rightarrow \varphi_n)$ (or equivalently $i \wedge \neg\varphi_n$) to T_n by letting $T_{n+1} = NTh(T_n \cup \{i \wedge \neg\varphi_n\})$. By deduction theorem, we know that T_{n+1} is s-consistent. Take T be the union of the all T_n . T is maximal, since for every formula, either it is in T or its negation is in T by our construction. As for s-consistency, we first show that $T = NTh(T)$. Suppose not. Then there exists ψ such that $i \rightarrow \psi \in T$ for all $i \in \mathbf{N}$ and $\psi \notin T$. Since T is maximal, we have $\neg\psi \in T$. Suppose $\psi = \varphi_j$ and $\neg\psi = \varphi_k$ in the enumeration, w.l.o.g., suppose $j < k$:

- If $T_j \cup \{\psi\}$ is s-consistent, then $\psi \in T_{j+1}$. Form the s-consistency of each T_n , $\neg\psi \notin T_n$ for each n and thus $\neg\psi \notin T$, contradiction.
- If not, then $\neg\psi \in T_j$. By the construction method of T_{j+1} , $\neg(i \rightarrow \psi) \in T_{j+1}$ for some i , contradiction.

If $\perp \in NTh(T)$, by $T = NTh(T)$ we know $\perp \in T$ and thus $\perp \in T_j$ for some $j \in \mathbf{N}$, contradiction. Hence $\perp \notin NTh(T)$, which means T is s-consistent. $\square$

By the construction of T , we can show that each maximal consistent named $\mathbf{NL}$ -theory contains at least one nominal. Suppose $\perp = \varphi_j$ in the enumeration. Clearly $T_j \cup \{\perp\}$ is inconsistent. Hence there is an i such that $T_{j+1} = NTh(T_k \cup \{i \wedge \neg\perp\})$ is s-consistent. It means $i \in T_{j+1} \subseteq T$. Thus each MCNS contains at least one nominal.

Definition 14. The canonical model for NL $\mathcal{M}^c := (W^c, R^c, R_{sup}^c, R_{inf}^c, V^c, \lambda^c)$ where W^c is the set of MCNS, R^c is defined as before; $R_{sup}^c tuv$ iff for all formulas φ, ψ , $\varphi \in u$ and $\psi \in v$ implies $\langle sup \rangle(\varphi, \psi) \in t$; $R_{inf}^c tuv$ iff for all formulas φ, ψ , $\varphi \in u$ and $\psi \in v$ implies $\langle inf \rangle(\varphi, \psi) \in t$; λ^c is the named valuation defined by $\lambda^c(i) = \{w \in W^c \mid i \in w\}$. We use $\mathcal{M}_w^c$ to denote the point-generated submodel of $\mathcal{M}^c$ given an MCNS w .

Note that by definition λ^c assigns to each $i \in \mathbf{N}$ a set, but it is not hard to check in the point-generated $\mathcal{M}_w^c$, $\lambda^c(i)$ is a singleton set given the axioms about nominals (cf. [7]). The existence lemma and truth lemma is standard for $\mathcal{M}_w^c$.

Before showing the completeness, we have some handy observations. First some notion. In a model, we say w_1 and w_2 are *accessible* iff there exists $n \in \omega$ such that $w_1 = x_0 R^* x_2 R^* \dots x_{n-1} R^* x_n = w_2$, where $R^* \in \{R, R^{-1}\}$. The accessible relation is clearly an equivalence relation. Based on Proposition 1, we can show:

Lemma 5. For any frame $\mathcal{F}$ validating 4, CR_G, CR_H and any $w_1, w_2 \in \mathcal{F}$, the following are equivalent:

- w_1 and w_2 are accessible;
- there exists $w \in \mathcal{F}$ such that $w_1 R w$, $w_2 R w$;
- there exists $w \in \mathcal{F}$ such that $w R w_1$, $w R w_2$.

By using this lemma, we have that for all pointed generated frame $\mathcal{F}$ that satisfies 4, CR_G, CR_H, every two points have an upper bound and a lower bound.

Theorem 7. NL is sound and strongly surjectively complete for the class of all TLSI-frames induced by lattices (by taking $\leq$, $R_{sup}^\forall$ and $R_{inf}^\exists$ as the relations).

PROOF (Sketch). For any s-consistent set of formulas Δ , extending to an MCNS w . Consider $\mathcal{M}_w^c$, R^c is clearly reflexive and transitive due to T and 4_G. For anti-symmetry, consider x, x' such that $x R^c x'$, $x' R^c x$. Note that x has some nominal in it, suppose $i \in x$. We know that $i \rightarrow G(Fi \rightarrow i)$, so $G(Fi \rightarrow i) \in x$. By $x R^c x'$ we have $Fi \rightarrow i \in x'$. By $x' R^c x$ and $i \in x$ we have $Fi \in x'$ and thus $i \in x'$. It implies $x = x'$ by NOM. Hence $(\mathcal{M}_w^c, R^c)$ is a partial order. In the following, let R_{ub}^c be the ternary relation defined by $R_{ub}^c tuv$ if $u R^c t$ and $v R^c t$, let $R_{lb}^c tuv$ if $t R^c u$ and $t R^c v$. Now we check the properties FOSup and FOInf essentially about supremum and infimum in Definition 5. Again, we only check FOSup. First, note that sup_G and sup_E are Sahlqvist, thus by canonicity we have (a) $\forall x \forall y \forall z (R_{sup}^c xyz \rightarrow R_{ub}^c xyz)$, and (b) $\forall x \forall y \forall z (R_{ub}^c xyz \rightarrow \exists t (t R x \wedge R_{sup}^c t y z))$.

For all $x, x' \in \mathcal{M}_w^c$, we need to find their supremum to satisfy FOSup. Since x and x' are accessible via w , by Lemma 5 and the canonicity of 4, CR_G, there is y such that $R_{ub}^c y x x'$. By (b) we have a z such that $R_{sup}^c z x x'$ and $z R y$. We claim z is the witness that we need. First note that, by (a), we have $x R^c z$ and $x' R^c z$. We need to show other upper bounds of x, x' are bigger than z . Since the worlds are named in $\mathcal{M}_w^c$, we assume $i \in x, j \in x', k \in z$. Then by truth lemma $k \wedge \langle sup \rangle(i, j) \in z$, By sup_U we have $HG(\langle sup \rangle(i, j) \rightarrow k) \in z$. For all t such that $x R^c t$ and $x' R^c t$, we

have $\langle \text{ub} \rangle(i, j) \in t$. By (b) again there is $t' R^c t$ such that $\langle \text{sup} \rangle(i, j) \in t'$. Since z and t' are accessible (thus have a common lower bound), and $\text{HG}(\langle \text{sup} \rangle(i, j) \rightarrow k) \in z$, we know $\langle \text{sup} \rangle(i, j) \rightarrow k \in t'$, thus $k \in t'$. It implies $t' = z$ and thus $z R^c t$. This means for all $x, x' \in \mathcal{M}_w^c$, there is a (unique) z such that $z = x \vee x'$. Similarly we can prove **FOInf**. Furthermore, it is not hard to show R_{sup}^c and R_{inf}^c are the induced binary relations of R^c based on the uniqueness of supremum and infimum for each pair. Thus $\mathcal{M}_w^c$ is a lattice-induced model and by truth lemma Δ is satisfied at w . $\square$

As for future work, we will study tense and modal logics over variants of lattices as relational structures, such as the distributive lattices and semilattices that are relevant in mereology, truthmaker semantics, and specific fields such as Medvedev's logic. The use of the binary modalities such as $\langle \text{sup} \rangle$ and $\langle \text{inf} \rangle$ can connect polyadic modal logic and various abstract algebras from a new perspective. For example, the corresponding properties in Definition 7 can all be turned into modal axioms, e.g., for symmetry law we can show $\langle \text{sup} \rangle(i, j) \leftrightarrow \langle \text{sup} \rangle(j, i)$ is a theorem in our proof system **NL**. A more interesting example is the modal correspondence of the absorption law with nominals $i \rightarrow \langle \text{sup} \rangle(\langle \text{inf} \rangle(j, i), i)$. We leave the systematic study of such a connection to a future occasion.

Acknowledgement. The authors thank John Burgess, Johan van Benthem, and the two anonymous reviewers of WoLLIC2022 for relevant pointers and suggestions. The research is supported by NSSF grant 19BZX135.

References

1. van Benthem, J.: Semantic parallels in natural language and computation. In: *Studies in Logic and the Foundations of Mathematics*, vol. 129 (1989)
2. van Benthem, J.: Modal logic as a theory of information. In: Copeland, J. (ed.) *Logic and Reality: Essays on the legacy of Arthur Prior*, pp. 135–158. Clarendon Press, Oxford (1996)
3. van Benthem, J.: Implicit and explicit stances in logic. *J. Philos. Log.* **48**(3), 571–601 (2019)
4. Birkhoff, G., Neumann, J.V.: *The Logic of Quantum Mechanics*. Springer, Dordrecht (1975)
5. Birkhoff, G.T.: *Lattice Theory*. American Mathematical Society (1967)
6. Blackburn, P., de Rijke, M., Venema, Y.: *Modal Logic*. Cambridge University Press, Cambridge (2001)
7. Blackburn, P.: Nominal tense logic. *Notre Dame J. Formal Logic* **34**(1), 56–83 (1993)
8. Blok, W.J.: The lattice of modal logics: an algebraic investigation. *J. Symb. Log.* **45**(2), 221–236 (1980)
9. Boole, G.: The mathematical analysis of logic. *Bull. Math. Biophys.* **12**(5), 107–107 (2009)
10. Burgess, J.P.: Basic tense logic. In: Gabbay, D., Guenther, F. (eds.) *Handbook of Philosophical Logic: Volume II: Extensions of Classical Logic*, pp. 89–133. Springer, Dordrecht (1984). https://doi.org/10.1007/978-94-009-6259-0_2
11. Fine, K.: Truth-maker semantics for intuitionistic logic. *J. Philos. Log.* **43**(2–3), 549–577 (2014)

12. Gargov, G., Goranko, V.: Modal logic with names. *J. Philos. Log.* **22**(6), 607–636 (1993)
13. Goldblatt, R., Hodkinson, I.: The McKinsey-Lemmon logic is barely canonical. *Australas. J. Logic* **5** (2007)
14. Hamkins, J.D., Kikuchi, M.: Set-theoretic mereology. *Logic Logical Philos.* Special issue “Mereology and beyond, Part II” **25**(3), 285–308 (2016)
15. Holliday, W.H.: On the modal logic of subset and superset: tense logic over Medvedev frames. *Stud. Logica.* **105**(1), 13–35 (2017)
16. Li, D., Wang, Y.: Mereological bimodal logics. *Rev. Symb. Logic* 1–36 (2022). <https://doi.org/10.1017/S175502032200003X>
17. Medvedev, Y.T.: Degrees of difficulty of the mass problems. *Dokl. Akad. Nauk SSSR* **104**, 501–504 (1955)
18. Prior, A.: *Time and Modality*. Oxford University Press, Clarendon Press (1957)
19. Rose, A.: Systems of logic whose truth-values form lattices. *Math. Ann.* **123**, 152–165 (1951)
20. Sadrzadeh, M., Dyckhoff, R.: Positive logic with adjoint modalities: proof theory, semantics, and reasoning about information. *Rev. Symb. Logic* **3**(3), 351–373 (2010)



Expressing Power of Elementary Quantum Recursion Schemes for Quantum Logarithmic-Time Computability

Tomoyuki Yamakami^(✉)

University of Fukui, 3-9-1 Bunkyo, Fukui 910-8507, Japan
tomoyukiyamakami@gmail.com

Abstract. Quantum computing has been studied over the past four decades based on two computational models of quantum circuits and quantum Turing machines. To capture quantum polynomial-time computability, a new recursion-theoretic approach was taken lately by Yamakami [J. Symb. Logic 80, pp. 1546–1587, 2020] by way of schematic definitions, which constitute a few initial quantum functions and a few construction schemes, including composition, branching, and multi-qubit quantum recursion. By taking a similar step, we look into quantum logarithmic-time computability and further explore the expressing power of elementary schemes designed for such quantum computation. In particular, we introduce an elementary form of the quantum recursion, called the fast quantum recursion, which helps us capture quantum logarithmic-time computability.

Keywords: schematic definition · quantum Turing machine · fast quantum recursion · quantum logarithmic-time computability

1 Background, Motivations, and Challenges

1.1 Schematic Definitions that Capture Quantum Computability

Over the four decades, the study of quantum computing has made a significant progress both in theory and in practice. Quantum computing is one of the most anticipated Nature-inspired computing paradigms today because it relies on the principle of quantum mechanics, which is assumed to govern Nature. The core of quantum computing is an exquisite handling of *superpositions*, which realize massive quantum “parallelism”, of quantum states. Early models of quantum computation were proposed by Benioff [4], Deutsch [5, 6], and Yao [17] as quantum analogues of Turing machines and Boolean circuits. Those models have been used in a theoretical study of quantum computing since their introduction.

A schematic approach has been a recent incentive for the full characterization of the notion of quantum polynomial-time computability [16]. The primary purpose of taking such a schematic approach toward quantum computability stems

from a successful development of recursion theory (or recursive function theory) for classical computability. Earlier, Peano, Herbrand, Gödel, and Kleene [7, 8] contributed to paving a road to a coherent study of computability and decidability from a purely logical aspect (see, e.g., [11] for references therein). In this theory, recursive functions are formulated in the following simple “schematic” way. (i) We begin with a few initial functions. (ii) We sequentially build more complicated functions by applying a set of construction schemes to the already-built functions. The description of how to construct a recursive function actually serves as a blueprint of the function, which resembles like a “computer program” of the modern times in such a way that each construction scheme is a command line of computer programs. Hence, a schematic definition itself may be viewed as a high-level programming language by which we can render a computer program that describes the movement of any recursive function. As the benefit of the use of such a schematic definition, when we express a target function as a series of schemes, the “size” of this series can serve as a “descriptive complexity measure” of the function. The primary purpose of a schematic approach is therefore to study such descriptive complexity of various functions. This schematic approach sharply contrasts the use of Turing machines and Boolean circuits to formulate recursive functions. A similar approach taken in [16] aims at capturing the polynomial-time computability of quantum functions,¹ each of which maps a finite-dimensional Hilbert space to itself, in place of the recursive functions.

Shown in [16] are a precise characterization of quantum polynomial-time computability and a quantum analogue of the normal form theorem. A key in his schematic definition is an introduction of the *multi-qubit quantum recursion scheme*, which is quite different in its formulation from the classical recursion scheme. See Sect. 3.1 for its formal definition. This quantum recursion scheme turns out to be quite powerful so that it helps us capture the notion of quantum polynomial-time computability, which is a quantum extension of deterministic polynomial-time computability.

There are a few advantages of using schematic definitions for quantum computing over other computational models, such as quantum Turing machines (QTMs) and families of quantum circuits. The most significant advantage is in fact that there is no use of the *well-formedness requirement* of QTMs and the *uniformity requirement* of quantum circuit families. These natural but cumbersome requirements are necessary to guarantee the unitary nature of QTMs and the algorithmic generation of quantum circuit families. Schematic definitions, on the contrary, avoid such extra requirements and make it much simpler to design a quantum algorithm given in the form of a quantum function for solving a target combinatorial problem. Remember that the schematic definition in [16] constitutes only a small number of initial quantum functions together with basic construction schemes and this fact helps us render a short procedural sequence of how to construct each target quantum function.

In a similar vein, we wish to turn our attention to a “limited” form of quantum computing, which can be naturally implemented by small space/time-

¹ This notion should be distinguished from the same terminology used in [15].

bounded QTMs as well as families of small size/depth-bounded quantum circuits, and we wish to examine how the schematic approach of [16] copes with such quantum computing.

1.2 Our Challenges in This Work

We intend to extend the scope of the existing research on quantum computability by way of an introduction of a new set of schematic definitions. In particular, we look into limited quantum computability and seek for an appropriate schematic definition to capture such computability in hope that we would promote a much better understanding of quantum computing in general.

The multi-qubit quantum recursion scheme of [16] captures quantum polynomial-time computability. A basic idea of this quantum recursion scheme is, starting with n qubits, to discard the first k qubits, depending on these qubits, and determine whether or not we continue applying the same scheme to the rest of the qubits. This process slowly consumes the entire n qubits and finally grinds to halt within linearly-many recursive steps. To describe limited quantum computability, in contrast, how can we weaken the quantum recursion scheme? Our attempt in this work is to speed up this recursive process significantly by discarding a bundle of $\lceil n/2 \rceil$ (or $\lfloor n/2 \rfloor$) qubits from the n qubits at each stage. Such a way of halving the number of qubits at each recursive stage makes the recursive process terminate significantly faster. In fact, this process ends in at most $\lceil \log n \rceil$ recursive steps. For this very reason, we intend to call this weakened recursive process the *fast quantum recursion scheme*.

Although the fast quantum recursion scheme is significantly weaker in power than the quantum recursion scheme of [16], the scheme still generates numerous interesting and useful quantum functions. We call the set of all quantum functions generated from a few initial quantum functions and a few construction schemes together with the fast quantum recursion scheme by *EQS* (elementary quantum schemes). See Sect. 3 for its precise definition. This set EQS turns out to precisely characterize *quantum logarithmic-time (logtime) computability*.

Due to the page limit, we omit proofs of certain supporting lemmas and even relevant topics. Those omitted parts will be included in the forthcoming complete version.

2 Preparation: Notions and Notation

We will briefly explain basic notions and useful notation used in this work.

2.1 Numbers, Strings, and Languages

The notations $\mathbb{Z}$, $\mathbb{Q}$, and $\mathbb{R}$ denote respectively the sets of all integers, of all rational numbers, and of all real numbers. In particular, we set $\bar{\mathbb{Q}} = \mathbb{Q} \cap [0, 1]$. The *natural numbers* are nonnegative integers and the notation $\mathbb{N}$ expresses the set of those numbers. We further write $\mathbb{N}^+$ to denote $\mathbb{N} - \{0\}$. For two integers

m, n with $m \leq n$, $[m, n]_{\mathbb{Z}}$ denotes the *integer interval* $\{m, m+1, m+2, \dots, n\}$. We abbreviate $[1, n]_{\mathbb{Z}}$ as $[n]$ if n is in $\mathbb{N}^+$. Moreover, $\mathbb{C}$ denotes the set of all complex numbers. We denote $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ for the ceiling and the floor functions, respectively. In this work, all *polynomials* have nonnegative integer coefficients and all *logarithms* are taken to the base 2. For convenience, we set $\log 0 = 0$. The notation i denotes $\sqrt{-1}$ and e is the base of natural logarithms. Given a complex number α , α^* denotes the *complex conjugate* of α . As for a nonempty set of quantum amplitudes, which is a subset of $\mathbb{C}$, we use the notation K .

A nonempty finite set of symbols (or letters) is called an *alphabet* and a sequence of such symbols from a fixed alphabet Σ is called a *string* over Σ . The total number of symbols in a string x is the *length* of x and is denoted $|x|$. The *empty string* λ is a unique string of length 0. A collection of those strings over Σ is a *language* over Σ . Throughout this work, we deal only with *binary strings*, which are sequences of 0s and 1s. Given a number $n \in \mathbb{N}$, Σ^n (resp., $\Sigma^{\leq n}$) denotes the set of all strings of length exactly n (resp., at most n). Let $\Sigma^* = \bigcup_{n \in \mathbb{N}} \Sigma^n$ and $\Sigma^+ = \Sigma^* - \{\lambda\}$.

2.2 Quantum States and Hilbert Spaces

We assume the reader's familiarity with basic quantum information and computation (see e.g., textbooks [9, 10]). A basic concept in quantum computing is a Hilbert space and unitary transformations over it. The *ket notation* $|\phi\rangle$ expresses a (column) vector in a Hilbert space and its transposed conjugate is expressed as a (row) vector of the dual space and is denoted by the *bra notation* $\langle\phi|$. The notation $\langle\psi|\phi\rangle$ denotes the *inner product* of two vectors $|\psi\rangle$ and $|\phi\rangle$. We use I to denote the identity matrix of an arbitrary dimension. A square complex matrix U is called *unitary* if U satisfies $UU^\dagger = U^\dagger U = I$, where $U^\dagger$ is the transposed conjugate of U .

A *quantum bit* (or a *qubit*, for short) is a linear combination of two basis vectors $|0\rangle$ and $|1\rangle$. The notation $\mathcal{H}_2$ indicates the Hilbert space spanned by $|0\rangle$ and $|1\rangle$. More generally, $\mathcal{H}_{2^n}$ refers to the Hilbert space spanned by the computational basis $B_n = \{|s\rangle \mid s \in \{0, 1\}^n\}$. Let $\mathcal{H}_\infty = \bigcup_{n \in \mathbb{N}^+} \mathcal{H}_{2^n}$. Given a quantum state $|\phi\rangle \in \mathcal{H}_\infty$, its *length* $\ell(|\phi\rangle)$ denotes the minimum number $n \in \mathbb{N}$ satisfying that $|\phi\rangle \in \mathcal{H}_{2^n}$. We use the generic notation $\mathbf{0}$ to denote the *null vector* of an arbitrary dimension. As a special case, we set $\ell(\mathbf{0}) = 0$ for the null vector $\mathbf{0}$ (although $\mathbf{0}$ belongs to $\mathcal{H}_k$ for any $k \in \mathbb{N}^+$). For convenience, we also set $\ell(\alpha) = 0$ for any scalar $\alpha \in \mathbb{C}$. For a quantum state $|\phi\rangle$, if we make a measurement in the computational basis, then each binary string x of length n is observed with probability $|\langle x|\phi\rangle|^2$. Thus, $|\phi\rangle$ can be expressed as $|0\rangle\langle 0|\phi\rangle + |1\rangle\langle 1|\phi\rangle$ and also as $\sum_{x \in \{0, 1\}^n} \langle x|\phi\rangle \otimes |x\rangle$, where $|\psi\rangle \otimes |\psi\rangle$ is the *tensor product* of $|\phi\rangle$ and $|\psi\rangle$. The *norm* of $|\phi\rangle$ is defined as $\sqrt{\langle\phi|\phi\rangle}$ and is denoted by $\| |\phi\rangle \|$.

A *qustring of length n* is a unit-norm quantum state in $\mathcal{H}_{2^n}$. As a special case, the null vector is treated as the *qustring of length 0* (although its norm is 0). Let Φ_n denote the set of all qustrings of length n and set Φ_∞ to be $\bigcup_{n \in \mathbb{N}} \Phi_n$.

For convenience, we set $LH(|\phi\rangle) = \lceil \ell(|\phi\rangle)/2 \rceil$ and $RH(|\phi\rangle) = \lfloor \ell(|\phi\rangle)/2 \rfloor$ for any quantum state $|\phi\rangle \in \mathcal{H}_\infty$. It then follows that $\ell(|\phi\rangle) = LH(|\phi\rangle) + RH(|\phi\rangle)$.

2.3 Conventions on the Bra and the Ket Notations

We consider only functions mapping $\mathcal{H}_\infty$ to $\mathcal{H}_\infty$ and these functions are generally referred to as *quantum functions* to differentiate from functions working with natural numbers or strings. To calculate the outcomes of such quantum functions, we take advantage of making a purely symbolic treatment of the bra and the ket notations together with the tensor product notation $\otimes$. We generally follow [16] for the specific usage of these notations. Hereafter, we briefly explain how to use them in this work.

The null vector $\mathbf{0}$ satisfies the following: for any $|\phi\rangle \in \mathcal{H}_\infty$, (i) $0 \otimes |\phi\rangle = |\phi\rangle \otimes 0 = \mathbf{0}$, (ii) $|\phi\rangle \otimes \mathbf{0} = \mathbf{0} \otimes |\phi\rangle = \mathbf{0}$, and (iii) if $|\psi\rangle$ is the null vector, then $\langle\phi|\psi\rangle = \langle\psi|\phi\rangle = 0$. Given two binary strings $u, s \in \{0, 1\}^+$, if $|s| = |u|$, then $\langle u|s\rangle = 1$ if $u = s$, and $\langle u|s\rangle = 0$ otherwise. On the contrary, when $|u| < |s|$, $\langle u|s\rangle$ denotes $|z\rangle$ if $s = uz$ for a certain string z . By contrast, if $|u| > |s|$, then $\langle u|s\rangle$ always denotes $\mathbf{0}$. More generally, if $|\phi\rangle = \sum_{u \in \{0,1\}^+} \sum_{s \in \{0,1\}^+} \alpha_{u,s} |u\rangle |s\rangle$ and $u_0 \in \{0, 1\}^+$, then $\langle u_0|\phi\rangle$ expresses the quantum state $\sum_{s \in \{0,1\}^+} \alpha_{u_0,s} |s\rangle$. For instance, if $|\phi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, then $\langle 0|\phi\rangle$ equals $\frac{1}{\sqrt{2}}|0\rangle$ and $\langle 1|\phi\rangle$ does $\frac{1}{\sqrt{2}}|1\rangle$. In a more general case with $|\xi\rangle = \sum_{u \in \{0,1\}^+} \beta_u |u\rangle$, $\langle \xi|\phi\rangle$ means $\sum_{u \in \{0,1\}^+} \beta_u^* \langle u|\phi\rangle$, which equals $\sum_{u \in \{0,1\}^+} \sum_{s \in \{0,1\}^+} \alpha_{u,s} \beta_u^* |s\rangle$. We often express $|\phi\rangle \otimes |\psi\rangle$ as $|\phi\rangle |\psi\rangle$. In particular, for strings s and u , $|s\rangle \otimes |u\rangle$ is further abbreviated as $|s, u\rangle$ or even $|su\rangle$. When $\langle s|\phi\rangle$ is just a scalar, say, $\alpha \in \mathbb{C}$, the notation $|s\rangle \otimes \langle s|\phi\rangle$ (or equivalently, $|s\rangle \langle s|\phi\rangle$) is equal to $\alpha |s\rangle$. The empty string λ should be treated quite differently. For our convenience, we automatically translate $|\lambda\rangle$ into the null vector $\mathbf{0}$ unless otherwise stated.

We also expand the norm notation $\|\cdot\|$ for vectors in the following way. For any quantum states $|\phi\rangle$ and $|\psi\rangle$ with $\ell(|\phi\rangle) = \ell(|\psi\rangle) > 0$, the notation $\|\langle\phi|\psi\rangle\|$ is used to express the absolute value $|\langle\phi|\psi\rangle|$. In general, for any scalar $\alpha \in \mathbb{C}$, $\|\alpha\|$ denotes $|\alpha|$. This notational convention is quite useful in handling the value obtained after making a measurement without worrying about whether the resulting object is a quantum state or a scalar.

3 A Schematic Definition for EQS

In what follows, we will present a schematic definition to introduce a class of special quantum functions on $\mathcal{H}_\infty$ (i.e., from $\mathcal{H}_\infty$ to $\mathcal{H}_\infty$). A quantum function is said to be *dimension-preserving* (resp., *norm-preserving*) if $\ell(|\phi\rangle) = \ell(f(|\phi\rangle))$ (resp., $\|\langle\phi|\phi\rangle\| = \|\langle f(|\phi\rangle)|f(|\phi\rangle)\|$) holds for any quantum state $|\phi\rangle \in \mathcal{H}_\infty$.

3.1 Elementary Quantum Schemes

As a starter, we discuss a skeleton of our schematic definition for a class of elementary quantum functions, which is later called *EQS* (elementary quantum schemes), involving eight initial quantum functions and three construction schemes.

We fix a nonempty amplitude set K throughout the rest of this work and we often ignore the reference to K .

Definition 1. The class EQS_0 is composed of all quantum functions constructed by selecting the initial quantum functions of Scheme I and then inductively applying Schemes II–III a finite number of times. Let $|\phi\rangle, |\phi_1\rangle, |\phi_2\rangle, \dots, |\phi_k\rangle$ be arbitrary quantum states in $\mathcal{H}_\infty$. If the item 8) of Scheme I is not used, we denote by $\widehat{EQS}_0$ instead of EQS_0 .

I. The initial quantum functions. Let $\theta \in [0, 2\pi) \cap K$ and $a \in \{0, 1\}$.

1) $I(|\phi\rangle) = |\phi\rangle$. (identity)

2) $PHASE_\theta(|\phi\rangle) = |0\rangle\langle 0|\phi\rangle + e^{i\theta}|1\rangle\langle 1|\phi\rangle$. (phase shift)

3) $ROT_\theta(|\phi\rangle) = \cos\theta|\phi\rangle + \sin\theta(|1\rangle\langle 0|\phi\rangle - |0\rangle\langle 1|\phi\rangle)$. (rotation around xy -axis at angle θ)

4) $NOT(|\phi\rangle) = |0\rangle\langle 1|\phi\rangle + |1\rangle\langle 0|\phi\rangle$. (negation)

5) $SWAP(|\phi\rangle) = \begin{cases} |\phi\rangle & \text{if } \ell(|\phi\rangle) \leq 1, \\ \sum_{a,b \in \{0,1\}} |ab\rangle\langle ba|\phi\rangle & \text{otherwise.} \end{cases}$ (swapping of 2 qubits)

6) $REMOVE(|\phi\rangle) = \sum_{s:|s|=1} \langle s|\phi\rangle \otimes |s\rangle$. (removal to the end)

7) $REP(|\phi\rangle) = \sum_{s:|s|=\ell(|\phi\rangle)-1} \langle s|\phi\rangle \otimes |s\rangle$. (replacement from the end)

8) $MEAS[a](|\phi\rangle) = |a\rangle\langle a|\phi\rangle$. (partial projective measurement)

II. The composition scheme. From g and h , we define $Compo[g, h]$ as follows:
 $Compo[g, h](|\phi\rangle) = g \circ h(|\phi\rangle) (= g(h(|\phi\rangle)))$.

III. The branching scheme.² From g and h , we define $Branch[g, h]$ as:

(i) $Branch[g, h](|\phi\rangle) = |\phi\rangle$ if $\ell(|\phi\rangle) \leq 1$,
(ii) $Branch[g, h](|\phi\rangle) = |0\rangle \otimes g(|\phi\rangle) + |1\rangle \otimes h(|\phi\rangle)$ otherwise.

If it is necessary to emphasize K , then we write $EQS_{K,0}$ and $\widehat{EQS}_{K,0}$.

We quickly provide simple examples of how to compute the items of Scheme I. For three bits $a, b, c \in \{0, 1\}$, we obtain $PHASE_\pi(|1\rangle) = -|1\rangle$, $ROT_{\pi/4}(|0\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$, $NOT(|a\rangle) = |1-a\rangle$, $SWAP(|abc\rangle) = |bac\rangle$, $REMOVE(|abc\rangle) = |bca\rangle$, and $REP(|abc\rangle) = |cab\rangle$. Moreover, for Scheme III, we obtain $Branch[I, NOT](ROT_{\pi/4}(|0\rangle|0\rangle) = Branch[I, NOT](\frac{1}{\sqrt{2}}(|00\rangle + |01\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$.

For later reference, let us review from [16] the *multi-qubit quantum recursion scheme*, which is a centerpiece of schematically characterizing quantum polynomial-time computability. In this work, this special scheme is referred to as *Scheme T*.

(T) The multi-qubit quantum recursion scheme [16]. Let g, h, p be quantum functions and let $k, t \in \mathbb{N}^+$ be numbers. Assume that p is dimension-preserving. For $\mathcal{F}_k = \{f_u\}_{u \in \{0,1\}^k}$, we define $kQRec_t[g, h, p|\mathcal{F}_k]$ as follows:

(i) $kQRec_t[g, h, p|\mathcal{F}_k](|\phi\rangle) = g(|\phi\rangle)$ if $\ell(|\phi\rangle) \leq t$,
(ii) $kQRec_t[g, h, p|\mathcal{F}_k](|\phi\rangle) = h(\sum_{u:|u|=k} |u\rangle \otimes f_u(|\psi_{p,\phi}\rangle))$ otherwise,

where $|\psi_{p,\phi}\rangle = p(|\phi\rangle)$ and $\mathcal{F}_k \subseteq \{kQRec_t[g, h, p|\mathcal{F}_k], I\}$.

² We remark that $Branch[g, h]$ can be expressed as an appropriate unitary matrix and thus it is a legitimate quantum operation to consider.

This quantum recursion scheme together with the items 1)–5) and 8) in Scheme I and Schemes II–III given above produces all polynomial-time computable quantum functions [16].

It is important to note that, if we allow Scheme T, then *REMOVE* and *REP* can be generated from the items 1)–5) of Scheme I and Schemes II–III [16] and thus they become redundant. Since we use a limited form of quantum recursion in this work, we intentionally include the items 6)–7) to Scheme I.

For later use, we define another quantum function, which splits the entire input qubits into two halves and then swaps them. This quantum function is not be included in our system but it will be used to support the description of a new scheme given in Sect. 3.2. Remark that, to construct this quantum function, we need Schemes I–III together with the quantum recursion (Scheme T).

$$*) \text{ HalfSWAP}(|\phi\rangle) = \begin{cases} |\phi\rangle & \text{if } \ell(|\phi\rangle) \leq 1, \\ \sum_{s: |s|=LH(|\phi\rangle)} \langle s|\phi\rangle \otimes |s\rangle & \text{otherwise.} \end{cases}$$

The *inverse* of *HalfSWAP*, denoted HalfSWAP^{-1} , is obtained directly from *HalfSWAP* by replacing $LH(|\phi\rangle)$ in the above definition with $RH(|\phi\rangle)$.

3.2 How to Deal with a Set of Different Objects

All quantum functions in Sect. 3.1 take only single quantum states in $\mathcal{H}_\infty$ as their inputs. For practical applications of these quantum functions, we need to further deal with a problem that consists of various different objects, such as numbers, graphs, matrices, etc. For this purpose, we wish to set up an appropriate *binary encoding* to transform those objects into a single quantum state with the help of designated “separators”. In the polynomial-time setting [16], the scheme of quantum recursion is powerful enough to handle several inputs altogether as a single encoded quantum state in a way similar to multiple tapes of a Turing machine being simulated by a single-tape Turing machine with only polynomial overhead. However, since we aim at capturing quantum logarithmic-time computability instead, we cannot take the same kind of approach to cope with multiple inputs. We thus need to ponder how to circumvent this difficulty. As our solution to this problem, we introduce extra schemes that handle encodings.

We assume the standard lexicographical order on $\{0, 1\}^*$: $\lambda, 0, 1, 00, 01, 10, 11, 000, \dots$ and, based on this ordering, we assign natural numbers to $\{0, 1\}^*$ as $\text{bin}(0) = \lambda$, $\text{bin}(1) = 0$, $\text{bin}(2) = 1$, $\text{bin}(3) = 00$, $\text{bin}(4) = 01$, $\text{bin}(5) = 10$, $\text{bin}(6) = 11$, $\text{bin}(7) = 000$, etc. In contrast, for a fixed number $k \in \mathbb{N}^+$, $\text{bin}_k(n)$ denotes the lexicographically n th string in $\{0, 1\}^k$ as long as $n \in [2^k]$. For instance, we obtain $\text{bin}_3(1) = 000$, $\text{bin}_3(2) = 001$, $\text{bin}_3(3) = 010$, etc.

Let us describe our encoding scheme for eight symbols in $\{0, 1, 2, \neg, B, H, S, T\}$, where $\neg$, B , H , S , and T respectively stand for an “ending”, a “blank”, a “head”, a “separator”, and a “time”. With the use of three bits, we set $\hat{0} = 000$, $\hat{1} = 001$, $\hat{B} = 010$, $\hat{\neg} = 011$, $\hat{2} = 111$, $\hat{H} = 100$, $\hat{S} = 110$, and $\hat{T} = 101$.

Given a binary string $s = s_1 s_2 \cdots s_k$, the notation $\tilde{s}^{(-)}$ denotes $\hat{s}_1 \hat{s}_2 \cdots \hat{s}_k$ and $\tilde{s}$ denotes $\tilde{s}^{(-)\dagger}$. For convenience, we also define $\tilde{\lambda} = \hat{\lambda}$. This makes us encode, for example, the number 8 into $\widetilde{\text{bin}(8)} = 001 = \hat{0}\hat{0}\hat{1}$ and $\widetilde{\text{bin}_3(2)}$ also equals 001 . Given a quantum state $|\phi\rangle \in \mathcal{H}_\infty$, we finally define its encoding $|\tilde{\phi}\rangle$ as $\sum_{s:|s|=\ell(|\phi\rangle)} |\tilde{s}\rangle \langle s|\phi\rangle$. Note that $\ell(|\tilde{\phi}\rangle) = 3\ell(|\phi\rangle) + 3$.

Fix a string $r_0 \in \{0, 1\}^+$ and let x denote any binary string with $|x| = k|r_0|$ for an appropriate number $k \in \mathbb{N}^+$. We first partition x section-wise into $x_1 x_2 \cdots x_k$ with $|x_i| = |r_0|$ for any $i \in [k]$. For convenience, we say that x *section-wise contains no* r_0 if $x_i \neq r_0$ holds for all indices $i \in [k]$. Let $NON_{r_0} = \{x \in \{0, 1\}^* \mid |x| \equiv 0 \pmod{|r_0|}, x \text{ section-wise contains no } r_0\}$. Similarly, we set $NON_{r_0}(|\phi\rangle) = \{x \in \{0, 1\}^* \mid \langle x r_0 | \phi \rangle \neq 0, |x| \equiv 0 \pmod{|r_0|}, x \text{ section-wise contains no } r_0\}$. In particular, when $r_0 = \hat{2}$, we omit r_0 from NON_{r_0} and $NON_{r_0}(|\phi\rangle)$.

We introduce the following *code skipping schemes*. We do not unconditionally include them to *EQS* but we use them in a certain restricted situation discussed later. This is because they are too powerful in use for our purpose.

*) From g, h and $r_0 \in \{0, 1\}^+$, we define $\text{CodeSKIP}_+[r_0, g, h]$ and $\text{CodeSKIP}_-[r_0, g, h]$ as follows:

$$\begin{aligned} (i) \quad \text{CodeSKIP}_+[r_0, g, h](|\phi\rangle) &= \begin{cases} |\phi\rangle & \text{if } NON_{r_0}(|\phi\rangle) = \emptyset, \\ \sum_{x \in NON_{r_0}(|\phi\rangle)} (g(|x r_0\rangle) \otimes h(\langle x r_0 | \phi \rangle)) & \text{o.w.} \end{cases} \\ (ii) \quad \text{CodeSKIP}_-[r_0, g, h](|\phi\rangle) &= \begin{cases} |\phi\rangle & \text{if } NON_{r_0}(|\phi\rangle) = \emptyset, \\ \sum_{x \in NON_{r_0}(|\phi\rangle)} (g(|x\rangle) \otimes h(\langle x | \phi \rangle)) & \text{o.w.} \end{cases} \end{aligned}$$

To handle these code skipping schemes, we introduce a new scheme, called Scheme IV, which is a variant of the multi-qubit quantum recursion scheme (Scheme T). Recall that, in Scheme T, we inductively discard k qubits from an input quantum state $|\phi\rangle$ until we consume all qubits except for the last t qubits. In the fast quantum recursion, on the contrary, we discard a half of the second part of input quantum state $|x r_0\rangle \otimes |\phi\rangle$. By halving the input at each step, this recursive process quickly terminates.

Definition 2. The class *EQS* is the smallest set of quantum functions that contains the quantum functions of Scheme I and is closed under Schemes I–III as well as Scheme IV defined below. Similarly, $\widehat{EQS}$ is defined with no use of the item 8) of Scheme I.

IV. The code-controlled fast quantum recursion scheme. From d, g, h , numbers $k, t \in \mathbb{N}^+$, $r_0 \in \{0, 1\}^+$, $\mathcal{P}_{|r_0|} = \{p_u\}_{u \in \{0, 1\}^{|r_0|}}$, and $\mathcal{F}_{|r_0|} = \{f_u\}_{u \in \{0, 1\}^{|r_0|}}$ (where d, g , and h are not defined using Scheme IV but d and h may be defined using $\text{CodeSKIP}_+[\cdot]$ and $\text{CodeSKIP}_-[\cdot]$), we define $F \equiv CFQRec_t[r_0, d, g, h | \mathcal{P}_{|r_0|}, \mathcal{F}_{|r_0|}]$ as follows:

$$\begin{aligned} (i) \quad F(|x r_0\rangle |\phi\rangle) &= g(|x r_0\rangle \otimes |\phi\rangle) \quad \text{if } x = \lambda \text{ or } \ell(|\phi\rangle) \leq t, \\ (ii) \quad F(|x r_0\rangle |\phi\rangle) &= \sum_{u:|u|=|r_0|} \sum_{v:|v|=\ell(\langle u | x r_0 \rangle)} (h(|u\rangle \otimes |v\rangle) \otimes p_u^{-1}(\langle v | \zeta_{u, p_u, \phi} \rangle)) \quad \text{o.w.,} \end{aligned}$$

where $x \in NON_{r_0}$, $|\zeta_{u,p_u,\phi}\rangle = \sum_{s:|s|=NH(|\phi\rangle)} (f_u(|u|x'r_0\rangle \otimes |s\rangle) \otimes \langle s|\psi_{p_u,\phi}\rangle)$, $d(|x'r_0\rangle) = |x'r_0\rangle$ with $x' \in NON_{r_0}$, $|\psi_{p_u,\phi}\rangle = p_u(|\phi\rangle)$, $NH \in \{LH, RH\}$, and $f_u \in \{F, I\}$. Moreover, p_u and NH must satisfy that either $p_u = I$ and $NH = LH$, or $p_u = HalfSWAP$ and $NH = RH$. In the other case where an input, say, $|y\rangle$ to F satisfies $NON_{r_0}(|y\rangle) = \emptyset$, we automatically set $F(|y\rangle) = |y\rangle$.

In (ii), since the size $|s|$ is exactly $LH(|\phi\rangle)$ (or $RH(|\phi\rangle)$), f_u takes inputs of length $(\ell(|x'r_0\rangle) - |r_0|) + \lceil \ell(|\phi\rangle)/2 \rceil$ (or $\lceil \ell(|\phi\rangle)/2 \rceil$). Hence, within $\lceil \log \ell(|\phi\rangle) \rceil$ recursive steps, the whole process terminates.

When $r_0 = 1$, in particular, Scheme IV (ii) has the form: $F(|0^m 1\rangle|\phi\rangle) = \sum_{v:|v|=m} h(|0\rangle|v\rangle) \otimes p_0^{-1}(|v\rangle|\zeta_{0,p_0,\phi}\rangle)$, where $|\zeta_{0,p_0,\phi}\rangle = \sum_{s:|s|=NH(|\phi\rangle)} (f_0(|0^{m-1}1\rangle \otimes |s\rangle) \otimes \langle s|\psi_{p_0,\phi}\rangle)$.

To understand Scheme IV better, we provide in Example 3 a simple example of how to calculate the quantum function $CFQRec_t[r_0, d, g, h|_{\mathcal{P}_{|r_0|}}, \mathcal{F}_{|r_0|}]$ step by step.

Example 3. We wish to demonstrate how to calculate the quantum function $F \equiv CFQRec_t[r_0, d, g, h|_{\mathcal{P}_{|r_0|}}, \mathcal{F}_{|r_0|}]$ when $r_0 = 1$, $d = I$, $g \equiv I$, and $h \equiv REMOVE^2 \circ Branch[I, NOT] \circ REP^2 \circ SWAP$. Note that $h(|01\rangle) = |11\rangle$ and $h(|010^m 1\rangle) = |10^{m+1} 1\rangle$ for any $n \in \mathbb{N}$. We further set $p_0 = HalfSWAP$, $p_1 = I$, $f_0 = F$, and $f_1 = I$. In short, F equals $CFQRec_1[1, d, g, h|_{\{p_0, p_1\}}, \{f_0, f_1\}]$. Let $x \in \{0, 1\}^*$ and $|\phi\rangle \in \mathcal{H}_\infty$.

- (1) We start with the base case. If $x = \lambda$, then $F(|1\rangle|\phi\rangle) = g(|1\rangle|\phi\rangle) = |1\rangle|\phi\rangle$. If $x = 0^m 1$ with $m \geq 1$ and $\ell(|\phi\rangle) \leq 1$, then $F(|0^m 1\rangle|\phi\rangle) = g(|0^m 1\rangle|\phi\rangle) = |0^m 1\rangle|\phi\rangle$.
- (2) Hereafter, we assume that $x \neq \lambda$ and $\ell(|\phi\rangle) \geq 2$. For simplicity, let $|\phi\rangle = |u\rangle$ for a certain string $u \in \{0, 1\}^*$. When $u = u_1 u_2 \cdots u_n$ and $1 \leq i < j \leq n$, we use the succinct notation $u_{[i,j]}$ to denote the string $u_i u_{i+1} \cdots u_j$.
- (a) We first calculate $F(|01\rangle|u\rangle)$, where $u = u_1 u_2 u_3 u_4$. Since $|u\rangle = |u_{[1,2]}\rangle |u_{[3,4]}\rangle$, we obtain $p_0(|u\rangle) = |u_{[3,4]}\rangle |u_{[1,2]}\rangle$. We also obtain $|\zeta_{0,p_0,u}\rangle = F(|1\rangle|u_{[3,4]}\rangle) \otimes |u_{[1,2]}\rangle = |1\rangle \otimes |u_{[3,4]}\rangle |u_{[1,2]}\rangle$ by (1), and thus $F(|01\rangle|u\rangle)$ equals $h(|0\rangle \otimes |1\rangle) \otimes p_0^{-1}(|u_{[3,4]}\rangle |u_{[1,2]}\rangle)$, which is $|11\rangle |u_{[1,2]}\rangle |u_{[3,4]}\rangle = |11\rangle |u\rangle$.
- (b) Next, we calculate $F(|001\rangle|u'\rangle)$ for $u' = u_1 u_2 \cdots u_8$. Note that $|\zeta_{0,p_0,u'}\rangle = F(|01\rangle|u'_{[5,8]}\rangle) \otimes |u'_{[1,4]}\rangle = |11\rangle \otimes |u'_{[5,8]}\rangle |u'_{[1,4]}\rangle$ by (a). From this, we obtain $F(|001\rangle|u'\rangle) = h(|0\rangle \otimes |11\rangle) \otimes p_0^{-1}(|u'_{[5,8]}\rangle |u'_{[1,4]}\rangle) = |101\rangle |u'\rangle$.

3.3 Basic Properties of EQS

We have formally introduced $\widehat{EQS}$ and EQS in Definition 2 with the use of Schemes I–IV. We then study the basic properties of quantum functions in $\widehat{EQS}$ and EQS throughout this subsection.

Lemma 4. *Any quantum function in $\widehat{EQS}$ is dimension-preserving and norm-preserving.*

When the length of an input qustring is “short” enough, it may be possible to use the scheme of “full-scale” quantum recursion, similar to Scheme T. Let h be any quantum function in EQS that is not defined using Scheme IV. Consider a quantum function $\hat{h}$ defined inductively as $\hat{h}(|r_0\rangle) = |r_0\rangle$ and $\hat{h}(|xr_0\rangle) = \sum_{u:|u|=|r_0|} h(|u\rangle \otimes \hat{h}(\langle u|x r_0\rangle))$ for any $x \in NON_{r_0}$ with $x \neq \lambda$. To construct such a quantum function $\hat{h}$, we generally use the k -qubit quantum recursion scheme and it may not be constructed by Schemes I–IV. Nonetheless, for relatively “short” inputs, we can compute the value of $\hat{h}$ within our system EQS .

Proposition 5. *Let g , h , and $\hat{h}$ be given above. There exists a quantum function F such that, for any $(x, |\phi\rangle)$ with $x \in NON_{r_0}$ and $|\phi\rangle \in \mathcal{H}_\infty$, if $|x| \leq |r_0| \log \ell(|\phi\rangle)$, then $F(|xr_0\rangle|\phi\rangle) = \hat{h}(|xr_0\rangle) \otimes |\phi\rangle$ hold. However, there is no guarantee that $F(|xr_0\rangle|\phi\rangle)$ matches $\hat{h}(|xr_0\rangle) \otimes |\phi\rangle$ when $|x| > |r_0| \log \ell(|\phi\rangle)$.*

We further merge k encoded strings $\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_k$ into a single string of the form $\tilde{x}_1\tilde{x}_2 \cdots \tilde{x}_k\hat{2}$, where $\hat{2}$ serves as an endmarker. Let $k \geq 2$ and let $g_1, g_2, \dots, g_k$ denote k arbitrary quantum functions. We then define

$$\begin{aligned} CodeSKIP_k[\hat{2}, g_1, \dots, g_k](|\tilde{x}_1\tilde{x}_2 \cdots \tilde{x}_k\hat{2}\rangle|\phi\rangle) \\ = g_1(|\tilde{x}_1\rangle) \otimes g_2(|\tilde{x}_2\rangle) \otimes \cdots \otimes g_k(|\tilde{x}_k\hat{2}\rangle) \otimes |\phi\rangle. \end{aligned}$$

This quantum function $CodeSKIP_k[\hat{2}, g_1, \dots, g_k]$ can be constructed from $g_1, g_2, \dots, g_k$ by applying $CodeSKIP_+[\cdot]$ as follows. Let $\bar{r}_0 = \hat{1}$ and $r_0 \equiv \hat{2}$. We set $G_{[k,k]} \equiv CodeSKIP_+[r_0, g_k, I]$. For any $i \in [k-1]$, we inductively set $G_{[i,k]} \equiv CodeSKIP_+[\bar{r}_0, g_i, G_{[i+1,k]}]$. The desired quantum function equals $G_{[1,k]}$.

Lemma 6. *Let $k \in \mathbb{N}^+$ and let $g_1, g_2, \dots, g_k$ be any quantum functions in EQS . There exists a quantum function F in EQS such that $F(|\hat{S}\tilde{x}_1\tilde{x}_2 \cdots \tilde{x}_k\hat{2}\rangle|\phi\rangle) = CodeSKIP_k[\hat{2}, g_1, \dots, g_k](|\hat{S}\tilde{x}_1\tilde{x}_2 \cdots \tilde{x}_k\hat{2}\rangle|\phi\rangle)$ as long as $|\hat{S}\tilde{x}_1\tilde{x}_2 \cdots \tilde{x}_k\hat{2}| \leq |\hat{2}| \log \ell(|\phi\rangle)$. However, the equality may not hold if $|\hat{S}\tilde{x}_1\tilde{x}_2 \cdots \tilde{x}_k\hat{2}| > |\hat{2}| \log \ell(|\phi\rangle)$.*

By Lemma 6, it is possible to use, e.g., $CodeSKIP[\hat{2}, g_1, \dots, g_k]$ as if it is an EQS function as long as the premise of the lemma is satisfied.

In general, we cannot repeatedly apply the same quantum function in an inductive manner. However, in a certain specific case, it is possible to make a repeated application of any quantum function in EQS .

Lemma 7. *Let $r_0 \in \{0, 1\}^+$ and let g be any quantum function in EQS . There exists a quantum function F in EQS such that $F(|\hat{T}^{\lceil \log \ell(|\phi\rangle) \rceil} \hat{S}\rangle|x\hat{2}\rangle|\phi\rangle) = |\hat{T}^{\lceil \log \ell(|\phi\rangle) \rceil} \hat{S}\rangle \otimes g^{m(|\phi\rangle)}(|x\hat{2}\rangle) \otimes |\phi\rangle$, where $m(|\phi\rangle) = \lceil \log \ell(|\phi\rangle) \rceil$.*

Proof. Let F denote the desired quantum function. We set $\mathcal{P} = \{I\}_{u \in \{0,1\}^3}$, $\mathcal{F}_3 = \{F\}_{u \in \{0,1\}^3}$, $h \equiv Branch_3[\{g_u\}_{u \in \{0,1\}^3}$ with $g_{\hat{T}} = CodeSKIP_+[\hat{S}, I, g]$, and $g_u = I$ for any $u \neq \hat{T}$. The quantum function F is then defined as $F \equiv CFQRec_1[\hat{2}, I, I, h|\mathcal{P}_3, \mathcal{F}_3]$. $\square$

The strength of Scheme IV is further exemplified by an implementation of a simple binary search algorithm.

Lemma 8. *Given any $x \in \{0, 1\}^k$, assume that x expresses $\text{bin}_k(m)$ for a certain number $m \in [2^k]$. For any s with $|s| = 2^k$, BinSearch satisfies the following equality: $\text{BinSearch}(|\hat{x}\rangle|\hat{0}\rangle|\hat{2}\rangle|s\rangle) = |\widehat{s_m}\rangle|\hat{x}\rangle|\hat{2}\rangle|s\rangle$, where s_m is the m th bit of s . This quantum function BinSearch is defined in EQS.*

4 Relationships to Quantum Computability

We have studied basic properties of quantum functions in EQS throughout Sect. 3. In this section, we will look into relationships of these quantum functions to other platforms of limited computability.

4.1 Comparison with Logarithmic-Time Quantum Turing Machines

Let us consider a computational model of *quantum Turing machine* (or QTM) that runs in logarithmic time. In what follows, we discuss a close relationship between quantum functions defined by Schemes I–IV and those quantum Turing machines.

A *logarithmic-time quantum Turing machine* (or a *logtime QTM*, for short) is equipped with a read-only input tape, multiple rewritable work tapes of $O(\log n)$ cells, and a rewritable index tape of $\lceil \log n \rceil + 1$ cells, where n refers to the length of an input. A QTM M is formally expressed as $(Q, \Sigma, \{\triangleright, \triangleleft\}, \delta, q_0, Q_{acc}, Q_{rej})$, where Q is a finite set of inner states, $\Sigma = \{0, 1\}$ is an alphabet, $\triangleright$ and $\triangleleft$ are endmarkers, δ is a quantum transition function, $q_0 (\in Q)$ is the initial (inner) state, and Q_{acc} (resp., Q_{rej}) ($\subseteq Q$) is a set of accepting (resp., rejecting) states. We use an additional convention that the input tape and the index tape, and all the work tapes have both endmarkers to mark the areas of these tapes to use. A logtime QTM M begins with an input qustring $|\phi\rangle$ given on the input tape. Let $|\phi\rangle = \sum_{x \in \Sigma^n} \alpha_x |x\rangle$ with $n \in \mathbb{N}^+$ and $k = \lceil \log n \rceil$.

To access the tape cell indexed by m , M first produces $\text{bin}_k(m)$ on the index tape with an auxiliary bit b and then enters a designated *query state*. If the index tape contains $|\text{bin}_k(m)\rangle|b\rangle$, then this state becomes $|\text{bin}_k(m)\rangle|b \oplus x_{(m)}\rangle$ as the consequence of the query, where $x_{(m)}$ is the m th input symbol of each input $x \in \Sigma^+$.

The quantum transition function δ takes a quantum transition of the form $\delta(q, \sigma, \tau) = \sum_r \alpha_r |p, \xi, \eta, d, d'\rangle$, where $r = (p, \xi, \eta, d, d')$, which indicates that, if M is in inner state q reading σ on the index tape and τ on the work tape, then, in a single step, with transition amplitude α_r , M changes q to p , writes ξ over σ moving the input-tape head in direction d , and writes η over τ moving the work-tape head in direction d' . For practicality, we can limit the scope of transition amplitudes of QTMs. In this work, we allow only the following two forms of quantum transitions: $\delta(q, \sigma, \tau) = \cos \theta |p_1, \xi_1, \eta_1, d_1, d'_1\rangle + \sin \theta |p_2, \xi_2, \eta_2, d_2, d'_2\rangle$ and $\delta(q, \sigma, \tau) = e^{i\theta} |p, \xi, \eta, d, d'\rangle$ (based on [3]).

A QTM is required to satisfy the so-called “well-formedness condition” (see, e.g., [13, 15]) to guarantee that the behaviors of the QTM obeys the law of quantum physics. A *surface configuration* of M on an input of length n is an element of $Q \times [n] \times \{0, 1, B\}^{\leq k} \times [k] \times \{0, 1, B\}^{\leq k} \times [k]$. The *time-evolution operator* of M on inputs of length n is a map from superpositions of surface configurations of M on inputs of length n to other superpositions of surface configurations resulting by a single application of δ of M . A QTM M is *well-formed* if its time-evolution operator of M preserves the ℓ_2 -norm.

At every step, we first apply δ to a superposition of surface configurations and then perform a measurement in the halting (inner) states (i.e., either accepting states or rejecting states). If M is not in a halting state, we move to the next step with the quantum state obtained by tracing out all halting surface configurations. Generally, QTMs can “recognize” sets of qustrings. See [14].

For convenience, we modify M slightly so that M produces 1 (resp., 0) in the start cell of the first work tape when M enters a designated final state (in place of accepting/rejecting states). We say that M *produces b with probability γ* if, after M halts, we observe b as an outcome of M with probability γ .

We call M *logarithmic time* (or logtime, for short) if we force M to stop its application of δ after $O(\log n)$ steps. We do not require that all computation paths to terminate within the specified time.

In the following two theorems (Theorems 9 and 10), we establish the “computational” equivalence between EQS and logtime QTMs. Assume that the input tape holds a superposition $|\phi\rangle$ of binary inputs, the index tape holds $B^m\triangleleft$, and the work tape holds $B^m\triangleleft$, where $m = \lceil c \log \ell(|\phi\rangle) \rceil$ for a certain constant $c > 0$. We translate the content of M ’s tapes into a single quantum state. Let z denote the content of a tape without the right endmarker $\triangleleft$. We expand z to $z_1 H z_2$ with $z = z_1 z_2$ by including a tape-head position, where H is a designated symbol indicating that the tape head is scanning the leftmost symbol of z_2 . We call this $z_1 H z_2$ a *modified tape content*. We translate it to $|\tilde{z}_1^{(-)} \hat{H} \tilde{z}_2^{(-)}\rangle$. The modified initial tape content with the endmarker is of the form $|\hat{H} \widetilde{B^m}\rangle$.

Theorem 9. *Any logtime QTM M with t work tapes can be simulated by an appropriate quantum function F in EQS in the sense that, for any $b \in \{0, 1\}$, $\alpha \in [0, 1]$, and $|\phi\rangle \in \Phi_\infty$, M takes input $|\phi\rangle$ and finally produces b with probability α iff $\|\langle b | \psi_{F, \xi} \rangle\|^2 = \alpha$ holds, where $|\xi\rangle = |\hat{S}\rangle |\widetilde{B^m}\rangle^{\otimes (t+2)} |\hat{2}\rangle \otimes |\phi\rangle$ and $|\psi_{F, \xi}\rangle = F(|\xi\rangle)$ with $m = \lceil c \log n \rceil$ for a certain constant $c \in \mathbb{N}^+$.*

Proof Sketch. Let M denote any logtime QTM equipped with a read-only input tape, a rewritable index tape, and t rewritable work tapes with $t \in \mathbb{N}^+$. It suffices to assume that M uses the binary input alphabet because, in any non-binary alphabet case, we can encode each input symbol into a binary one using an appropriate encoding scheme.

We treat the content of each tape (except for the input tape) as a code block of an appropriate quantum function. We maintain the content of the index tape and the work tapes as a part of qustring. We intend to simulate each move of M on $|\phi\rangle$ by applying an appropriately defined quantum function. A run of M

from the initial configuration to halting configurations can be simulated using the fast quantum recursion. To explain this simulation, for convenience, we split each move of M into two separate processes: an input access and a tape head move.

We use $CodeSKIP_+[\cdot]$ and $CodeSKIP_-[\cdot]$ to simulate a tape head move on $t + 1$ tapes (except for the input tape) of M . Recall that, in a single step, a tape head changes a tape symbol and moves to an adjacent cell. This process can be expressed by a quantum function as follows. Note that, by Lemma 6, it suffices to focus on each block of encoded tape content. The modified tape content of each tape is translated into $\hat{z}_1^{(-)} \hat{H} \hat{z}_2^{(-)} \hat{\cdot}$. We need to consider three consecutive cells whose middle cell is being scanned by the tape head. Let $\hat{\sigma}_1 \hat{H} \hat{\sigma}_2$ denote the encoded content of such consecutive cells. Here, we set $J = NOT$ if $\sigma_2 \neq \tau_2$ and $J = I$ otherwise. A change of $\hat{\sigma}_1 \hat{H} \hat{\sigma}_2$ to $\hat{H} \hat{\sigma}_1 \hat{\tau}_2$ is simulated by applying $CodeSKIP_+[\hat{H}, h, I]$, where $h \equiv REMOVE^2 \circ SWAP \circ J \circ REP^2$. In contrast, we apply $CodeSKIP_-[\hat{H}, I, h']$ to change $\hat{\sigma}_1 \hat{H} \hat{\sigma}_2$ to $\hat{\sigma}_1 \hat{\tau}_2 \hat{H}$, where $h' \equiv J \circ SWAP$.

Recall that M uses only two forms of quantum transitions. Those transitions can be correctly simulated by the items 1)–3) of Scheme I.

We then simulate a movement of the tape head on a work tape. Assume that M is in inner state q , reading σ at the k th cell of the work tape, where w is written. We express this configuration as $|\tilde{q}\rangle|\tilde{w}\rangle|0^m1\rangle$. Proposition 5 makes it possible to use in part the scheme of quantum recursion (Scheme T).

Next, we simulate a query access to an input bit. Assume that $|\phi\rangle$ is written on an input tape and $|s\rangle$ is written on an index tape. Assume that the index tape contains $|\text{bin}_k(m)\rangle|a\rangle$, where a is an auxiliary bit. After the query, M changes it into $|x\rangle|a \oplus x_{(m)}\rangle$. We need to build a quantum function that simulates this entire process. This can be done by applying *BitSearch* and $COPY_1$, where $COPY_1$ is defined as $COPY_1(|a\rangle|\phi\rangle) = \sum_{s \in \{0,1\}} |a \oplus s\rangle\langle s|\phi\rangle$ for any quantum state $|\phi\rangle \in \mathcal{H}_\infty$ and any bit a .

Finally, we combine the above two types of moves into one and express it by a single application of a certain quantum function, say, K . Note that M accesses only the first $O(\log \ell(|\phi\rangle))$ cells of the work tape. We thus use Scheme IV to repeatedly apply K at most $\lceil \ell(|\phi\rangle) \rceil$ times. $\square$

The converse of the above theorem is stated as Theorem 10. Assume that inputs of quantum functions have the form $|\gamma_\phi\rangle = \sum_{x \in NON_{r_0}} \alpha_x |xr_0\rangle \otimes |\phi\rangle$.

Theorem 10. *All quantum functions F defined by Schemes I–IV are computed by certain logtime QTMs M in the sense that, for any $b \in \{0, 1\}$ and $|\phi\rangle \in \Phi_\infty$, $\|\langle \psi_{F,\phi} | \xi_{M,\phi} \rangle\| = 1$ holds, where $|\psi_{F,\phi}\rangle = F(|\phi\rangle)$, and $|\xi_{M,\phi}\rangle$ is the superposition of final configurations of M on $|\phi\rangle$ together with $\sum_{x \in NON_{r_0}} \alpha_x |xr_0\rangle$ produced on a work tape.*

Note that M 's tapes may contain additional “garbage” information, which may be a quantum state of large dimension, and thus we need to use $\|\cdot\|$ instead of $|\cdot|$.

Proof Sketch of Theorem 10. We can show that all quantum functions F defined by the items 1)–8) of Scheme I are computed by certain logtime QTM. The items 1)–5) of Scheme I can be computed easily. For the items 6)–7), we first need to find the location of the last bit of an input string. To do so, we need to compute the length of the input string. This was shown in [1, Lemma 6.1]. For the item 8), it then suffices to observe the first qubit using $|a\rangle$.

We next show that Schemes II–III are realized by QTMs running in log time by induction on the construction process of a target quantum function F . An application of a certain scheme changes the order of input qubits. It is thus important to trace the changes made by the schemes. For this purpose, we need to remember the construction history of F in one of the work tapes. This is possible because the number of construction steps (Schemes I–IV) (thus the number of scheme applications) is finite. We use this construction history to follow the changes of qubit locations made by the items 5)–7) of Scheme I.

Let us consider $F \equiv \text{Compo}[g, h]$ in Scheme II. By induction hypothesis, there are two logtime QTMs M_g and M_h that respectively compute g and h . We construct another QTM M that first checks whether $\ell(|\phi\rangle) \leq 1$. If so, we do nothing. Assume otherwise. On any input $|\phi\rangle$, we first run M_h on $|\phi\rangle$. When M_h halts, we run M_g . In these runs of M_h and M_g , we need to modify their computations by following the construction history generated during the simulation of h by M_h .

For Scheme III, consider the quantum function $F \equiv \text{Branch}[g, h]$. Take two logtime QTMs M_g and M_h for g and h , respectively. We design a new QTM, say, N to behave as follows. On input $|\phi\rangle$, we read the first qubit and determine which machine (either M_g or M_h) to run on the rest of the input. Since both M_g and M_h simulate g and h , respectively, this new machine N simulates F as well.

Concerning Scheme IV, let $F \equiv \text{CFQRec}_t[r_0, d, g, h | \mathcal{P}_{|r_0|}, \mathcal{F}_{|r_0|}]$ and assume that an input has the form $|xr_0\rangle|\phi\rangle$. We first calculate the length $\ell(|\phi\rangle)$ by the binary search strategy in log time. If either $\ell(|\phi\rangle) < t$ or $x = \lambda$, then we run M_g on $|xr_0\rangle|\phi\rangle$ and halt. Assume otherwise. If h is defined using none of $\text{CodeSKIP}_+[\cdot]$ and $\text{CodeSKIP}_-[\cdot]$, then induction hypothesis guarantees the existence of a logtime QTM M_h to use. Hence, assume that h is constructed using at least one of $\text{CodeSKIP}_+[\cdot]$ and $\text{CodeSKIP}_-[\cdot]$. First, we build a QTM that simulates $\text{CodeSKIP}_+[r_0, g', h']$ without requiring “logarithmic” runtime. Such a QTM can be easily built by searching for the first appearance of r_0 and run $M_{g'}$ and $M_{h'}$ in parallel. It is important to note that, as long as the input length to this QTM is $O(\log \ell(|\phi\rangle))$, the QTM halts within $O(\log \ell(|\phi\rangle))$ steps. Similarly, we can build another QTM for $\text{CodeSKIP}_-[r_0, g', h']$.

Starting with $|xr_0\rangle|\phi\rangle$, we first move a work-tape head passing over the $\lceil \ell(|\phi\rangle) \rceil$ blocks of size $|r_0|$ each. Recursively, we move back the tape head and run M_h . For this purpose, we write 0 and 1 on an index tape whenever we choose $p_u = I$ and $p_u = \text{HalfSWAP}$, respectively, because we need to trace the location of the start of the recursively halved input until we reach either $x = \lambda$ or $\ell(|\phi\rangle) < t$. The entire algorithm thus requires $O(\log \ell(|\phi\rangle))$ steps. $\square$

4.2 Computational Complexity of BQLOGTIME

It was shown in [12, Lemma 8] how to simulate a one-tape well-formed stationary QTM running in linear time on an appropriate one-tape probabilistic Turing machine (or a PTM) in linear time. In a similar vein, we study the computational complexity of logtime QTMs.

Let us consider a language L over $\{0, 1\}$ that satisfies the following condition: there are a constant $\varepsilon \in [0, 1/2)$ and a logtime QTM M whose amplitude set is K such that (i) for any input $x \in L$, M accepts x with probability at least $1 - \varepsilon$ and (ii) for any $x \notin L$, M rejects x with probability at least $1 - \varepsilon$. These conditions are referred to as *bounded-error probability*. The notation BQLOGTIME_K denotes the collection of all such languages L . In the classical setting, the notation PPLOGTIME is used to express the family of all languages recognized by unbounded-error logtime PTMs. Moreover, we can define the nondeterministic variant of DLOGTIME [1] and denote it by NLOGTIME .

Theorem 11. $\text{BQLOGTIME}_{\mathbb{Q}} \subsetneq \text{PPLOGTIME}$ and $\text{NLOGTIME} \not\subseteq \text{BQLOGTIME}_{\mathbb{C}}$.

For the proof of this theorem, due to the characterization theorem of EQS in terms of logtime QTMs, it suffices to verify the following impossibility assertion. The parity function $\text{Parity}(x)$ is defined by $\text{Parity}(x_1x_2 \cdots x_n) = \bigoplus_{i=1}^n x_i$ for any n bits $x_1, x_2, \dots, x_n \in \{0, 1\}$.

Lemma 12. *No logtime QTM can compute the following functions with bounded-error probability: the parity function and the OR function.*

We postpone the proof of Lemma 12 and, assuming that Lemma 12 is true, we first prove Theorem 11.

Proof of Theorem 11. Consider a language L in $\text{BQLOGTIME}_{\mathbb{Q}}$, which is recognized by a logtime QTM M with bounded-error probability. We first modify M in the following way. We prepare two extra work tapes. One of them is used as an internal clock by moving a tape head always to the right. To avoid any unwanted interference after a computation halts prematurely, we use the other extra tape as a “garbage tape” to which M dumps all information produced at the time of entering halting states, so that M continues its operation without actually halting. Lemma 8 of [12] shows the existence of a constant $d \in \mathbb{N}^+$ and an NTM N such that $d^{\text{Time}_M(x)} \cdot p_M(x) = \#N(x) - \#\bar{N}(x)$ for every x , where $\#N(x)$ (resp., $\#\bar{N}(x)$) denotes the total number of accepting (resp., rejecting) computation paths of N on input x . This equality holds for logtime machines. The desired logtime PTM is obtained from N by assigning an equal probability to each nondeterministic transition.

Obviously, the OR function can be computed by an appropriately chosen logtime NTM. The separation between $\text{BQLOGTIME}_{\mathbb{Q}}$ and NLOGTIME comes from Lemma 12 because the OR function can be nondeterministically computed in log time. Since PPLOGTIME includes NLOGTIME , we also obtain the separation between $\text{BQLOGTIME}_{\mathbb{Q}}$ and PPLOGTIME . $\square$

To complete the proof of Theorem 11, we still need to provide the proof of Lemma 12. As noted in Sect. 3.2, we can produce a superposition of all indices of the input tape cells in log time so that we can access all the input bits quantumly at once. Since the input tape is read-only, this type of input access can be realized as a (black-box) query model used in the study of quantum query complexity.

In a (black-box) quantum query model, we solve a given problem as follows. We prepare a series of unitary transformations $U_0, U_1, \dots, U_k$ and a special oracle transformation Q_x that changes $|bin_k(m)\rangle|\phi\rangle$ to $(-1)^{x_{(m)}}|bin_k(m)\rangle|\phi\rangle$, where $x_{(m)}$ is the m th bit of a binary input x . Initially, we start with a simple initial quantum state $|\psi_0\rangle$, such as $|0^m\rangle$. We then compute $U_k Q_x \cdots Q_x U_1 Q_x U_0 |\psi_0\rangle$ by making k queries. Finally, we measure the resulting quantum state in the computational basis. In this model, we are concerned with the minimum number of queries necessary to solve the target problem.

Proof of Lemma 12. This proof comes from a result of quantum query complexity gap between quantum and deterministic query complexity of the parity function. Assume that a logtime QTM computes the parity function with bounded-error probability.

Claim 13. *Any logtime QTM can be simulated by a (black-box) quantum query model with $O(\log n)$ queries.*

Proof. Recall that a logtime QTM has a read-only input tape, which holds an input string. Whenever the QTM makes a query on the i th position, the machine immediately receives the information on the i th bit of the input string. We view the input tape as an oracle in a (black-box) query model and we further view this process as a procedure of forming a superposition of query words regarding positions and receiving their answers from the oracle.

When M changes $|bin_k(m)\rangle|b\rangle$ to $|bin_k(m)\rangle|b \oplus x_{(m)}\rangle$ in a query process, we simulate this transformation in the quantum query model as follows. Let $|\phi\rangle = |bin_k(m)\rangle$. Starting with $|\phi\rangle|b\rangle$, swap two registers, apply WH (Walsh-Hadamard), apply CQ_x (Controlled- Q_x) apply WH , and swap back the registers, where $CQ_x(|0\rangle|\phi\rangle) = |0\rangle|\phi\rangle$ and $CQ_x(|1\rangle|\phi\rangle) = (-1)^{x_{(m)}}|1\rangle|\phi\rangle$. We then obtain $|b \oplus x_{(m)}\rangle|\phi\rangle$, as requested. $\square$

Since the given QTM makes only $O(\log n)$ queries, we transform this QTM to a query model of $O(\log n)$ queries. By Claim 13, the parity function requires only $O(\log n)$ queries in the quantum query model. However, it is shown in [3] that, for the parity function of n Boolean variables, $n/2$ queries are necessary in the bounded-error quantum query model (while n queries are necessary in the deterministic query model). This is obviously a contradiction. The case of the OR function can be similarly handled. $\square$

References

1. Barrington, D.A.M., Immerman, N., Straubing, H.: On uniformity within NC^1 . J. Comput. System Sci. **41**, 274–306 (1990)

2. Barenco, A., et al.: Elementary gates for quantum computation. *Phys. Rev. A* **52**, 3457–3467 (1995)
3. Beals, R., Buhrman, H., Cleve, R., Mosca, M., de Wolf, R.: Quantum lower bounds by polynomials. *J. ACM* **48**, 778–797 (2001)
4. Benioff, P.: The computer as a physical system: a microscopic quantum mechanical Hamiltonian model of computers represented by Turing machines. *J. Statist. Phys.* **22**, 563–591 (1980)
5. Deutsch, D.: Quantum theory, the Church-Turing principle, and the universal quantum computer. *Proc. R. Soc. London Ser. A* **400**, 97–117 (1985)
6. Deutsch, D.: Quantum computational networks. *Proc. R. Soc. London Ser. A* **425**, 73–90 (1989)
7. Kleene, S.C.: General recursive functions of natural numbers. *Math. Ann.* **112**, 727–742 (1936)
8. Kleene, S.C.: Recursive predicates and quantifiers. *Trans. AMS* **53**, 41–73 (1943)
9. Kitaev, A.Y., Shen, A.H., Vyalıy, M.N.: *Classical and Quantum Computation* (Graduate Studies in Mathematics). American Mathematical Society (2002)
10. Nielsen, M.A., Chuang, I.L.: *Quantum Computation and Quantum Information*, 10th edn. Cambridge University Press, Cambridge (2016)
11. Soare, R.I.: Computability and recursion. *Bull. Symb. Log.* **2**(3), 284–321 (1996)
12. Tadaki, K., Yamakami, T., Lin, J.C.H.: Theory of one-tape linear-time Turing machines. *Theor. Comput. Sci.* **411**, 22–43 (2010)
13. Yamakami, T.: A foundation of programming a multi-tape quantum Turing machine. In: Kutyłowski, M., Pacholski, L., Wierzbicki, T. (eds.) *MFCS 1999*. LNCS, vol. 1672, pp. 430–441. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48340-3_39
14. Yamakami, T.: Quantum NP and a quantum hierarchy. In: *IFIP TCS 2002* (Under the Title of Foundations of Information Technology in the Era of Network and Mobile Computing), The International Federation for Information Processing, vol. 96 (Track 1), pp. 323–336. Kluwer Academic Press (2002)
15. Yamakami, T.: Analysis of quantum functions. *Int. J. Fund. Comput. Sci.* **14**, 815–852 (2003)
16. Yamakami, T.: A schematic definition of quantum polynomial time computability. *J. Symb. Log.* **85**, 1546–1587 (2020)
17. Yao, A.C.: Quantum circuit complexity. In: *FOCS 1993*, pp. 80–91 (1993)



Multityped Abstract Categorical Grammars and Their Composition

Pierre Ludmann, Sylvain Pogodalla^(✉), and Philippe de Groote

Université de Lorraine, CNRS, Inria, LORIA, 54000 Nancy, France
`{pierre.ludmann,sylvain.pogodalla,philippe.degroote}@loria.fr`

Abstract. This article introduces multityped abstract categorical grammars and show that a suitable composition operation can be defined.

Keywords: Abstract Categorical Grammar · Grammar Composition · Overloaded Signature

1 Introduction

Abstract categorical grammars (ACGs, [4]) is an expressive compositional framework for both syntax and semantics. It is known to deeply encode several context-free formalisms, such as context-free grammars (CFGs) or tree-adjoining grammars (TAGs, [7]). However, just as for these symbolic grammatical formalisms, large-scale grammars exhibit a combinatorial explosion of parsing ambiguity. A widespread method to tackle this issue is to use statistics and probabilities, leading for instance to probabilistic CFGs (pCFGs, [2, 11]) and probabilistic TAGs (pTAGs, [3, 10]). An important goal would then be to also extend ACGs with probabilities or weights.

Yet, ACGs come with features that make this extension non-trivial. In particular, as Fig. 1a illustrates with the main ACGs given as examples in this article, ACGs can be composed by making the parse structures of a grammar $\mathcal{G}_{t2s}$ the surface structures of another ACG $\mathcal{G}_{der2t}$, simply by *function composition*. The resulting composition $\mathcal{G} = \mathcal{G}_{t2s} \circ \mathcal{G}_{der2t}$ is an ACG.

Introducing preferences to some specific structures at a given level, eventually expressed by weights, requires to be able to distinguish between the structures at this level. For instance, distinguishing between left or right branching trees from other binary trees in $\Lambda(\Sigma_{tree})$ (done in Example 4), distinguishing between derivation trees of $\Lambda(\Sigma_{derivation})$ where adjunctions occur high or deep, distinguishing between strings containing some specific sequences in $\Lambda(\Sigma_{str})$, etc. Such distinctions, exemplified with dotted arrows in Fig. 1 and informally called distinguishing schemas, typically rely on defining an additional ACG to control the generated structures, such as $\mathcal{G}_{lr-tree}$ that maps $\Lambda(\Sigma_{lr-tree})$ to $\Lambda(\Sigma_{tree})$ in Fig. 1b, pretty much in the same way as adding states in an automaton allows for more fine-grained distinction between accepted inputs. This highlights an important new grammatical object: the composition of an ACG (plain arrows on Fig. 1)

with another ACG (dotted arrows) allowing for distinguishing between parse structures of the former ACG, e.g., $\mathcal{G}_{t2s} \circ \mathcal{G}_{lr-tree}$.

We then expect to have a well-behaved composition of these new grammatical objects as well. In particular the result should indeed be one of these new grammatical objects, as Fig. 1c illustrates (also see its formally defined counterpart in Fig. 5). We will see that the dotted arrows of Fig. 1 correspond to specific ACGs, namely *relabelings*. The difficulty we address here is then to combine these new grammatical objects into equivalent ones that also make use of relabelings.

To this end, this article introduces *multityped* ACGs (mACGs). Multityped ACG are the underlying discrete mathematical structures that will support weighting extension (not presented here). It also shows that a suitable notion of composition can be defined for multityped ACGs.

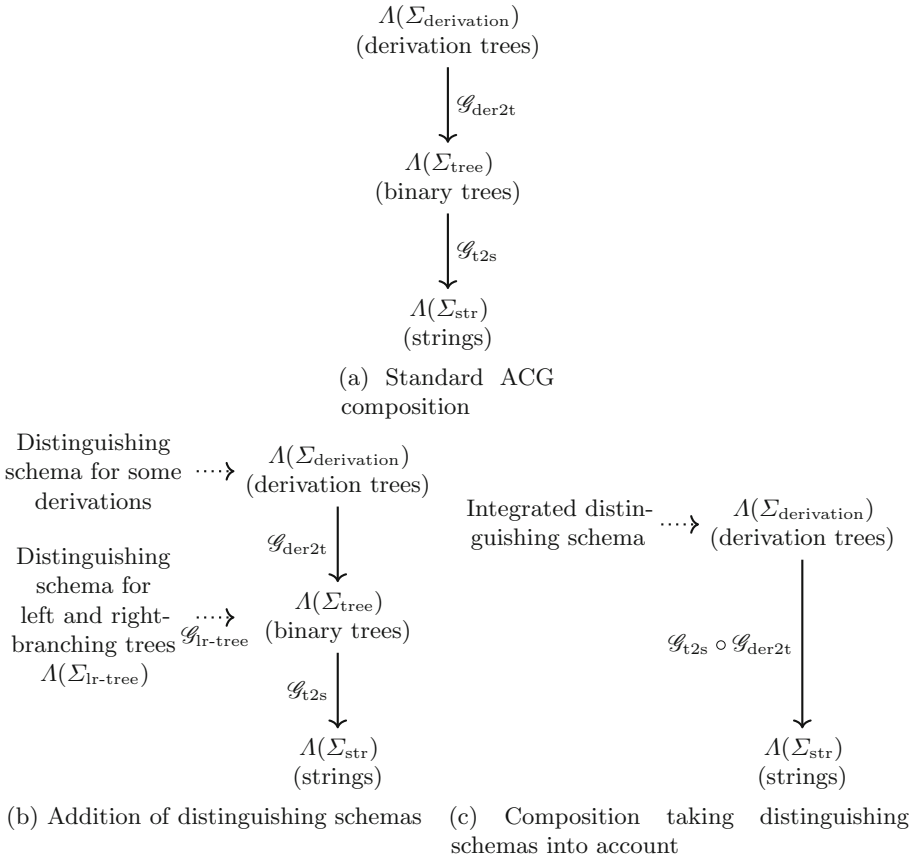


Fig. 1. Composition of ACGs

Section 2 sets the main mathematical notions used in the article and introduces examples to be used through the different sections. It then briefly but for-

mally introduces ACGs. Using these notions, Sect. 2.2 develops the motivations and the directions for extending ACGs. Section 3 defines *overloaded* signatures and *multityped* ACGs, and Sect. 4 addresses the problem of multityped ACG composition.

2 Abstract Categorical Grammars

ACGs were introduced by [4]. With respect to other grammatical formalisms (e.g., CFGs, TAGs, or combinatory categorical grammar, [13]), ACGs have the following discriminating key features:

- An ACG actually defines *two* languages: an *abstract* and an *object* one. The former can generally be seen as the set of admissible parse structures, while the latter can be seen as the set of corresponding surface structures in a broad sense: strings, for instance, but also logical formulas.
- The languages are sets of (linear) λ -terms, that generalize both strings and trees.
- A *lexicon* defines the relation between the abstract and the object language. It is a homomorphism that interprets abstract structures as object ones.
- Because both abstract and object languages are sets of λ -terms, ACGs have built-in support for grammar composition.

2.1 Definitions

Definition 1 (Linear Implicative Type). *Let A be a set of atomic types. The set $\mathcal{T}_A$ of linear implicative types built upon A is inductively defined as follows:*

- if $a \in A$, then $a \in \mathcal{T}_A$;
- if $\alpha, \beta \in \mathcal{T}_A$, then $(\alpha \multimap \beta) \in \mathcal{T}_A$.

In order to save parentheses, we use the usual convention of right association, i.e., we write $\alpha_1 \multimap \alpha_2 \multimap \cdots \alpha_n \multimap \alpha$ for $(\alpha_1 \multimap (\alpha_2 \multimap \cdots (\alpha_n \multimap \alpha) \cdots))$.

Definition 2 (Type Morphism and Relabeling). *Given two sets of atomic types, A and B , a mapping $h : \mathcal{T}_A \rightarrow \mathcal{T}_B$ is called a type homomorphism (or a type substitution) if it satisfies the following condition: $\forall \alpha, \beta \in \mathcal{T}_A, h(\alpha \multimap \beta) = h(\alpha) \multimap h(\beta)$.*

A type substitution mapping atomic types to atomic types is called a type relabeling.

Thus, a type morphism is fully defined by how it maps atomic types to linear implicative types.

Definition 3 (Skeleton of a Type). *Let A be a set of atomic types and let $\mathfrak{A} = \{\square\}$. The skeleton of a type $\alpha \in \mathcal{T}_A$ is $\text{skel}_A(\alpha)$ where $\text{skel}_A : \mathcal{T}_A \rightarrow \mathcal{T}_{\mathfrak{A}}$ is the relabeling mapping atomic types of $\mathcal{T}_A$, i.e., elements of A , to the unique atomic type $\square$.*

For instance, $\text{skel}((NP \multimap S) \multimap NP) = (\Box \multimap \Box) \multimap \Box$.

Definition 4 (Higher-Order Linear Signature). A higher-order linear signature is a triple $\Sigma = \langle A, C, \tau \rangle$, where:

- A is a finite set of atomic types;
- C is a finite set of constants;
- $\tau : C \rightarrow \mathcal{T}_A$ is a function that assigns to each constant in C a linear implicative type in $\mathcal{T}_A$.

Given, a higher-order linear signature Σ , we write A_Σ , C_Σ , and τ_Σ for its respective components.

Definition 5 (Linear λ -Term). Let X be an infinite countable set of variables and Σ be a higher-order linear signature (X and C_Σ disjoint). The set $\Lambda(\Sigma)$ of linear λ -terms built upon Σ is inductively defined as follows:

- if $c \in C_\Sigma$, then $c \in \Lambda(\Sigma)$;
- if $x \in X$, then $x \in \Lambda(\Sigma)$;
- if $x \in X$, $t \in \Lambda(\Sigma)$, and x occurs free in t exactly once, then $(\lambda x.t) \in \Lambda(\Sigma)$;
- if $t, u \in \Lambda(\Sigma)$, and the sets of free variables of t and u are disjoint, then $(tu) \in \Lambda(\Sigma)$.

$\Lambda(\Sigma)$ is provided with the usual notions of α -conversion, β -reduction, and η -reduction [1]. In this paper, all linear λ -terms are identified up to α -conversion. Let t and u be linear λ -terms. We write $t \rightarrow_\beta u$ and $t =_\beta u$ for the relations of β -reduction and β -equivalence, respectively.

Definition 6 (λ -Term Homomorphism and Relabeling). Let Σ_1 and Σ_2 be two signatures. We say that a mapping $h : \Lambda(\Sigma_1) \rightarrow \Lambda(\Sigma_2)$ is a λ -term homomorphism if it satisfies the following conditions: $\forall t, u \in \Lambda(\Sigma_1), x \in X, h(x) = x$, $h(\lambda x.t) = \lambda x.h(t)$, and $h(tu) = h(t)(h(u))$.

A λ -term homomorphism that maps constants of Σ_1 , i.e., elements of C_{Σ_1} , to constants of Σ_2 , i.e., elements of C_{Σ_2} , is called a term relabeling.

Thus, a λ -term homomorphism is fully defined by how it maps constants to linear λ -terms.

Definition 7 (Typing Context and Judgment). Let Σ be a higher-order linear signature Σ . A typing context Γ is a finite multiset of pairs $x : \alpha$ such that x is a variable and $\alpha \in \mathcal{T}_{A_\Sigma}$.

A typing judgment $\Gamma \vdash_\Sigma t : \alpha$ assigns the type $\alpha \in \mathcal{T}_{A_\Sigma}$ to the term $t \in \Lambda(\Sigma)$ in the context Γ if $\Gamma \vdash_\Sigma t : \alpha$ is the root of a derivation tree obeying the rules of Table 1.

We note $t :_\Sigma \alpha$ (or simply $t : \alpha$ when Σ is explicit from the context) for $\vdash_\Sigma t : \alpha$, i.e., when the typing context is empty. We also note $\Gamma \vdash u : \alpha =_\beta \Delta \vdash v : \beta$ for $u =_\beta v$, $\alpha = \beta$ and $\Gamma = \Delta$.

Table 1. Inference rules, given a signature Σ

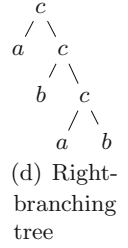
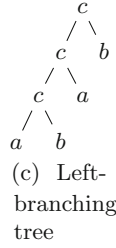
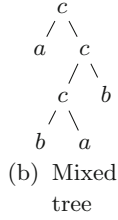
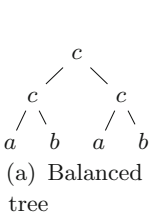
$$\begin{array}{c}
\frac{}{x : \alpha \vdash_{\Sigma} x : \alpha} \text{(var)} \quad \frac{\Gamma, x : \alpha \vdash_{\Sigma} t : \beta}{\Gamma \vdash_{\Sigma} \lambda x. t : \alpha \multimap \beta} \text{(abs)} \\
\frac{}{\vdash_{\Sigma} c : \tau_{\Sigma}(c)} \text{(cst)} \quad \frac{\Gamma_1 \vdash_{\Sigma} t : \alpha \multimap \beta \quad \Gamma_2 \vdash_{\Sigma} u : \alpha}{\Gamma_1, \Gamma_2 \vdash_{\Sigma} t u : \beta} \text{(app)}
\end{array}$$

Example 1 (Tree Signature). We can define a signature Σ_{tree} for binary trees with a binary symbol c and two nullary symbols a and b as follows: $\Sigma_{\text{tree}} = \langle A_{\Sigma_{\text{tree}}}, C_{\Sigma_{\text{tree}}}, \tau_{\Sigma_{\text{tree}}} \rangle$ where: $A_{\Sigma_{\text{tree}}} = \{T\}$; $C_{\Sigma_{\text{tree}}} = \{a, b, c\}$; and $\tau_{\Sigma_{\text{tree}}}(a) = T$, $\tau_{\Sigma_{\text{tree}}}(b) = T$, $\tau_{\Sigma_{\text{tree}}}(c) = T \multimap T \multimap T$.

Terms of $\Lambda(\Sigma_{\text{tree}})$ of type T include, for instance, the ones defined in Eqs. (1a–1d) and which correspond to the trees of Fig. 2.

$$t_{2a} = c(cab)(cab) \quad (1a) \quad t_{2c} = c(c(cab)a)b \quad (1c)$$

$$t_{2b} = ca(c(cba)b) \quad (1b) \quad t_{2d} = ca(cb(cab)) \quad (1d)$$

**Fig. 2.** Terms of $\Lambda(\Sigma_{\text{tree}})$

Example 2 (String Signature). We define a signature Σ_{str} for strings using the a and b string symbols as follows: $\Sigma_{\text{str}} = \langle A_{\Sigma_{\text{str}}}, C_{\Sigma_{\text{str}}}, \tau_{\Sigma_{\text{str}}} \rangle$ where:

- $A_{\Sigma_{\text{str}}} = \{\sigma\}$ will allow us to define the complex type of strings $\sigma = o \multimap o$ for which concatenation is functional composition (the λ -term $+$ = $\lambda f g. \lambda z. f(gz) : \sigma \multimap \sigma \multimap \sigma$) and the empty string is the identity ($\lambda x. x : \sigma$)
- $C_{\Sigma_{\text{str}}} = \{a, b\}$
- $\tau_{\Sigma_{\text{str}}}(a) = \tau_{\Sigma_{\text{str}}}(b) = \sigma$

For instance, the string ab corresponds to the term $\lambda z. a(bz)$ of $\Lambda(\Sigma_{\text{str}})$; and it duplication $abab$ is $\lambda z. a(bz) + \lambda z. a(bz) = \lambda z. a(b(a(bz))) (= a + b + a + b)$.

Definition 8 (Simple Lexicon and Relabeling). Let $\Sigma_1 = \langle A_1, C_1, \tau_1 \rangle$ and $\Sigma_2 = \langle A_2, C_2, \tau_2 \rangle$ be two higher-order signatures. A lexicon $\mathcal{L} : \Sigma_1 \rightarrow \Sigma_2$ is a pair $\langle F, G \rangle$ such that:

- $F : A_1 \rightarrow \mathcal{T}_{A_2}$ is a type homomorphism that interprets linear implicative types built upon Σ_1 as linear implicative types built upon Σ_2 ;
- $G : C_1 \rightarrow \Lambda(\Sigma_2)$ is a term homomorphism that interprets linear λ -terms built upon Σ_1 as linear λ -terms built upon Σ_2 ;
- the interpretation functions are compatible with the typing relation, i.e., for any $c \in C_1$, $\vdash_{\Sigma_2} G(c) : F(\tau_1(c))$ is derivable.

When F and G are respectively type and term relabelings, the lexicon is called a relabeling.

In the sequel, given such a lexicon $\mathcal{L} = \langle F, G \rangle$, $\mathcal{L}(a)$ will stand for either $F(a)$ or $G(a)$, according to the context and $a := b$ will stand for $\mathcal{L}(a) = b$. We also write $\mathcal{L}(x_1 : \alpha_1, \dots, x_n : \alpha_n)$ for the context $x_1 : \mathcal{L}(\alpha_1), \dots, x_n : \mathcal{L}(\alpha_n)$, and $\mathcal{L}(\Gamma \vdash_{\Sigma_1} u : \alpha)$ for $\mathcal{L}(\Gamma) \vdash_{\Sigma_2} \mathcal{L}(u) : \mathcal{L}(\alpha)$.

Lemma 1. Let $\mathcal{R}$ be a relabeling from Σ_1 to Σ_2 . Let $t \in \Lambda(\Sigma_2)$ and $N \in \Lambda(\Sigma_1)$ such that $\mathcal{R}(N) =_{\beta} t$. There exists a unique $M =_{\beta} N$ such that $\mathcal{R}(M) = t$.

Proof. The β -equivalence boils down to the two cases where either $\mathcal{R}(N)$ or t is a redex and reduction is performed in one step.

The first case is $\mathcal{R}(N) = (\lambda x.u)v \rightarrow_{\beta} t$. Since relabelings preserve structures, N has the form $(\lambda x.N_0)N_1$. Thus $M = N_0[x := N_1]$.

The second case is $t = (\lambda x.u)v \rightarrow_{\beta} \mathcal{R}(N)$. Since relabelings preserve structures, N has the form $N_0[x := N_1]$. Thus $M = (\lambda x.N_0)N_1$.

Definition 9 (Abstract Categorical Grammar). An abstract categorical grammar is a quadruple, $\mathcal{G} = \langle \Sigma_1, \Sigma_2, \mathcal{L}, S \rangle$, where:

- Σ_1 and Σ_2 are two higher-order linear signatures. They are called the abstract vocabulary and the object vocabulary, respectively. Terms of $\Lambda(\Sigma_1)$ (resp. $\Lambda(\Sigma_2)$) are called abstract terms (resp. object terms).
- $\mathcal{L} : \Sigma_1 \rightarrow \Sigma_2$ is a lexicon from the abstract vocabulary to the object vocabulary.
- S is a set of distinguished abstract types of the abstract vocabulary.

Example 3 (Tree Interpretation). Using the signatures defined in Examples 1 and 2, we can now define an ACG $\mathcal{G}_{\text{t2s}} = \langle \Sigma_{\text{tree}}, \Sigma_{\text{str}}, \mathcal{L}_{\text{t2s}}, \{T\} \rangle$ where

- $\mathcal{L}_{\text{t2s}}(T) = \sigma$, i.e., a tree (a term of $\Lambda(\Sigma_{\text{tree}})$) is interpreted as string.
- $\mathcal{L}_{\text{t2s}}(a) = a$ and $\mathcal{L}_{\text{t2s}}(b) = b$.
- $\mathcal{L}_{\text{t2s}}(c) = \lambda l.r.l + r$, where $+$ is the concatenation of strings.

We then check that $\mathcal{L}_{\text{t2s}}(t_{2a}) = \mathcal{L}_{\text{t2s}}(t_{2b}) = \mathcal{L}_{\text{t2s}}(t_{2c}) = \mathcal{L}_{\text{t2s}}(t_{2d}) = a + b + a + b$.

Definition 10 (Abstract and Object Languages). Let $\mathcal{G} = \langle \Sigma_1, \Sigma_2, \mathcal{L}, S \rangle$ be an ACG. The abstract language of $\mathcal{G}$ is $\mathcal{A}(\mathcal{G}) = \{t \in \Lambda(\Sigma_1) \mid \exists \alpha \in S, \vdash_{\Sigma_1} t : \alpha\}$. The object language of $\mathcal{G}$ is $\mathcal{O}(\mathcal{G}) = \{u \in \Lambda(\Sigma_2) \mid \exists t \in \mathcal{A}(\mathcal{G}), u = \mathcal{L}(t)\}$.

Given an ACG $\mathcal{G} = \langle \Sigma_1, \Sigma_2, \mathcal{L}, S \rangle$, the ACG parsing of a term u of $\Lambda(\Sigma_2)$ amounts to finding $t \in \mathcal{A}(\mathcal{G})$ such that $\mathcal{L}(t) = u$.

Example 4 (ACG Composition). The ACG $\mathcal{G}_{t2s}$ of Example 3 defines the object language $\mathcal{O}(\mathcal{G}_{t2s}) = (a|b)^+$.

Regarding the abstract language $\mathcal{O}(\mathcal{G}_{t2s})$, it contains any binary tree, such as the ones exemplified in Fig. 2. For instance, $a + b + a + b$ has five antecedents by $\mathcal{L}_{t2s}$ (among which the terms corresponding to the trees of Fig. 2).

Let us now assume we are only interested in left or right-branching trees. We are going to define an ACG $\mathcal{G}_{lr-tree} = \langle \Sigma_{lr-tree}, \Sigma_{tree}, \mathcal{L}_{lr-tree}, S \rangle$ in which terms of the abstract language are left or right-branching trees, and compose it with $\mathcal{G}_{t2s}$ so that we do not need to redefine the string interpretation.

We define a signature $\Sigma_{lr-tree}$ with $A_{\Sigma_{lr-tree}} = \{T_u, T_l, T_r\}$ and $C_{\Sigma_{lr-tree}} = \{a_u, b_u, a_l, b_l, a_r, b_r, c_l, c_r\}$. The type assignment $\tau_{\Sigma_{lr-tree}}$ is such that: $\tau_{\Sigma_{lr-tree}}(v_w) = T_w$ for $v \in \{a, b\}$ and $w \in \{u, l, r\}$, while $\tau_{\Sigma_{lr-tree}}(c_l) = T_l \multimap T_u \multimap T_l$ and $\tau_{\Sigma_{lr-tree}}(c_r) = T_u \multimap T_r \multimap T_r$.

The definition of $\mathcal{L}_{lr-tree} : A_{\Sigma_{lr-tree}} \rightarrow \Lambda(\Sigma_{tree})$ is straightforward: $\mathcal{L}_{lr-tree}(T_v) = T$ for any $v \in \{u, l, r\}$ and $\mathcal{L}_{lr-tree}(x_v) = x$ for any $x \in \{a, b, c\}$ and $v \in \{u, l, r\}$. Terms of $\mathcal{A}(\mathcal{G}_{lr-tree})$ (when $S = \{T_l, T_r\}$) then only consists of left-branching or right-branching trees, and they can be interpreted as strings by $\mathcal{L}_{t2s} \circ \mathcal{L}_{lr-tree}$. For instance, $a + b + a + b$ now only has two antecedents by $\mathcal{L}_{t2s} \circ \mathcal{L}_{lr-tree}$: the terms of $\mathcal{A}(\mathcal{G}_{\Sigma_{lr-tree}})$ that are interpreted by $\mathcal{L}_{lr-tree}$ as terms corresponding to the trees of Figs. 2c and 2d.

2.2 Distinguishing Between Structures

Albeit through quite a simple lexicon, Example 4 illustrates the composition ability of ACGs. The lexicon $\mathcal{L}_{lr-tree}$ is actually an example of a *relabeling*. From the Σ_{tree} point of view, it means that c is not considered with the full generality of the $T \multimap T \multimap T$ type, but either with the more specific type $T_l \multimap T_u \multimap T_l$ or the more specific type $T_u \multimap T_r \multimap T_r$. On the other hand, its interpretation by $\mathcal{L}_{t2s}$ remains unchanged.

We decided to restrict the admissible parse structures to left-branching and right-branching trees. However, we eventually are interested in *preferences* and *ranking*, not only in membership. So we want to distinguish left and right branching trees (and possibly give them preference), but we don't want to exclude some other tree structures. For instance, we could add a constant c'_l with type $T_r \multimap T_u \multimap T_l$, and a constant c'_r with type $T_u \multimap T_l \multimap T_r$. This is reminiscent to adding states in tree automata to distinguish transitions triggered by a same symbol. Because all these constants are interpreted as c , we could equivalently assign c this set of types and possibly add a weight to each of its typings, pretty much as weights can be added to transition of tree automata [6, 9].

Note that, although ACGs in the examples feature abstract constants with parameters of atomic type, the approach also extends to higher-order signatures where abstract constants may have functions as parameters.

However, we first have to precisely define multityped ACGs, and check what the implications on ACG operations such as composition are. In particular, the relabelings can introduce distinction between structures at any level when composing ACGs. We therefore expect them to combine in order to define an equivalent distinction for the resulting composed ACG.

3 Multityped Abstract Categorical Grammars

3.1 Overloaded Signatures

The main differences between simple signatures and overloaded ones is the relational nature of the type assignment, and the addition of the skeleton constraint.

Definition 11 (Overloaded Signature). *An overloaded signature is a triple $\Pi = \langle A, C, \tau^+ \rangle$, where:*

- A is a finite set of atomic types;
- C is a finite set of constants;
- $\tau^+ : C \rightarrow \mathcal{P}_f(\mathcal{T}_A)$ is a function that assigns to each constant in C a non-empty finite set of linear implicative types in $\mathcal{T}_A$;
- a constant's types share a skeleton: $\forall c \in C, \forall \alpha, \beta \in \tau^+(c), \text{skel}(\alpha) = \text{skel}(\beta)$

Given a signature $\Pi = \langle A, C, \tau^+ \rangle$, we write $\mathcal{T}_\Pi$ for $\mathcal{T}_A$ and $\Lambda(\Pi)$ for $\Lambda(C)$.

Example 5. We can now formally define the overloaded signature Π mentioned in Sect. 2.2 as a follow up to Example 4. $\Pi = \langle A, C, \tau^+ \rangle$ where: $A = \{T_u, T_l, T_r\}$; $C = \{a, b, c\}$; τ^+ is given by Table 2.

Note that Π satisfies the skeleton constraint. It would not be the case, for instance, if a was additionally given an implicative type such as $T_l \multimap T_r$.

Table 2. The overloaded signature Π

$$a : \left\{ \begin{array}{l} T_u, \\ T_l, \\ T_r \end{array} \right\} \qquad b : \left\{ \begin{array}{l} T_u, \\ T_l, \\ T_r \end{array} \right\} \qquad c : \left\{ \begin{array}{l} T_l \multimap T_u \multimap T_l, \\ T_u \multimap T_r \multimap T_r \end{array} \right\}$$

Typing judgments are still defined by induction on the typing rules featured in Table 1, but for the typing rule of constants which is slightly amended as follows:

$$\frac{\alpha \in \tau_\Pi(c)}{\vdash_\Pi c : \alpha} (\text{cst}+).$$

It can be shown (see Sect. 3.3) that to any overloaded signature Π corresponds a triple $(\Sigma_\Pi^\mathcal{T}, \Sigma_\Pi^A, \mathcal{R}_\Pi)$ where $\Sigma_\Pi^\mathcal{T}$ (the *type carrier* of Π) and Σ_Π^A (the *term carrier* of Π) are simple signatures and $\mathcal{R}_\Pi : \Sigma_\Pi^\mathcal{T} \rightarrow \Sigma_\Pi^A$ is a relabeling, such that $\Gamma \vdash_\Pi M : \alpha$ iff there exists $t \in \Lambda(\Sigma_\Pi^\mathcal{T})$ such that $\mathcal{R}_\Pi(t) = M$ and $\Gamma \vdash_{\Sigma_\Pi^\mathcal{T}} t : \alpha$ (see Lemma 2). The reverse also holds (see Lemma 3).

3.2 Lexicons and Multityped ACGs

We now want to define a lexicon $\mathcal{L}^+$ between an overloaded signature Π and a simple signature Σ . A guiding principle is to refer to the carriers of Π and to the relabeling $\mathcal{R}_\Pi : \Sigma_\Pi^\mathcal{T} \rightarrow \Sigma_\Pi^\Lambda$ that simulates Π (see Lemma 2). We can then consider the simple lexicons $\mathcal{L}^\mathcal{T} : \Sigma_\Pi^\mathcal{T} \rightarrow \Sigma$ and $\mathcal{L}^\Lambda : \Sigma_\Pi^\Lambda \rightarrow \Sigma$. These two lexicons respectively act on the linear implicative types, and on the linear λ -terms built upon Π .

To preserve typing judgments, we require $\mathcal{L}^\mathcal{T}$ and $\mathcal{L}^\Lambda$ to comply with the diagram in Fig. 3a: for any $\Gamma \vdash_{\Sigma_\Pi^\mathcal{T}} t : \alpha$, $\mathcal{L}^\mathcal{T}(\Gamma \vdash_{\Sigma_\Pi^\mathcal{T}} t : \alpha) =_\beta \mathcal{L}^\Lambda \circ \mathcal{R}_\Pi(\Gamma \vdash_{\Sigma_\Pi^\mathcal{T}} t : \alpha)$. From Sect. 3.3, this is equivalent to requiring that for any $\Gamma \vdash_\Pi M : \alpha$, there exists $t \in \Lambda(\Sigma_\Pi^\mathcal{T})$ such that $\mathcal{R}_\Pi(t) = M$ and $\mathcal{L}^\mathcal{T}(\Gamma) \vdash_\Sigma \mathcal{L}^\mathcal{T}(t) : \mathcal{L}^\mathcal{T}(\alpha) =_\beta \mathcal{L}^\Lambda([\Gamma]_{\sim_\Pi}) \vdash_\Sigma \mathcal{L}^\Lambda(M) : \mathcal{L}^\Lambda([\alpha]_{\sim_\Pi})$ where $\Gamma = x_1 : \alpha_1, \dots, x_n : \alpha_n$ and $[\Gamma]_{\sim_\Pi} = x_1 : [\alpha_1]_{\sim_\Pi}, \dots, x_n : [\alpha_n]_{\sim_\Pi}$.

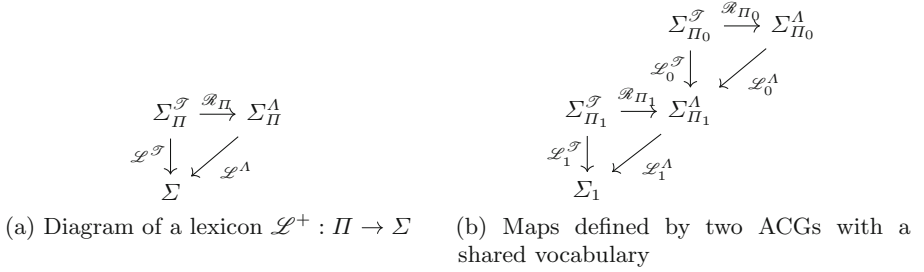


Fig. 3. Diagrams describing mACGs

Thus, we define the lexicon $\mathcal{L}^+$ from Π to Σ such that $\forall \alpha \in \mathcal{T}_\Pi, \mathcal{L}^+(\alpha) = \mathcal{L}^\mathcal{T}(\alpha)$ and $\forall M \in \Lambda(\Pi), \mathcal{L}^+(M) = \mathcal{L}^\Lambda(M)$. In particular, $\mathcal{L}^+$ maps Π -equivalent (atomic) types to the same type.

Definition 12 (Lexicon). Let $\Pi = \langle A, C, \tau^+ \rangle$ be an overloaded signature and Σ a simple signature. A lexicon from Π to Σ is a pair $\mathcal{L}^+ = \langle F, G \rangle$ such that:

- $F : \mathcal{T}_\Pi \rightarrow \mathcal{T}_\Sigma$ is a type homomorphism that interprets linear implicative types built upon Π as linear implicative types built upon Σ ;
- $G : \Lambda(\Pi) \rightarrow \Lambda(\Sigma)$ is a term homomorphism that interprets linear λ -terms built upon Π as linear λ -terms built upon Σ ;
- the interpretation homomorphisms are compatible with the typing relation: $\forall c \in C, \forall \alpha \in \tau^+(c), \vdash_\Pi G(c) : F(\alpha)$
- the type homomorphism maps the skeleton equivalence to the equality: $\forall a, b \in A, a \sim_\Pi b \Rightarrow F(a) = F(b)$

We write $\mathcal{L}^+(\alpha)$ for $F(\alpha)$, $\mathcal{L}^+(M)$ for $G(M)$, $\mathcal{L}^+(\Gamma)$ for $F(\Gamma)$, and $\mathcal{L}^+(\Gamma \vdash_\Pi M : \alpha)$ for $\mathcal{L}^+(\Gamma) \vdash_\Sigma \mathcal{L}^+(M) : \mathcal{L}^+(\alpha)$.

As for simple lexicons (see Definition 8), it is sufficient to define a lexicon on the domain signature.

Conversely, given a lexicon $\mathcal{L}^+ : \Pi \rightarrow \Sigma$, one defines $\mathcal{L}^\mathcal{T} : \Sigma_\Pi^\mathcal{T} \rightarrow \Sigma$ the *type lexicon* of $\mathcal{L}^+$ and $\mathcal{L}^A : \Sigma_\Pi^A \rightarrow \Sigma$ the *term lexicon* of $\mathcal{L}^+$ as follows:

$$\begin{aligned} \mathcal{L}^\mathcal{T}(t) &= \mathcal{L}^+ \circ \mathcal{R}_\Pi(t) \text{ if } t \in \Lambda(\Sigma_\Pi^\mathcal{T}) & \mathcal{L}^\mathcal{T}(\alpha) &= \mathcal{L}^+(\alpha) \text{ if } \alpha \in \mathcal{T}_{\Sigma_\Pi^\mathcal{T}} \\ \mathcal{L}^A(M) &= \mathcal{L}^+(M) \text{ if } M \in \Lambda(\Sigma_\Pi^A) & \mathcal{L}^A([\alpha]_{\sim_\Pi}) &= \mathcal{L}^+(\alpha) \text{ if } [\alpha]_{\sim_\Pi} \in \mathcal{T}_{\Sigma_\Pi^A} \end{aligned}$$

Then: $\mathcal{L}^+$ and $\mathcal{L}^\mathcal{T}$ act the same on linear implicative types; $\mathcal{L}^+$ and $\mathcal{L}^A$ act the same on linear λ -terms; and $\mathcal{L}^\mathcal{T}$ and $\mathcal{L}^A$ comply with the Fig. 3a.

Example 6 (Lexicon from an Overloaded Signature). Using the overloaded signature of Table 2 and the simple signature Σ_{str} defined in Example 2, we can define the lexicon:

$$T_u, T_l, T_r := \sigma \qquad a := a \qquad b := b \qquad c := \lambda l.r.l + r$$

Because a is assigned the three atomic types T_u , T_l , and T_r these types are skeleton equivalent and are required to be interpreted by the same type of $\mathcal{T}_{A_{\Sigma_{\text{str}}}}$.

Definition 13 (Multityped ACG). A multityped abstract categorial grammar (mACG) is a quadruple $\mathcal{G}^+ = \langle \Pi, \Sigma, \mathcal{L}^+, S \rangle$ where:

- Π is an overloaded signature;
- Σ is a simple signature;
- $\mathcal{L}^+ : \Pi \rightarrow \Sigma$ is a lexicon from Π to Σ ;
- S is a set of distinguished abstract types of $\mathcal{T}_\Pi$.

Definition 14 (Abstract and Object Languages of a Multityped ACG). Let $\mathcal{G}^+ = \langle \Pi, \Sigma, \mathcal{L}^+, S \rangle$ be an ACG. The abstract language of $\mathcal{G}^+$ is $\mathcal{A}(\mathcal{G}^+)$, where $\mathcal{A}(\mathcal{G}^+) = \{t \in \Lambda(\Pi) \mid \exists \alpha \in S, \vdash_\Pi t : \alpha\}$.

The object language of $\mathcal{G}^+$ is $\mathcal{O}(\mathcal{G}^+) = \{u \in \Lambda(\Sigma) \mid \exists t \in \mathcal{A}(\mathcal{G}^+), u = \mathcal{L}^+(t)\}$.

Example 7 (Multityped ACG). We can now define the multityped ACG $\mathcal{G}^+ = \langle \Pi, \Sigma_{\text{str}}, \mathcal{L}^+, \{T_l, T_r\} \rangle$ from the lexicon of Example 6. As in Example 3, $\mathcal{L}^+(t_{2a}) = \mathcal{L}^+(t_{2b}) = \mathcal{L}^+(t_{2c}) = \mathcal{L}^+(t_{2d}) = a + b + a + b$ (the t_i are defined in Eqs. 1a–1d). However, only $t_{2c} = c(c(cab)a)b$ and $t_{2d} = ca(cb(cab))$ can be given a type in Π (T_l and T_r , resp.). They both belong to $\mathcal{A}(\mathcal{G}^+)$.

3.3 Multityped ACGs as Commutative Diagrams

This section shows that overloaded signatures are actually as expressive as simple signatures. Indeed, as illustrated in Fig. 3a, an overloaded signature Π can be simulated with two simple ones linked with a relabeling $\mathcal{R}_\Pi : \Sigma_\Pi^\mathcal{T} \rightarrow \Sigma_\Pi^A$. As a result, overloading preserves properties of simple higher-order linear signatures.

We identify types that can occur at the same position in terms of $\Lambda(\Pi)$.

Definition 15 (Skeleton Equivalence). *Let Π be an overloaded signature. Then $\sim_\Pi$ is the smallest equivalence relation such that:*

- *if there exists a constant $c \in \Pi$ such that $\alpha, \beta \in \tau^+(c)$, then $\alpha \sim_\Pi \beta$;*
- *if $\alpha_0 \multimap \alpha_1 \sim_\Pi \beta_0 \multimap \beta_1$, then $\alpha_0 \sim_\Pi \beta_0$ and $\alpha_1 \sim_\Pi \beta_1$;*
- *if $\alpha_0 \sim_\Pi \beta_0$ and $\alpha_1 \sim_\Pi \beta_1$, then $\alpha_0 \multimap \alpha_1 \sim_\Pi \beta_0 \multimap \beta_1$;*

We say α and β are Π -equivalent for $\alpha \sim_\Pi \beta$.

Note that skeleton equivalence implies skeleton equality. Then, we define the relabeling, its domain and its codomain.

Definition 16 (Carriers). *Let $\Pi = \langle A, C, \tau^+ \rangle$ be an overloaded signature.*

- *The type carrier $\Sigma_\Pi^\mathcal{T} = \langle A, C', \tau_0 \rangle$ of Π is the simple signature where $C' = \bigsqcup_{c \in C} \{c^\alpha \mid \alpha \in \tau^+(c)\}$ and $\tau_0 : c^\alpha \mapsto \alpha$.*
- *The term carrier $\Sigma_\Pi^A = \langle A/\sim_\Pi, C, \tau_1 \rangle$ of Π is the simple signature where $\tau_1 : c \mapsto [\alpha]_{\sim_\Pi}$ and $\alpha \in \tau_c^+$ is any type of c ;*
- *The carrier relabeling $\mathcal{R}_\Pi : \Sigma_\Pi^\mathcal{T} \rightarrow \Sigma_\Pi^A$ of Π is the relabeling such that $\forall c^\alpha \in C', \mathcal{R}_\Pi(c^\alpha) = c$ and $\forall a \in A, \mathcal{R}_\Pi(a) = [a]_{\sim_\Pi}$.*

The domain $\Sigma_\Pi^\mathcal{T}$ of $\mathcal{R}_\Pi$ is called the type carrier of Π , for it builds the same linear implicative types as Π . Moreover, to each constant c of Π corresponds $|\tau^+(c)|$ constants $c^\alpha : \alpha$ in $\Sigma_\Pi^\mathcal{T}$, one for each $\alpha \in \tau^+(c)$. Therefore, type-checking a term $u : \beta \in \Lambda(\Pi)$ corresponds to checking whether there is some $t : \beta \in \Lambda(\Sigma_\Pi^\mathcal{T})$ such that $\mathcal{R}_\Pi(t) = u$, as Lemma 2 shows.

The codomain Σ_Π^A of $\mathcal{R}_\Pi$ is called the term carrier of Π , for it builds the same linear λ -terms as Π . So that Σ_Π^A is a simple signature, its set of linear implicative types is the quotient of $\mathcal{T}_\Pi$ by the skeleton equivalence $\sim_\Pi$.

As for the carrier relabeling $\mathcal{R}_\Pi$, it maps the duplicated constants c^α to their generators c and the types α to their equivalence classes $[\alpha]_{\sim_\Pi}$.

Lemma 2 (Carriers). *Let $\Pi = \langle A, C, \tau^+ \rangle$ be an overloaded signature. Let $\Sigma_\Pi^\mathcal{T} = \langle A, C', \tau_0 \rangle$ be its type carrier, $\Sigma_\Pi^A = \langle A/\sim_\Pi, C, \tau_1 \rangle$ be its term carrier and $\mathcal{R}_\Pi : \Sigma_\Pi^\mathcal{T} \rightarrow \Sigma_\Pi^A$ be its carrier relabeling.*

Then $\Gamma \vdash_\Pi M : \alpha$ iff there exists $t \in \Lambda(C')$ such that $\mathcal{R}_\Pi(t) = M$ and $\Gamma \vdash_{\Sigma_\Pi^\mathcal{T}} t : \alpha$.

Lemma 3 proves that, in turn, any relabeling defines an overloaded signature satisfying a similar property.

Lemma 3. ($\mathcal{R}$ -Overloaded Signature). *Let $\Sigma_0 = \langle A_0, C_0, \tau_0 \rangle$ and $\Sigma_1 = \langle A_1, C_1, \tau_1 \rangle$ be simple signatures and $\mathcal{R} : \Sigma_0 \rightarrow \Sigma_1$ a relabeling. Define the signature overloaded with $\mathcal{R}$ (or $\mathcal{R}$ -overloaded signature) as $\Pi_\mathcal{R} = \langle A_0, C_1, \tau^+ \rangle$ the overloaded signature with $\tau^+ : c \mapsto \tau_0(\mathcal{R}^{-1}(c))$.*

Then $\Gamma \vdash_{\Pi_\mathcal{R}} M : \alpha$ iff there exists $t \in \Lambda(C_0)$ such that $\mathcal{R}(t) = M$ and $\Gamma \vdash_{\Sigma_0} t : \alpha$.

Notice the signature overloaded with the carrier relabeling of Π is Π , i.e., for any overloaded signature Π , $\Pi_{\mathcal{R}\Pi} = \Pi$. Conversely, given the simple signature Σ_0 and Σ_1 , and the relabeling $\mathcal{R} : \Sigma_0 \rightarrow \Sigma_1$, we have that Σ_0 and $\Sigma_{\Pi_{\mathcal{R}}}$ are isomorphic, Σ_1 and $\Sigma_{\Pi_{\mathcal{R}}}$ are isomorphic, and $\mathcal{R}$ and $\mathcal{R}_{\Pi_{\mathcal{R}}}$ are isomorphic.

4 The Composition Problem

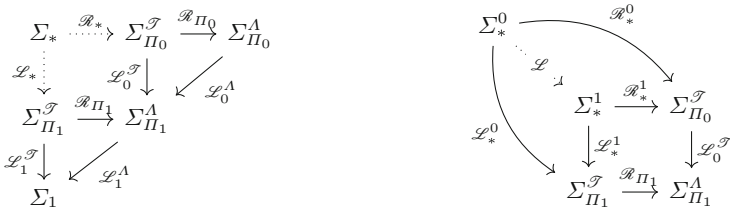
Lexicons of multityped ACGs map overloaded signatures to simple signatures. Therefore, composition of ACGs cannot be as simple as lexicon composition since their domains and codomains are of different kinds. We show that a well-behaved composition for mACGs can be defined using a pullback in a specific construction. This pullback turns out to be the construction provided by [8] to show ACGs are closed by intersection with regular languages. This section reminds this construction and shows that it is indeed a pullback.

4.1 Composition as Pullback

In this section, consider two mACGs: $\mathcal{G}_0^+ = \langle \Pi_0, \Sigma_0, \mathcal{L}_0^+, S_0 \rangle$ on the one hand and $\mathcal{G}_1^+ = \langle \Pi_1, \Sigma_1, \mathcal{L}_1^+, S_1 \rangle$ on the other hand. We aim to provide a well-behaved definition of their composition $\mathcal{G}_1^+ \circ \mathcal{G}_0^+$.

Figure 3b illustrates the composition problem. We look to extract a triangle defining a lexicon, like in Fig. 4a, from this diagram. The term lexicon of this triangle should be $\mathcal{L}_1^\Lambda \circ \mathcal{L}_0^\Lambda$.

A possibility is to find a simple signature Σ_* which is mapped to both type carriers $\Sigma_{\Pi_0}^\mathcal{T}$ and $\Sigma_{\Pi_1}^\mathcal{T}$. A lexicon $\mathcal{L}_* : \Sigma_* \rightarrow \Sigma_{\Pi_1}^\mathcal{T}$ means the overloading of Π_1 is accounted for in Σ_* . And, with a relabeling $\mathcal{R}_* : \Sigma_* \rightarrow \Sigma_{\Pi_0}^\mathcal{T}$, Σ_* is a finer type carrier than $\Sigma_{\Pi_0}^\mathcal{T}$. For Σ_* , $\mathcal{L}_*$, and $\mathcal{R}_*$ are not given but to be found, Fig. 4a depicts them with dotted arrows. Thus, we obtain a new lexicon $\mathcal{L}^+$ from the $(\mathcal{R}_{\Pi_0} \circ \mathcal{R}_*)$ -overloaded signature (see Lemma 3) to the simple signature Σ_1 : the type lexicon of $\mathcal{L}^+$ is $\mathcal{L}_1^\mathcal{T} \circ \mathcal{L}_*$, and the term lexicon of $\mathcal{L}^+$ is $\mathcal{L}_1^\Lambda \circ \mathcal{L}_0^\Lambda$.



(a) A solution to the composition problem with Σ_* , the lexicon $\mathcal{L}^\mathcal{T} = \mathcal{L}_1^\mathcal{T} \circ \mathcal{L}_*$, and the relabeling $\mathcal{R} = \mathcal{R}_{\Pi_0} \circ \mathcal{R}_*$ (b) A solution $\langle \Sigma_*^1, \mathcal{L}_*^1, \mathcal{R}_*^1 \rangle$ is better than a solution $\langle \Sigma_*^0, \mathcal{L}_*^0, \mathcal{R}_*^0 \rangle$ if there exists a unique $\mathcal{L} : \Sigma_*^0 \rightarrow \Sigma_*^1$

Fig. 4. Solutions to the composition problem

Now, a given situation of our composition problem may lead to several solutions for Σ_* , $\mathcal{L}_*$, and $\mathcal{R}_*$. To discriminate them, we say the solution triple $\langle \Sigma_*^1, \mathcal{L}_*^1, \mathcal{R}_*^1 \rangle$ is a better solution than the solution triple $\langle \Sigma_*^0, \mathcal{L}_*^0, \mathcal{R}_*^0 \rangle$ if there exists a unique simple lexicon $\mathcal{L} : \Sigma_*^0 \rightarrow \Sigma_*^1$ such that $\mathcal{L}_*^0 =_\beta \mathcal{L}_*^1 \circ \mathcal{L}$ and $\mathcal{R}_*^0 =_\beta \mathcal{R}_*^1 \circ \mathcal{L}$. See Fig. 4b for the corresponding diagram.

The best solution of the composition problem in Fig. 4a with respect to the order in Fig. 4b is actually called *the pullback*. Pullbacks are universal constructions in category theory [12]. The category at hand here is the one where objects are the (simple) signatures and arrows are the lexicons.

However, the pullback as the best solution to the composition problem in Fig. 4a comes with two issues. First, the pullback may not exist in some situations. Second, the pullback is actually the minimum for the order in Fig. 4a when the $\mathcal{R}_*^i$ are not restricted to relabelings (the arrows of our category are simple lexicons). As such, there is no reason *a priori* for $\mathcal{R}_*$ to be a relabeling in the pullback triple $\langle \Sigma_*, \mathcal{L}_*, \mathcal{R}_* \rangle$.

In Sect. 4.2, we revisit a signature construction by [8]. We name it *fiber product signature*, for this construction is built upon the fiber product of atomic types and linear implicative types, and the fiber product of constants and linear λ -terms. We show the fiber product signature is not only a solution to Fig. 4a but, furthermore, the pullback Σ_* , and the associated $\mathcal{R}_*$ is indeed a relabeling.

Moreover, [8] shows the associated $\mathcal{L}_*$ generates the object language of $\mathcal{G}_0^+$, up to the relabeling $\mathcal{R}_{\Pi_1} : \mathcal{O}(\langle \Sigma_*, \Sigma_{\Pi_1}^\mathcal{T}, \mathcal{L}_*, \mathcal{R}_*^{-1}(S_0) \rangle) = \mathcal{R}_{\Pi_1}^{-1}(\mathcal{O}(\mathcal{G}_0^+))$.

Thus, defining the lexicon $\mathcal{L}^+ = \langle \mathcal{L}_1^\mathcal{T} \circ \mathcal{L}_*, \mathcal{L}_1^A \circ \mathcal{L}_0^A \rangle$, the multityped ACGs $\mathcal{G}_0^+$ and $\mathcal{G}_1^+$ compose into the multityped ACG $\mathcal{G}_1^+ \circ \mathcal{G}_0^+ = \langle \Sigma_*, \Sigma_1, \mathcal{L}^+, \mathcal{R}_*^{-1}(S_0) \rangle$ (here Σ_* is seen as the $(\mathcal{R}_{\Pi_0} \circ \mathcal{R}_*)$ -overloaded signature; see Lemma 3).

Notice that, similarly to simple ACGs, the object language of the composition is the image of the abstract language under the composition:

$$\mathcal{O}(\mathcal{G}_1^+ \circ \mathcal{G}_0^+) = \mathcal{L}_1^+(\mathcal{O}(\mathcal{G}_0^+)) = \mathcal{L}_1^+ \circ \mathcal{L}_0^+(\mathcal{A}(\mathcal{G}_0^+))$$

Example 8 (Composition of Multityped ACGs). Example 7 introduced a multityped ACG for which admissible abstract structures are left and right-branching trees. We now want to enforce these trees to be the derived trees of a tree-adjointing grammar. To this end, we follow the construction provided by [5] (not detailed here) and define $\Sigma_{\text{derivation}}$ with the atomic types $\{T_a, T\}$ (T_a stands for the place where adjunction is possible) and the constants of Table 3.

We then define the ACG $\mathcal{G}_{\text{der2t}} = \langle \Sigma_{\text{derivation}}, \Sigma_{\text{tree}}, \mathcal{L}_{\text{der2t}}, \{T\} \rangle$ where $\mathcal{L}_{\text{der2t}}$ is the lexicon of Table 3 (with the corresponding initial and auxiliary trees of the equivalent tree-adjointing grammar on the right-hand side). The (usual) composition of ACGs $\mathcal{G}_{\text{t2s}} \circ \mathcal{G}_{\text{der2t}}$ generates binary trees that are, in addition, the derived tree of a given tree-adjointing grammar.

For instance, the term $t_{2c} = c(c(cab)a)b$ corresponding to the tree of Fig. 2c has two antecedents by $\mathcal{G}_{\text{der2t}}$: $b_l I(a_r(aI)b_u)$ and $b_l I(b_l(aI)a_u)$. Both are, of course, antecedents of $a + b + a + b$ by $\mathcal{G}_{\text{t2s}} \circ \mathcal{G}_{\text{der2t}}$.

The construction of Sect. 4.2 will allow us to define the composition of the multityped ACG $\mathcal{G}^+$ and $\mathcal{G}_{\text{der2t}}$ (see Example 9 for the resulting lexicon), where, in order to avoid even more types and constants, $\Sigma_{\text{derivation}}$ can be considered as a trivial multityped ACG with the identity as relabeling. The admissible parse structures are derivation trees that are furthermore interpreted as left and right-branching derived trees.

Table 3. The signature $\Sigma_{\text{derivation}}$ and the lexicon $\mathcal{L}_{\text{der2t}}$

$a_r, b_l : T_a \multimap T \multimap T$	$a_u, b_u : T$	$a, b : T_a \multimap T_a$	$I : T_a$
$T := T$	$T_a := T \multimap T$	$a_u := a$	$b_u := b$
$a_r := \lambda r t.r \text{ (cat)}$	$b_l := \lambda r t.r \text{ (ctb)}$	$a := \lambda r t.r \text{ (cta)}$	$b := \lambda r t.r \text{ (cbt)}$
			

4.2 Kanazawa's Construction

[8] defines the construction of a simple ACG to demonstrate the closure of string-encoding (resp. tree-encoding) ACGs under intersection with regular languages (resp. regular tree languages). The construction actually generalizes to any simple signature that is the codomain of both a lexicon and a relabeling. In this paper, we name this construction the *fiber product* and introduce a dedicated notation.

Definition 17 (Fiber Product, [8]). Let Σ , Σ_1 , and $\Sigma_0 = \langle A_0, C_0, \tau_0 \rangle$ be simple signatures. Let $\mathcal{L} : \Sigma_0 \rightarrow \Sigma$ be a simple lexicon and $\mathcal{R} : \Sigma_1 \rightarrow \Sigma$ a relabeling. Their fiber product signature is the signature $\Sigma_0 \times_{\Sigma} \Sigma_1 = \langle A, C, \tau \rangle$ with the lexicon $\mathcal{L}_* : \Sigma_0 \times_{\Sigma} \Sigma_1 \rightarrow \Sigma_1$ and the relabeling $\mathcal{R}_* : \Sigma_0 \times_{\Sigma} \Sigma_1 \rightarrow \Sigma_0$ such that:

- the set of atomic types of the fiber product $\Sigma_0 \times_{\Sigma} \Sigma_1$ is the fiber product of the atomic types of Σ_0 and the linear implicative types built upon Σ_1 over $\mathcal{L}$ and $\mathcal{R}$: $A = \{a^\beta \mid a \in A_0, \beta \in \mathcal{T}_{\Sigma_1}, \mathcal{L}(a) = \mathcal{R}(\beta)\}$
- the set of constants of the fiber product $\Sigma_0 \times_{\Sigma} \Sigma_1$ is the fiber product of the constants of Σ_0 and the linear λ -terms built upon Σ_1 over $\mathcal{L}$ and $\mathcal{R}$: $C = \{c_{N;\beta} \mid c \in C_0, \vdash_{\Sigma_1} N : \beta \text{ such that } \mathcal{L}(\vdash_{\Sigma_0} c : \tau_0(c)) = \mathcal{R}(\vdash_{\Sigma_1} N : \beta)\}$
- τ assigns a constant $c_{N;\beta}$ to $\text{anti}(\tau_0(c), \beta)$, a most specific common anti-instance of $\tau_0(c)$ and β , with $\text{anti}(\alpha_0 \multimap \alpha_1, \beta_0 \multimap \beta_1) = \text{anti}(\alpha_0, \beta_0) \multimap \text{anti}(\alpha_1, \beta_1)$ and $\text{anti}(a, \beta) = a^\beta$ for $a \in A_0$
- $\mathcal{L}_*$ forgets the baseline: $\forall a^\beta \in A, \mathcal{L}_*(a^\beta) = \beta$; $\forall c_{N;\beta} \in C, \mathcal{L}_*(c_{N;\beta}) = N$
- $\mathcal{R}_*$ forgets the adornment: $\forall a^\beta \in A, \mathcal{R}_*(a^\beta) = a$; $\forall c_{N;\beta} \in C, \mathcal{R}_*(c_{N;\beta}) = c$

Example 9 (Fiber Product Signature Resulting from the Composition of mACGs in Example 8). Table 4 provides the signature $\Sigma_{\text{derivation}} \times \Sigma_{\text{tree}}$ as in Example 8. It follows the construction given in Definition 17 above with $\Sigma_0 = \Sigma_{\text{derivation}}$, $\Sigma_1 = \Sigma_{\text{lr-tree}}$, and $\Sigma = \Sigma_{\text{tree}}$. It corresponds to the diagram of Fig. 5. Note that, in order to avoid having even more types and constants, we are in the special case where the top most signature $\Pi_0 = \Sigma_{\text{derivation}}$ is simple, i.e. the relabeling is the identity on $\Sigma_{\text{derivation}}$.

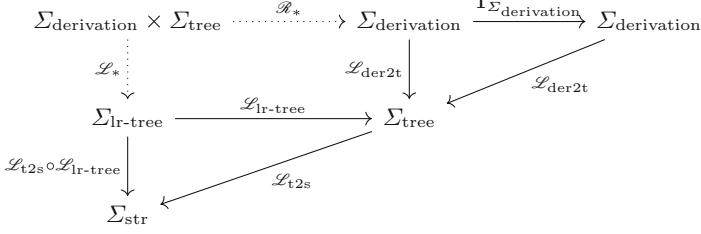


Fig. 5. Solution to the composition problem for Example 8

[8] showed that the fiber product signature makes the square diagram in Fig. 6 commute: $\mathcal{L} \circ \mathcal{R}_* = \mathcal{R} \circ \mathcal{L}_*$. Thus, $\Sigma_0 \times_{\Sigma} \Sigma_1$ is a solution to our composition problem pictured in Fig. 4a. For the remainder of this section, let consider an arbitrary fiber product signature as depicted in Fig. 6.

$$\begin{array}{ccc} \Sigma_0 \times_{\Sigma} \Sigma_1 & \xrightarrow{\mathcal{R}_*} & \Sigma_0 \\ \downarrow \mathcal{L}_* & & \downarrow \mathcal{L} \\ \Sigma_1 & \xrightarrow{\mathcal{R}} & \Sigma \end{array}$$

Fig. 6. The fiber product signature $\Sigma_0 \times_{\Sigma} \Sigma_1$

To prove the fiber product signature is the pullback, we prove the stronger property that, for any pair of Σ_0 and Σ_1 -judgments mapped to the same Σ -judgment under $\mathcal{L}$ and $\mathcal{R}$, there exists a unique typing judgment built upon $\Sigma_0 \times_{\Sigma} \Sigma_1$ which is mapped to that pair under $\mathcal{R}_*$ and $\mathcal{L}_*$.

First, we prove this property for types. The unique type built upon the fiber product is the most specific common anti-instance built for the fiber product.

Lemma 4. *Let $\alpha \in \mathcal{T}_{\Sigma_0}$ and $\beta \in \mathcal{T}_{\Sigma_1}$ such that $\mathcal{L}(\alpha) = \mathcal{R}(\beta)$. Then $\gamma = \text{anti}(\alpha, \beta)$ is the unique $\gamma \in \mathcal{T}_{\Sigma_0 \times_{\Sigma} \Sigma_1}$ such that $\mathcal{R}_*(\gamma) = \alpha$ and $\mathcal{L}_*(\gamma) = \beta$.*

Proof. For the proof legibility, we will write $\text{anti}(\Gamma, \Delta)$ for the context $x_1 : \text{anti}(\alpha_1, \beta_1), \dots, x_n : \text{anti}(\alpha_n, \beta_n)$ when $\Gamma = x_1 : \alpha_1, \dots, x_n : \alpha_n$ and $\Delta = x_1 : \beta_1, \dots, x_n : \beta_n$. In particular, $\text{anti}(\emptyset, \emptyset) = \emptyset$.

Table 4. Constants of $\Sigma_{\text{derivation}} \times \Sigma_{\text{tree}}$

$$\begin{aligned}
& a_{ua_u:T_u}, b_{ub_u:T_u} : \text{anti}(T, T_u) = T^{T_u} \\
& a_{ua_l:T_l}, b_{ub_l:T_l} : T^{T_l} \\
& a_{ua_r:T_r}, b_{ub_r:T_r} : T^{T_r} \\
& a_{r\lambda r\ t.r}(c_l\ a_l\ t):(T_l \multimap T_u) \multimap T_u \multimap T_u : T_a^{T_l \multimap T_u} \multimap T^{T_u} \multimap T^{T_u} \\
& a_{r\lambda r\ t.r}(c_l\ a_l\ t):(T_l \multimap T_l) \multimap T_u \multimap T_l : T_a^{T_l \multimap T_l} \multimap T^{T_u} \multimap T^{T_l} \\
& a_{r\lambda r\ t.r}(c_l\ a_l\ t):(T_l \multimap T_r) \multimap T_u \multimap T_r : T_a^{T_l \multimap T_r} \multimap T^{T_u} \multimap T^{T_r} \\
& a_{r\lambda r\ t.r}(c_r\ a_u\ t):(T_r \multimap T_u) \multimap T_r \multimap T_u : T_a^{T_r \multimap T_u} \multimap T^{T_r} \multimap T^{T_u} \\
& a_{r\lambda r\ t.r}(c_r\ a_u\ t):(T_r \multimap T_l) \multimap T_r \multimap T_l : T_a^{T_r \multimap T_l} \multimap T^{T_r} \multimap T^{T_l} \\
& a_{r\lambda r\ t.r}(c_r\ a_u\ t):(T_r \multimap T_r) \multimap T_r \multimap T_r : T_a^{T_r \multimap T_r} \multimap T^{T_r} \multimap T^{T_r} \\
& b_{l\lambda r\ t.r}(c_l\ t\ b_u):(T_l \multimap T_u) \multimap T_l \multimap T_u : T_a^{T_l \multimap T_u} \multimap T^{T_l} \multimap T^{T_u} \\
& b_{l\lambda r\ t.r}(c_l\ t\ b_u):(T_l \multimap T_l) \multimap T_l \multimap T_l : T_a^{T_l \multimap T_l} \multimap T^{T_l} \multimap T^{T_l} \\
& b_{l\lambda r\ t.r}(c_l\ t\ b_u):(T_l \multimap T_r) \multimap T_l \multimap T_r : T_a^{T_l \multimap T_r} \multimap T^{T_l} \multimap T^{T_r} \\
& b_{l\lambda r\ t.r}(c_r\ t\ b_r):(T_r \multimap T_u) \multimap T_u \multimap T_u : T_a^{T_r \multimap T_u} \multimap T^{T_u} \multimap T^{T_u} \\
& b_{l\lambda r\ t.r}(c_r\ t\ b_r):(T_r \multimap T_l) \multimap T_u \multimap T_l : T_a^{T_r \multimap T_l} \multimap T^{T_u} \multimap T^{T_l} \\
& b_{l\lambda r\ t.r}(c_r\ t\ b_r):(T_r \multimap T_r) \multimap T_u \multimap T_r : T_a^{T_r \multimap T_r} \multimap T^{T_u} \multimap T^{T_r} \\
& a_{\lambda r\ t.r}(c_l\ t\ a_u):(T_l \multimap T_u) \multimap T_l \multimap T_u : T_a^{T_l \multimap T_u} \multimap T_a^{T_l \multimap T_u} \\
& a_{\lambda r\ t.r}(c_l\ t\ a_u):(T_l \multimap T_l) \multimap T_l \multimap T_l : T_a^{T_l \multimap T_l} \multimap T_a^{T_l \multimap T_l} \\
& a_{\lambda r\ t.r}(c_l\ t\ a_u):(T_l \multimap T_r) \multimap T_l \multimap T_r : T_a^{T_l \multimap T_r} \multimap T_a^{T_l \multimap T_r} \\
& a_{\lambda r\ t.r}(c_r\ t\ a_r):(T_r \multimap T_u) \multimap T_u \multimap T_u : T_a^{T_r \multimap T_u} \multimap T_a^{T_r \multimap T_u} \\
& a_{\lambda r\ t.r}(c_r\ t\ a_r):(T_r \multimap T_l) \multimap T_u \multimap T_l : T_a^{T_r \multimap T_l} \multimap T_a^{T_r \multimap T_l} \\
& a_{\lambda r\ t.r}(c_r\ t\ a_r):(T_r \multimap T_r) \multimap T_u \multimap T_r : T_a^{T_r \multimap T_r} \multimap T_a^{T_r \multimap T_r} \\
& b_{\lambda r\ t.r}(c_l\ b_l\ t):(T_l \multimap T_u) \multimap T_u \multimap T_u : T_a^{T_l \multimap T_u} \multimap T_a^{T_l \multimap T_u} \\
& b_{\lambda r\ t.r}(c_l\ b_l\ t):(T_l \multimap T_l) \multimap T_u \multimap T_l : T_a^{T_l \multimap T_l} \multimap T_a^{T_l \multimap T_l} \\
& b_{\lambda r\ t.r}(c_l\ b_l\ t):(T_l \multimap T_r) \multimap T_u \multimap T_r : T_a^{T_l \multimap T_r} \multimap T_a^{T_l \multimap T_r} \\
& b_{\lambda r\ t.r}(c_r\ b_u\ t):(T_r \multimap T_u) \multimap T_r \multimap T_u : T_a^{T_r \multimap T_u} \multimap T_a^{T_r \multimap T_u} \\
& b_{\lambda r\ t.r}(c_r\ b_u\ t):(T_r \multimap T_l) \multimap T_r \multimap T_l : T_a^{T_r \multimap T_l} \multimap T_a^{T_r \multimap T_l} \\
& b_{\lambda r\ t.r}(c_r\ b_u\ t):(T_r \multimap T_r) \multimap T_r \multimap T_r : T_a^{T_r \multimap T_r} \multimap T_a^{T_r \multimap T_r} \\
& I_{\lambda x.x:T_u \multimap T_u} : T_a^{T_u \multimap T_u} \\
& I_{\lambda x.x:T_l \multimap T_l} : T_a^{T_l \multimap T_l} \\
& I_{\lambda x.x:T_r \multimap T_r} : T_a^{T_r \multimap T_r}
\end{aligned}$$

By induction over α .

When α is atomic, so is γ . By construction of the fiber product, $\gamma = \alpha^\beta = \text{anti}(\alpha, \beta)$.

When α is an arrow type, so is $\mathcal{R}(\beta)$. Since inverse relabelings preserve syntactic structures, so is β , and the induction hypothesis applies twice.

On typing judgments, the property actually has more interest when formulated up to β -equivalence.

Lemma 5. *Let $\pi_0 = \Gamma \vdash_{\Sigma_0} t : \alpha$ and $\pi_1 = \Delta \vdash_{\Sigma_1} N : \beta$ such that $\mathcal{L}(\pi_0) =_{\beta} \mathcal{R}(\pi_1)$. Then $\pi_* = \text{anti}(\Gamma, \Delta) \vdash_{\Sigma_0 \times_{\Sigma} \Sigma_1} M : \text{anti}(\alpha, \beta)$ is the unique π_* up to β -equivalence such that $\mathcal{R}_*(\pi_*) =_{\beta} \pi_0$ and $\mathcal{L}_*(\pi_*) =_{\beta} \pi_1$.*

Proof. By induction over the Σ_0 -judgment.

When t is a constant, Γ is an empty context. Since $\mathcal{R}_*$ preserves structures, suitable judgments feature terms β -equivalent to a constant. By construction of the fiber product, $\vdash_{\Pi_0} c_{N:\beta} : \alpha^{\beta}$ is the unique solution.

When t is a variable x , then $N = x$, $\Gamma = x : \alpha$ and $\Delta = x : \beta$. Necessarily, $M = x$ and Lemma 4 applies.

When $t = \lambda x.u$, then $\alpha = \alpha_0 \multimap \alpha_1$ by inversion lemma. Since $\mathcal{R}_*$ preserves structures, M is β -equivalent to an abstraction over x . Since $\mathcal{R}$ preserves structures, $N =_{\beta} \lambda x.P$. By subject equivalence, $\Gamma \vdash_{\Sigma_1} \lambda x.P : \beta$. By inversion lemma, $\beta = \beta_0 \multimap \beta_1$. The induction hypothesis applies to $\Gamma, x : \alpha_0 \vdash_{\Sigma_0} u : \alpha_1$ with $\Delta, x : \beta_0 \vdash_{\Sigma_1} P : \beta_1$ and yields $\Theta, x : \gamma_0 \vdash_{\Sigma_0 \times_{\Sigma} \Sigma_1} Q : \gamma_1$. By construction of anti , $\gamma_0 \multimap \gamma_1 = \text{anti}(\alpha, \beta)$.

When $t = uv$, then Lemma 1 yields $N =_{\beta} N_u N_v$. Since N is linear, subject equivalence applies: $\Delta \vdash_{\Sigma_1} N_u N_v : \beta$. Thus, the last rule in this judgment is an application and the induction hypothesis applies to both premises. It yields $\Theta_u \vdash_{\Sigma_0 \times_{\Sigma} \Sigma_1} M_u : \text{anti}(\alpha' \multimap \alpha, \beta' \multimap \beta)$ and $\Theta_v \vdash_{\Sigma_0 \times_{\Sigma} \Sigma_1} M_v : \text{anti}(\alpha', \beta')$. By construction of anti , the typing judgment $\Theta_u, \Theta_v \vdash_{\Pi_0} M_u M_v : \text{anti}(\alpha, \beta)$ holds.

This property on typing judgments also holds for syntactic equality. One could even drop the linearity of the λ -terms, for this version does not need subject expansion.

Theorem 1. *The fiber product signature is the pullback.*

Proof. Let Σ' be a simple signature with two lexicons F and G such that $F \circ \mathcal{R} =_{\beta} G \circ \mathcal{L}$. From Lemma 5, we build the lexicon (F, G) (see Fig. 4b). Given a Σ' -judgment $\Gamma \vdash_{\Sigma'} t : \alpha$, we define the lexicon $(F, G)(\Gamma \vdash_{\Sigma'} t : \alpha)$ as the unique $\Sigma_0 \times_{\Sigma} \Sigma_1$ -judgment Lemma 5 builds from the premises $F(\Gamma \vdash_{\Sigma'} t : \alpha)$ and $G(\Gamma \vdash_{\Sigma'} t : \alpha)$. The lexicon (F, G) can take no other value, for it would go against the uniqueness proven in Lemma 5.

5 Conclusion

We have introduced multityped ACGs and showed how to compose them. The need to address the issue of the combinatorial explosion of parsing ambiguity by means of probabilistic or weighted ACGs motivated this extension. The proposed construction, as a fiber product, shows how the resulting ACG uses the overloadings of each of the ACG. For instance, in Fig. 4a, Σ_* and the relabeling $\mathcal{R}_*$ depend both on $\mathcal{R}_{\Pi_1}$ and $\mathcal{R}_{\Pi_0}$. Since, as discussed in Sect. 2.2, the overloaded signatures seem to be the relevant place to introduce weights, it shows that the weights of a parse structure of an ACG $\mathcal{G}_1^+$, when considered as an object structure of an ACG $\mathcal{G}_0^+$, should be able to play a role in the parse

structure of the resulting ACG $\mathcal{G}^+ = \mathcal{G}_1^+ \circ \mathcal{G}_0^+$. Further work will take advantage of this feature, which fully apply to higher-order ACGs as well, to explore ways towards weighted ACGs.

References

1. Barendregt, H.P.: The Lambda Calculus: Its Syntax and Semantics. Studies in Logic and the Foundations of Mathematics, vol. 103. North-Holland (1984)
2. Booth, T., Thompson, R.: Applying probability measures to abstract languages. *IEEE Trans. Comput.* **C-22**(5), 442–450 (1973). <https://doi.org/10.1109/T-C.1973.223746>
3. Chiang, D.: Statistical parsing with an automatically-extracted tree adjoining grammar. In: Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics, pp. 456–463. Association for Computational Linguistics, Hong Kong (2000). <https://doi.org/10.3115/1075218.1075276>. ACL anthology: P00-1058
4. de Groote, P.: Towards abstract categorial grammars. In: Proceedings of 39th Annual Meeting of the Association for Computational Linguistics, pp. 148–155 (2001). ACL anthology: P01-1033
5. de Groote, P.: Tree-adjoining grammars as abstract categorial grammars. In: Proceedings of the Sixth International Workshop on Tree Adjoining Grammars and Related Frameworks (TAG+6), pp. 145–150. Università di Venezia (2002). ACL anthology: W02-2220.
6. Fülöp, Z., Vogler, H.: Weighted tree automata and tree transducers. In: Droste, M., Kuich, W., Vogler, H. (eds.) *Handbook of Weighted Automata*. An EATCS Series, pp. 313–403. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01492-5_9
7. Joshi, A.K., Schabes, Y.: Tree-Adjoining Grammars. In: Rozenberg, G., Salomaa, A. (eds.) *Handbook of Formal Languages*, vol. 3, pp. 69–123. Springer, Heidelberg (1997). https://doi.org/10.1007/978-3-642-59126-6_2. Chap. 2
8. Kanazawa, M.: Abstract families of abstract categorial languages. *Electron. Notes Theor. Comput. Sci.* **165**, 65–80 (2006). <https://doi.org/10.1016/j.entcs.2006.05.037>. Proceedings of the 13th Workshop on Logic, Language, Information and Computation (WoLLIC 2006)
9. Maletti, A., Satta, G.: Parsing algorithms based on tree automata. In: Proceedings of the 11th International Conference on Parsing Technologies (IWPT 2009), pp. 1–12. Association for Computational Linguistics, Paris, France (2009). ACL anthology: W09-3801
10. Resnik, P.: Probabilistic tree-adjoining grammar as a framework for statistical natural language processing. In: COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics (1992). <https://doi.org/10.3115/992133.992135>. ACL anthology: C92-2065
11. Salomaa, A.: Probabilistic and weighted grammars. *Inf. Control* **15**(6), 529–544 (1969). [https://doi.org/10.1016/S0019-9958\(69\)90554-3](https://doi.org/10.1016/S0019-9958(69)90554-3)
12. Simmons, H.: *An Introduction to Category Theory*. Cambridge University Press, Cambridge (2011). <https://doi.org/10.1017/CBO9780511863226>
13. Steedman, M., Baldridge, J.: Combinatory categorial grammar. In: Borsley, R., Börjars, K. (eds.) *Non-Transformational Syntax: Formal and Explicit Models of Grammar*, chap. 5. Wiley-Blackwell (2011)



Interval Probability for Sessions Types

Bogdan Aman^{1,2}  and Gabriel Ciobanu^{1,2} 

¹ Romanian Academy, Institute of Computer Science, Iași, Romania

`bogdan.aman@iit.academiaromana-is.ro`

² Alexandru Ioan Cuza University, Iași, Romania

`gabriel@info.uaic.ro`

Abstract. We propose a typing system based on multiparty session types and probabilistic interactions. We define a probabilistic process calculus using nondeterministic external choices and probabilistic internal choices. Interval probability (a generalization of traditional probability using pairs of lower and upper probabilities) are used for global and local types. We introduce a decidable proof system involving interval probability to represent and quantify the uncertain behaviour of concurrent processes. This proof system ensures type preservation and safety, even when the use of probabilities allows a process to be considered well-typed for several types. Additionally, processes remain well-typed after removing their probabilities.

1 Introduction

Multiparty session types provide a type discipline for communication-centric systems able to enforce strong communication safety for distributed processes [10]. This is achieved by using global types of the interaction between several processes that are projected to local types against which processes can be statically type-checked. Well-typed processes interact correctly following the multiparty session protocol (defined as a structured sequence of interactions).

The theory of session types is working in the absence of systems errors (such as transport-level failure), without any tolerance regarding the interactions or allowable variation for various parameters of a session protocol [10]. This inflexibility could be relaxed by a probabilistic approach, both at the abstract level of global/local types and at the concrete level of processes. To get a more flexible session type theory for realistic protocols in distributed systems, we use interval probability for global and local types, and allow probabilistic selection made internally by the interacting processes (together with nondeterministic branching determined by an external process). Using interval probability in the framework of process calculi and session types represents our main contribution.

The foundations of the interval probability theory are based on a system of three axioms in addition to Kolmogorov's axioms¹, and the theory does not

¹ This new probabilistic approach satisfies the standard axioms for determining the probability of an event [21]. The probability of an event is a real number in the interval $[0, 1]$, and the sum of the probabilities of the elementary events is 1; these axioms are presented by A.Kolmogorov in [13].

depend upon interpretations of the probability concept. In fact, the idea of interval probability is not novel. We can mention the Dempster-Shafer theory for reasoning with uncertainty by using upper and lower probabilities [7]. Even earlier in 1921, when Keynes used explicit intervals for approaching probabilities [12]². In analytic philosophy, the probability judgements were formulated in terms of interval-valued functions [14], and in terms of upper and lower probabilities [22]. Information on how interval probability differs from the classic Bayesian approach can be found in [9].

To have an enhanced understanding of the quantitative aspects of uncertainty that might arise in multiparty session types, we use a probability interval by considering sets of probabilities (rather than a single probability) in terms of lower and upper probabilities. These probability intervals appear for instance in voting polls, where it is reported an interval given by a certain percentage and a margin error. To illustrate our approach, we consider a simple example inspired by survey polls in which the pollsters communicate their own beliefs and receive the beliefs of others in the form of numerical probabilities [20].

This paper is organized as follows. Section 2 presents the syntax and semantics of our calculus; the key ideas are explained by using a running example. Section 3 describes both global and local types used in session types, as well as the connection between them. Section 4 presents the new probabilistic typing system, and the main results. The probabilistic extension preserves the classical properties of a typing system; additionally, it can determine the probability interval of certain behaviours. Section 5 concludes and discusses certain related probabilistic approaches. involving typing systems.

2 Processes in Probabilistic Multiparty Sessions

In this section we describe a probabilistic extension of a synchronous multiparty session process calculus presented in [16]; we use the same notations. Since our focus is on probabilities, note that our syntax is simplified in the style of [8], namely it has no dedicated input/output prefixes nor session initiations. This probabilistic extension allows probabilistic selection made internally by the communicating processes, and also branching controlled by an external process.

Syntax. Informally, a session represents a unit of conversation given by a series of interactions between multiple parties. A session is established via a shared name representing a public interaction point, and consists of communication actions performed on fresh session channels. The syntax is presented in Table 1, where the following notations are used: probabilities p , channels c and s , values v , variables x , labels l , role r and process variables X in order to define recursive behaviours. In what follows, $fc(P)$ denotes the set of free channels with roles in P , while $fv(P)$ denotes the set of free variables in P .

² B. Russell called this book “undoubtedly the most important work on probability that has appeared for a very long time”.

Table 1. Syntax of the probabilistic multiparty session calculus

<i>Processes</i>	$P ::= c[r] \oplus_{i \in I} \langle p_i : l_i(v_i); P_i \rangle \mid c[r] \&_{i \in I} \{l_i(x_i); P_i\} \mid (\nu s)P$ $\mid \text{def } D \text{ in } P \mid X(\tilde{v}\tilde{c}) \mid \mathbf{0} \mid P \mid Q \mid \text{err}$
<i>Declaration</i>	$D ::= X(\tilde{x}) = P$
<i>Context</i>	$\mathcal{E} ::= [] \mid (\nu s)\mathcal{E} \mid \text{def } D \text{ in } \mathcal{E} \mid \mathcal{E} \mid P$
<i>Channel</i>	$c ::= x \mid s[r]$
<i>Values</i>	$v ::= s \mid \text{true} \mid \text{false} \mid 3 \mid \dots$

The process $c[r] \oplus_{i \in I} \langle p_i : l_i(v_i); P_i \rangle$ performs an internal choice on the channel c towards role r , the labelled value $l_i(v_i)$ is sent with the probability p_i , and the execution continues as process P_i . Depending on which label is selected, there are $|I|$ possible continuations, all unique due to the uniqueness of the labels in each set. When all the sets of probabilistic choices of a process P contain only one element with probability 1, then the process P can be defined using the syntax presented in [16] (by discarding all the probabilities). On the other hand, the process $c[r] \&_{i \in I} \{l_i(x_i); P_i\}$ waits for an external choice on the channel c from role r . If the label l_i is used to communicate, then the execution continues as P_i in which all the occurrences of the variable x_i are replaced by the received value (note that for all $i \in I$, the variables x_i are bound with scope P_i).

The restriction $(\nu s)P$ delimits the scope of a session s to P . Process definition $\text{def } D \text{ in } P$ and process call $X(\tilde{v}\tilde{c})$ describe recursion, with D being a process declaration of the form $X(\tilde{x}) = P$; the call invokes X by replacing its formal parameters $\tilde{x}$ with the actual ones $\tilde{v}$ from the process call. Just like in [16], we consider closed process declarations for which we have $fv(P) \subseteq \tilde{x}$ and $fc(P) = \emptyset$ whenever $X(\tilde{x}) = P$. The inaction $\mathbf{0}$ represents a terminated process, while the parallel composition $P \mid Q$ represents two processes that execute concurrently and possibly communicate on a shared channel. The error process is denoted by err . As usual, evaluation contexts are processes with some holes [6]. A channel c can be either a variable x or a channel with role $s[r]$, namely a multiparty communication channel for role r in a session s . The fact that channels have specific roles in a session is inspired by the system presented in [6]. Values v can be integers, Boolean and strings (i.e., base values).

Example 1. The following process describes a simple survey poll in which *Alice* either responds to some questions of *Bob* or declines the polling. *Bob* can either stop the polling if is pleased with the received responses, or has some other polling questions to perform. Just like in [20], *Alice* and *Bob* communicate their own choices and receive the choices of others having numerical probabilities attached to them, probabilities providing a well-defined numerical scale for responses. The initial system can be described by:

$$\text{System} = (\nu s)(\text{Bob} \mid (\nu \text{PhoneNoA})(\nu \text{IMIdA})\text{Alice}).$$

The name s indicates the session involving the processes *Alice* and *Bob* composed in parallel. Session s has two roles: r_B for the process performing the survey poll, and r_A for the process answering the poll. The processes *Alice* and *Bob* communicate in this session by using the channel with role $s[r_A]$ and $s[r_B]$, respectively. In a similar manner, *PhoneNoA* and *IMIdA* represent sessions names used to model sessions established between *Alice* and *Bob* after the communication performed inside the session s ends. Note that the scopes of *PhoneNoA* and *IMIdA* include only the process *Alice* initially, as the process *Bob* is informed about which session to join after communication with *Alice*.

$Alice = \text{def } A(y) = A_1 \text{ in } A_2$, where A_1 and A_2 are the following processes:

$$A_1 = y[r_B] \& \{ \text{talk}(t_1).y[r_B] \oplus \langle 0.6 : \text{yes}(t).A(y) , 0.3 : \text{no}(t).A(y) , 0.1 : \text{quit}(t).\mathbf{0} \rangle \} , \\ \text{quit}(t_2).\mathbf{0} \} ;$$

$$A_2 = s[r_A][r_B] \oplus \langle 0.6 : \text{ComPhone}(\text{PhoneNoA}).A(\text{PhoneNoA}[r_A]) , \\ 0.35 : \text{ComIM}(\text{IMIdA}).A(\text{IMIdA}[r_A]) , 0.05 : \text{noComm}(\text{no}).\mathbf{0} \rangle .$$

$Bob = \text{def } B(y, t) = B_1 \text{ in } B_2$, where B_1 and B_2 are the following processes:

$$B_1 = y[r_A] \oplus \langle 0.95 : \text{talk}(t).y[r_A] \& \{ \text{yes}(t).B(y, \text{next}(t)) , \\ \text{no}(t).B(y, \text{next}(t)) , \text{unsure}(t).B(y, \text{next}(t)) , \text{quit}(t).\mathbf{0} \} , \\ 0.05 : \text{quit}(\text{no}).\mathbf{0} \rangle ;$$

$$B_2 = s[r_B][r_A] \& \{ \text{ComPhone}(x_A).B(x_A[r_B], Q_1) , \\ \text{ComIM}(x'_A).B(x'_A[r_B], Q_1) , \text{noComm}(x''_A).\mathbf{0} \} .$$

We explain the concurrent evolution of processes *Alice* and *Bob* by describing the interaction between them. In the first step, using the channel with role $s[r_A]$, the process *Alice* decides whether to respond to the poll either by phone, instant messaging, or not answering now. In each case, process *Alice* has attached the probabilities marking its willingness to take the appropriate branch. Suppose the *ComPhone* label is used; in this case *Alice* and *Bob* communicate by using the session channel *PhoneNoA* that represents the unique phone number of *Alice*. The process *Bob* is implemented by invoking $B(x_A[r_B], Q_1)$, where x_A becomes *PhoneNoA* after communication. Here Q_1 stands for the first question that process A has to respond by using one of the labels *yes*, *no*, *unsure* or *quit*. Notice that the process *Alice* does not provide the answer by using *unsure*, as does not consider this to be an option for the given question. After receiving an answer to the question, process *Bob* can continue to ask the next question from its list (illustrated by *next*). The processes can quit the poll by performing a selection of either the label *quit* or *noComm* (when appropriate). Notice that the *quit* and *noComm* have low attached probabilities; this means that these options have very low chances of being chosen when the survey poll is conducted.

Operational Semantics. Next, $s \notin \text{fc}(P)$ means that there does not exist a role r such that $s[r] \in \text{fc}(P)$. Also, $\text{dpv}(D)$ denotes the set of process variables declared in D , while $\text{fpv}(P)$ denotes the set of process variables which occur free in P .

Table 2. Operational semantics of the probabilistic multiparty session calculus

$s[r_1][r_2] \oplus_{j \in J} \langle p_j : l_j(v_j); P_j \rangle \mid s[r_2][r_1] \&_{i \in I} \{l_i(x_i); P'_i\} \rightarrow_{p_k} P_k \mid P'_k \{\tilde{v}_k / \tilde{x}_k\}$ (if $k \in J \subseteq I$)	(COM)
$\text{def } X(\tilde{x}) = P \text{ in } (X(\tilde{v}\tilde{c}) \mid Q) \rightarrow_1 \text{def } X(\tilde{x}) = P \text{ in } (P\{\tilde{v}\tilde{c}/\tilde{x}\} \mid Q)$ (if $\tilde{x} = x_1 \dots x_n, \tilde{v} = v_1 \dots v_i c_{i+1} \dots c_n$)	(CALL)
$P \rightarrow_p P' \text{ implies } \mathcal{E}[P] \rightarrow_{p'} \mathcal{E}[P'] \text{ (where } p' = p \cdot \text{nextProc}(P) / \text{nextProc}(\mathcal{E}[P]) \text{)}$	(CTXT)
$P \equiv P' \text{ and } P' \rightarrow_p Q' \text{ and } Q \equiv Q' \text{ implies } P \rightarrow_p Q$	(STRUCT)
$s[r_1][r_2] \oplus_{j \in J} \langle p_j : l_j(v_j); P_j \rangle \mid s[r_2][r_1] \&_{i \in I} \{l_i(x_i); P'_i\} \rightarrow_{p_k} \text{err}$ (if $k \in J$ and $k \notin I$)	(ERR)

The operational semantics is based on the structural congruence defined by:

$$\begin{aligned}
P|Q &\equiv Q|P & (P|Q)|R &\equiv P|(Q|R) & P|0 &\equiv P & (\nu s)0 &\equiv 0 & \text{def } D \text{ in } 0 &\equiv 0 \\
(\nu s)(\nu s')P &\equiv (\nu s')(\nu s)P & (\nu s)P|Q &\equiv (\nu s)(P|Q) & (\text{if } s \notin \text{fc}(Q)) \\
\text{def } D \text{ in } (\nu s)P &\equiv (\nu s)\text{def } D \text{ in } P & (\text{if } s \notin \text{fc}(D)) \\
\text{def } D \text{ in } (P|Q) &\equiv (\text{def } D \text{ in } P)|Q & (\text{if } \text{dpv}(D) \cap \text{fpv}(Q) = \emptyset) \\
\text{def } D \text{ in } \text{def } D' \text{ in } P &\equiv \text{def } D' \text{ in } \text{def } D \text{ in } P \\
(\text{if } (\text{dpv}(D) \cup \text{fpv}(D)) \cap \text{dpv}(D') &= (\text{dpv}(D') \cup \text{fpv}(D')) \cap \text{dpv}(D) = \emptyset).
\end{aligned}$$

The semantics of our probabilistic calculus is inductively defined in Table 2, up to the defined structural congruence $\equiv$ and including α -conversion. A process P has an error if and only if for some context $\mathcal{E}$ it holds that $P = \mathcal{E}[\text{err}]$.

Rule (COMM) models the communication between two roles r_1 and r_2 operating in the same session s by using a branching process and a probabilistic selection process. If the process $s[r_1][r_2] \oplus_{j \in J} \langle p_j : l_j(v_j); P_j \rangle$ chooses a branch with label l_k and probability p_k , then the corresponding branch with label l_k is chosen also in process $s[r_2][r_1] \&_{i \in I} \{l_i(x_i); P'_i\}$. The first process continues as P_k , while the second one as $P'_k \{\tilde{v}_k / \tilde{x}_k\}$ obtained by using the communicated value $\tilde{v}_k$ and substituting it for the variable $\tilde{x}_k$. It is worth noting that it is possible that the options offered by the selection process are fewer than the ones offered by the branching process because the last one has to take into account all the possible continuations (depending on different scenarios). However, all the options of the selection set J should be present in the branching set I (expressed by the condition $J \subseteq I$) in order to avoid choosing a branch with no correspondence. However, if there exists an option k of the selection set J that is not present in the branching set I (expressed by the conditions $k \in J$ and $k \notin I$) and one chooses the branch k with no correspondence, then by applying the rule (ERR) the process reduces to **err**.

Rule (CALL) instantiates a process call by using its definition and replacing its formal parameters $\tilde{x}$ with the actual ones $\tilde{v}$. The side condition of this rule ensures that the number of variables of the definition is equal with the number of variables appearing in the process call. Rule (STRUCT) states that the transition relation is closed under structural congruence. Rule (CTXT) states that

the transition can happen under restriction, process definition and parallel composition. The new value p' of the probability attached to the reduction arrow is a normalization of the old probability value p with respect to the number of rules (COMM) and (CALL) that can be applied. To compute the number of possible rules (COMM) and (CALL) that can be applied to a process P , we use the *nextProc* function defined as follows:

$$\text{nextProc}(P) = \begin{cases} 1 + \text{nextProc}(P'') & \text{if } P \equiv s[r_1][r_2] \oplus_{j \in J} \langle p_j : l_j(v_j); P_j \rangle \\ & \quad | s[r_2][r_1] \&_{i \in I} \{l_i(x_i); P'_i\} | P'' \\ \text{nextProc}(P') + \text{nextProc}(P'') & \text{if } P \equiv (\nu s)P' | P'' \\ k + \text{nextProc}(P') + \text{nextProc}(P'') & \text{if } P \equiv \text{def } D \text{ in } (X(\tilde{v}_1) | \dots | X(\tilde{v}_k) | P') | P'' \\ & \quad \text{where } D ::= X(\tilde{x}) = Q \text{ and } X(\tilde{v}) \notin P' \\ 0 & \text{otherwise.} \end{cases}$$

Since the function *nextProc* is used only for applying the rule (CTXT), this means that $\text{nextProc}(P) > 0$ and $\text{nextProc}(\mathcal{E}[P]) > 0$ as both P and $\mathcal{E}[P]$ can perform at least one reduction step.

The need for this function will be seen in the following examples.

Example 2. (cont.). This example illustrates the semantics. To help the readability, we underline the actions used in reduction. Thus, the initial process becomes:

$$\begin{aligned} & (\nu s)(\text{Bob} | (\nu \text{PhoneNoA}, \text{IMIdA})\text{Alice}) \equiv \\ & \equiv (\nu s)(\nu \text{PhoneNoA})(\nu \text{IMIdA})(\text{def } B(y, t) = B_1 \text{ in def } A(y) = A_1 \text{ in} \\ & \quad \underline{s[r_B][r_A]} \& \{ \underline{\text{ComPhone}(x_A).B(x_A[r_B], Q_1)}, \\ & \quad \quad \underline{\text{ComIM}(x'_A).B(x'_A[r_B], Q_1)}, \text{noComm}(x'_A).0 \} \\ & \quad | \underline{s[r_A][r_B]} \oplus \langle 0.6 : \underline{\text{ComPhone}(\text{PhoneNoA}).A(\text{PhoneNoA}[r_A])}, \\ & \quad \quad \underline{0.35 : \text{ComIM}(\text{IMIdA}).A(\text{IMIdA}[r_A])}, 0.05 : \text{noComm}(\text{no}).0 \rangle) \\ & = \text{System}_1. \end{aligned}$$

Let us note that only a (COM) rule is applicable, followed by an application of the (CTXT) rule in which the context $\mathcal{E} = (\nu s)(\nu \text{PhoneNoA})(\nu \text{IMIdA})\text{def } B(y, t) = B_1 \text{ in def } A(y) = A_1 \text{ in } [\]$ is used. This is captured by $\text{nextProc}(\text{System}_1) = 1$. Assume that *Alice* decides to respond to the poll by phone by using the *ComPhone* label. Then the next step is obtained with probability 0.6:

$$\begin{aligned} & \rightarrow_{0.6} (\nu s)(\nu \text{PhoneNoA})(\nu \text{IMIdA})(\text{def } B(y, t) = B_1 \text{ in def } A(y) = A_1 \text{ in} \\ & \quad B(x_A[r_B], Q_1)\{\text{PhoneNoA}/x_A\} | A(\text{PhoneNoA}[r_A])) \\ & \equiv (\nu s)(\nu \text{PhoneNoA})(\nu \text{IMIdA})(\text{def } B(y, t) = B_1 \text{ in def } A(y) = A_1 \text{ in} \\ & \quad B(\text{PhoneNoA}[r_B], Q_1) | A(\text{PhoneNoA}[r_A])) \\ & = \text{System}_2. \end{aligned}$$

Notice that the (CALL) rule is applicable to two different process calls, followed by an application of the (CTXT) rule. This is captured by $\text{nextProc}(\text{System}_2) = 2$, and thus the probability of applying the (CTXT) rule is half of the probability of applying a (CALL) rule. Assume that the process definition $P = \text{def } B(y, t) = B_1 \text{ in } B(\text{PhoneNoA}[r_B], Q_1)$ is executed inside the context $\mathcal{E} = (\nu s)(\nu \text{PhoneNoA})(\nu \text{IMIdA})([\] | \text{def } A(y) = A_1 \text{ in } A(\text{PhoneNoA}[r_A]))$, namely $\text{System}_2 = \mathcal{E}[P]$

$$\begin{aligned}
&\equiv (\nu s)(\nu \text{PhoneNoA})(\nu \text{IMIdA}) (\text{def } B(y, t) = B_1 \text{ in } B(\text{PhoneNoA}[r_B], Q_1) \\
&\quad | \text{def } A(y) = A_1 \text{ in } A(\text{PhoneNoA}[r_A])) \\
&\rightarrow_{0.5} (\nu s)(\nu \text{PhoneNoA})(\nu \text{IMIdA}) (\text{def } B(y, t) = B_1 \text{ in } B_1\{\text{PhoneNoA}[r_B]/y, Q_1/t\} \\
&\quad | \text{def } A(y) = A_1 \text{ in } A(\text{PhoneNoA}[r_A]))
\end{aligned}$$

The evolution continues by applying the rules of Table 2.

3 Global and Local Types

Probabilistic estimations of behaviours are unsatisfying if they come with no measure of reliability and soundness. To validate the probabilistic behaviours in session types, we introduce a decidable proof system based on the concept of interval probability to represent and quantify uncertainty in the study of concurrent processes. The concept of interval probability is more general than classical probability for describing uncertainty; as a special case, classical probability fits properly into the theory of interval probability [21].

The **global types** $G, G', \dots$ presented in Table 3 describe the global behaviour of a probabilistic multiparty session process. We use the probability interval ι having the form $[d_1, d_2]$ in which $d_1, d_2 \in [0, 1]$ and $d_1 \leq d_2$. If $\iota = [d, d]$, we use the shorthand notation $\iota = d$.

Table 3. Global types of the session typing

<i>Global</i>	$G ::= r_1 \rightarrow r_2 : \{\iota_i : l_i(S_i).G_i\}_{i \in I} \mid \mu t.G \mid t \mid \text{end}$
<i>Sorts</i>	$S ::= \text{bool} \mid \text{nat} \mid \dots$

Type $r_1 \rightarrow r_2 : \{\iota_i : l_i(S_i).G_i\}_{i \in I}$ states that a participant with role r_1 sends with a probability belonging to the probability interval ι_i a message of type S_i to a participant with role r_2 by using label l_i ; then we have the interactions described by G_i . Each l_i is unique, as it was already assumed when defining the processes. Type $\mu t.G$ is recursive, where type variable t is guarded in the standard way (it appears only under a prefix).

Example 3. The global type formalizing the *PhoneNoA* session of Example 1:

$$G_A = \mu t.r_B \rightarrow r_A : \left\{ \begin{array}{l} \iota_1 : \text{talk}(\text{string}).r_A \rightarrow r_B : \left\{ \begin{array}{l} \iota_2 : \text{yes}(\text{string}).t, \\ \iota_3 : \text{no}(\text{string}).t; \\ \iota_4 : \text{unsure}(\text{string}).t; \\ \iota_5 : \text{quit}(\text{string}).\text{end} \end{array} \right\} \\ \iota_6 : \text{quit}(\text{string}).\text{end} \end{array} \right\}.$$

If $\iota_6 = [0.95, 1]$, this implies that *Bob* (the person in charge of performing the poll) is more likely to quit than to complete the poll (as required). This kind of behaviour should not be allowed, by imposing $\iota_6 = [0, 0.1]$, meaning that the chance for *Bob* to stop the polling is minimal.

Table 4. Local types of the session typing

<i>Local</i>	$T ::= r \oplus_{i \in I} \mathbf{l}_i : !l_i \langle S_i \rangle . T_i \mid r \&_{i \in I} ?l_i \langle S_i \rangle . T_i \mid \mu t . T \mid t \mid \text{end}$
<i>Sorts</i>	$S ::= \text{nat} \mid \text{bool} \mid \dots$

The **local types** $T, T', \dots$ presented in Table 4 describe the local behaviour of processes; they also represent a connection between the global types and the processes of our calculus.

The selection $r \oplus_{i \in I} \mathbf{l}_i : !l_i \langle S_i \rangle . T_i$ describes a channel that can choose a label l_i with a probability belonging to the probability interval $\mathbf{l}_i$ (for any $i \in I$), and send it to r together with a variable of type S_i ; then the channel must be used as described by T_i . The branching type $r \&_{i \in I} ?l_i \langle S_i \rangle . T_i$ describes a channel that can receive a label l_i from role r (for some $i \in I$, chosen by r), together with a variable of type S_i ; then the channel must be used as described in T_i . The labels l_i of selection and branching types are all distinct, and their order is irrelevant. The recursive type $\mu t . T$ and type variable t model infinite behaviours. Type end is the type of a terminated channel (and it is often omitted). The base types $S, S_0, S_1, \dots$ can be types like *bool*, *int*, etc.

We define the projection of a global type to a local type for each participant.

Definition 1. *The projection $G \upharpoonright r$ for a participant with role r appearing in a global type G is inductively defined as:*

- $(r_1 \rightarrow r_2 : \{\mathbf{l}_i : l_i \langle S_i \rangle . G_i\}_{i \in I}) \upharpoonright r =$

$$\begin{cases} r_1 \oplus_{i \in I} \mathbf{l}_i : !l_i \langle S_i \rangle . (G_i \upharpoonright r) & \text{if } r = r_1 \neq r_2 \\ r_2 \&_{i \in I} ?l_i \langle S_i \rangle . (G_i \upharpoonright r) & \text{if } r = r_2 \neq r_1 \\ G_1 \upharpoonright r & \text{if } r \neq r_1 \text{ and } r \neq r_2, \forall i, j \in I, G_i \upharpoonright r = G_j \upharpoonright r \end{cases};$$
- $(\mu t . G) \upharpoonright r = \begin{cases} \mu t . (G \upharpoonright r) & \text{if } G \upharpoonright r \neq \text{end} \\ \text{end} & \end{cases};$
- $t \upharpoonright r = t$;
- $\text{end} \upharpoonright r = \text{end}$.

When none of the conditions hold, the projection is undefined.

In the global type $r_1 \rightarrow r_2 : \{\mathbf{l}_i : l_i \langle S_i \rangle . G_i\}_{i \in I}$, the values of the probability interval $\mathbf{l}_i = [d_{1i}; d_{2i}]$ with $i \in I$ should not be arbitrarily assigned, but rather satisfy some restrictions [5].

Definition 2. *Consider a set of probabilities $\{\mathbf{l}_i\}_{i \in I}$, where the probability intervals have the form $\mathbf{l}_i = [d_{1i}; d_{2i}]$. The set $\{\mathbf{l}_i\}_{i \in I}$ is called proper if*

$$\sum_{i \in I} d_{1i} \leq 1 \leq \sum_{i \in I} d_{2i},$$

while the set $\{\mathbf{l}_i\}_{i \in I}$ is called reachable if for all $i, j \in I$ it holds that

$$(\sum_{i \neq j} d_{1j}) + d_{2i} \leq 1 \leq (\sum_{i \neq j} d_{2j}) + d_{1i}.$$

By defining a probability interval to be proper and reachable, we avoid cases in which there does not exist $p_i \in \iota_i$ for all $i \in I$ such that $\sum_{i \in I} p_i = 1$. More details about the probability intervals and operations over them are in [5].

Following the methodology from [10], we design a distributed system in terms of global types, which is then implemented as a set of processes respecting local types that are obtained as projections of such global types. Due to this approach, the notion of well-formed local and global types arises naturally. From now on we will implicitly assume that all local and global types are well-formed, and all sets of probability intervals are proper and reachable.

Example 4. Consider the global type G_A of Example 3, in which for all $i \in \{1, \dots, 5\}$ it holds that $\iota_i = [0, 1]$, and also $\iota_6 = [0.95, 1]$. According to Definition 1, $G_A \upharpoonright r$ is defined for all roles. Moreover, the set $\{\iota_i\}_{i \in \{2,3,4,5\}}$ is proper and reachable, and the set $\{\iota_i\}_{i \in \{1,6\}}$ is proper but not reachable.

4 Interval Probability for Multiparty Session Types

We describe how to connect the probabilistic processes of Sect. 2 to the global and local types using probability intervals (presented in Sect. 3). For this, we introduce sortings Γ and typings Δ , namely

$$\Gamma :: = \emptyset \mid \Gamma, x : S \mid \Gamma, X : \tilde{S}\tilde{T} \quad \text{and} \quad \Delta :: = \emptyset \mid \Delta, \{s[r_i] : T_i\}_{i \in I}.$$

The typing system uses a map from shared names to their sorts $(S, S', \dots)$. A sorting $(\Gamma, \Gamma', \dots)$ is a finite map from names to sorts, and from process variables to sequences of sorts and types. A typing $(\Delta, \Delta', \dots)$ records linear usage of session channels. Given two typings Δ and Δ' , their disjoint union is denoted by Δ, Δ' (by assuming that their domains contain disjoint sets of session channels).

Since the processes interact, their dynamics is formalized as in [10] by a labelled type reduction relation $\Rightarrow_\iota$ on typing Δ given by next three rules:

- $s[r_1] : r_2 \oplus_{i \in I} \iota_i : !l_i \langle S_i \rangle . T_i, s[r_2] : r_1 \&_{i \in I} ?l_i(S_i) . T_i \Rightarrow_{\iota_k} s[r_1] : T_k, s[r_2] : T'_k$ for $k \in I$;
- $s[r] : \mu t . T \Rightarrow_1 s[r] : T \{ \mu t . T / t \}$;
- $\Delta, \Delta' \Rightarrow_\iota \Delta, \Delta''$ whenever $\Delta' \Rightarrow_\iota \Delta''$.

The first rule corresponds to sending/receiving a value of type S_k by using label l_k on session channel s by the participant r_2 ; notice that the probability interval of the sending type is placed on the arrow (as it will be required in proving type preservation under evolution). The second rule is used to unfold local types with probability interval $[1, 1]$, while the third rule is used to compose typings when only a part of a typing is changing.

We formalize in what follows what it means for Δ to be safe [17], a property that does not depend on the value of the probability intervals present in Δ .

Table 5. Typing system for processes in probabilistic multiparty sessions

$\Gamma, x : S \vdash x : S \quad \Gamma \vdash v : S$	(TVAR), (TVAL)
$\frac{\forall i. \Gamma \vdash v_i : S_i \quad \forall i. \Gamma \vdash P_i \triangleright \Delta, c : T_i \quad p_i \in \iota_i}{\Gamma \vdash c[r] \oplus_{i \in I} \langle p_i : l_i(v_i); P_i \rangle \triangleright \Delta, c : r \oplus_{i \in I} \iota_i : l_i(S_i).T_i}$	(TSELECT)
$\frac{\forall i. \Gamma, x_i : S_i \vdash P_i \triangleright \Delta, c : T_i}{\Gamma \vdash c[r] \&_{i \in I} \{l_i(x_i); P_i\} \triangleright \Delta, c : r \&_{i \in I} ?l_i(S_i).T_i}$	(TBRANCH)
$\frac{\Delta \text{ end only} \quad \Gamma \vdash P \triangleright \Delta \quad \Gamma \vdash Q \triangleright \Delta'}{\Gamma \vdash \mathbf{0} \triangleright \Delta \quad \Gamma \vdash P \mid Q \triangleright \Delta, \Delta'}$	(TEND), (TCONC)
$\frac{pid(G) = I \quad \forall i. G \upharpoonright r_i = T_i \quad \Gamma \vdash P \triangleright \Delta, \{s[r_i] : T_i\}_{i \in I} \quad safe(\{s[r_i] : T_i\}_{i \in I})}{\Gamma \vdash (\nu s)P \triangleright \Delta}$	(TRES)
$\frac{\Gamma, x : \tilde{S}, X : \tilde{S}T_1 \dots T_n \vdash P \triangleright c_1 : T_1, \dots, c_n : T_n \quad \Gamma, X : \tilde{S}T_1 \dots T_n \vdash Q \triangleright \Delta}{\Gamma \vdash \text{def } X(\tilde{x}c_1 \dots c_n) = P \text{ in } Q \triangleright \Delta}$	(TDEF)
$\frac{\Gamma \vdash \tilde{v} : \tilde{S} \quad \Delta \text{ end only}}{\Gamma, X : \tilde{S}T_1 \dots T_n \vdash X(\tilde{v}c_1 \dots c_n) \triangleright \Delta, c_1 : T_1, \dots, c_n : T_n}$	(TCALL)

Definition 3. φ is a safety property of typing contexts iff:

- $\varphi(\Delta, s[r_1] : r_2 \oplus_{i \in I} \iota_i : l_i(S_i).T_i, s[r_2] : r_1 \&_{j \in J} ?l_j(S_j).T'_j)$ implies $I \subseteq J$;
- $\varphi(\Delta, s[p] : \mu t.T)$ implies $\varphi(\Delta, s[p] : T\{\mu t.T/t\})$;
- $\varphi(\Delta)$ and $\Delta \Rightarrow_\iota \Delta'$ implies $\varphi(\Delta')$.

We say Δ is safe whenever $\varphi(\Delta)$ for some safety property φ . The fact that Δ is safe is denoted by $safe(\Delta)$.

The type assignment system for processes is given in Table 5, where $pid(G)$ denotes the set of participants in G . We use the judgement $\Gamma \vdash P \triangleright \Delta$ saying that “under the sorting Γ , process P has typing Δ ”. (TVAR) and (TVAL) are the rules for typing variables and values. (TSELECT) and (TBRANCH) are the rules for typing selection and branching, respectively. These rules state that selection (branching) process is well-typed if $c[r]$ has a compatible selection (branching) type, and the continuations P_i (for all $i \in I$) are well-typed with respect to the session types. Since rule (TSELECT) types probabilistic actions by using probability intervals, the rule should check if the probabilities belong to the corresponding probability interval. Rule (TCONC) composes two processes if their local types are disjoint. Rule (TEND) is standard; “ Δ end only” means that Δ contains only end as session types. (TRES) is the restriction rule for session names and claims that $(\nu s)P$ is well-typed in Γ ; this happens if the types T_i of the session s are exactly the projections of the same global type G into all participants of G , and the obtained typing $\{s[r_i] : T_i\}_{i \in I}$ is safe. (TDEF) says that a process definition $\text{def } X(\tilde{x}c_1 \dots c_n) = P \text{ in } Q$ is well-typed if both P and Q are well-typed in their typing contexts. Rule (TCALL) says that a process call $X(\tilde{v}c_1 \dots c_n)$ is well-typed if the actual parameters $\tilde{v}c_1 \dots c_n$ have compatible types with respect to X .

We are now able to prove that this probabilistic typing system is sound, namely its term checking rules admit only terms that are consistent with respect to the structural congruence and operational semantics.

A *subject reduction* result ensures that the type of an expression is preserved during its evaluation. For the proof of the subject reduction, we need the following substitution and weakening results.

Lemma 1.

- (substitution) $\Gamma, x : S \vdash P \triangleright \Delta$ and $\Gamma \vdash v : S$ imply $\Gamma \vdash P\{v/x\} \triangleright \Delta$.
- (type weakening) Whenever $\Gamma \vdash P \triangleright \Delta$ is derivable, then its weakening is also derivable, namely $\Gamma \vdash P \triangleright \Delta, \Delta'$ for disjoint Δ' , where Δ' contains only *end*.
- (sort weakening) If $X \notin \text{dom}(\Gamma)$ and $\Gamma \vdash P \triangleright \Delta$, then $\Gamma, X : \tilde{S}\tilde{T} \vdash P \triangleright \Delta$.
- (sort strengthening) If $X \notin \text{fpv}(P)$ and $\Gamma, X : \tilde{S}\tilde{T} \vdash P \triangleright \Delta$, then $\Gamma \vdash P \triangleright \Delta$.

Proof. The proofs are rather standard, following those presented in [10].

Theorem 1 (subject equivalence). $\Gamma \vdash P \triangleright \Delta$ and $P \equiv P'$ imply $\Gamma \vdash P' \triangleright \Delta$.

Proof. The proof is by induction on $\equiv$, showing (in both ways) that if one side has a typing, then the other side has the same typing. Since the structural congruence $\equiv$ does not consume or modify the probabilistic actions, the probability intervals in the types remain unchanged. The full proof is available in [2].

Theorem 2 (subject reduction). $\Gamma \vdash P \triangleright \Delta$, $\text{safe}(\Delta)$ and $P \rightarrow_p P'$ imply $\Gamma \vdash P' \triangleright \Delta'$, where $\Delta' = \Delta$ or $\Delta \Rightarrow_\iota \Delta'$ with $p \cdot \text{nextProc}(P) \in \iota$.

Proof. By induction on the derivation of $P \rightarrow_{p_i} P'$. There is a case for each operational semantics rule, and for each operational semantics rule we consider the typing system rule generating $\Gamma \vdash P \triangleright \Delta$. Since the reduction relations $\rightarrow_p$ and $\Rightarrow_\iota$ keep track of the consumed probabilities in processes and probability intervals in types, this result is able to ensure a relation between the consumed probabilities and probability intervals at each reduction step.

- Case (CTXT): $P_0 \rightarrow_{p_0} P'_0$ implies $\mathcal{E}[P_0] \rightarrow_p \mathcal{E}[P'_0]$,
where $p = p_0 \cdot \text{nextProc}(P_0) / \text{nextProc}(\mathcal{E}(P_0))$

Let $P \equiv \mathcal{E}[P_0]$ and $P' \equiv \mathcal{E}[P'_0]$. The proof is by structural induction on $\mathcal{E}$.

- Case $\mathcal{E} = []$. This case is trivial as $P \rightarrow_p P'$ is in fact $P_0 \rightarrow_p P'_0$.
- Case $\mathcal{E} = (\nu s)\mathcal{E}'$. By assumption, $\Gamma \vdash (\nu s)\mathcal{E}'[P_0] \triangleright \Delta$. By inverting the rule (TRES), we get that $\Gamma \vdash \mathcal{E}'[P_0] \triangleright \Delta, \{s[r_i] : T_i\}_{i \in I}$, where $\text{pid}(G) = |I|$, $\forall i. G \upharpoonright r_i = T_i$ and $\text{safe}(\{s[r_i] : T_i\}_{i \in I})$. By applying rule (CTXT), $P_0 \rightarrow_{p_0} P'_0$ implies $\mathcal{E}'[P_0] \rightarrow_{p'} \mathcal{E}'[P'_0]$, where $p' = p_0 \cdot \text{nextProc}(P_0) / \text{nextProc}(\mathcal{E}'[P_0])$. By induction, $\Gamma \vdash \mathcal{E}'[P'_0] \triangleright \Delta', \{s[r_i] : T_i\}_{i \in I}$, where $\Delta' = \Delta$ or $\Delta \Rightarrow_\iota \Delta'$ and $p' \cdot \text{nextProc}(\mathcal{E}'[P_0]) \in \iota$. By applying rule (TRES), we get that $\Gamma \vdash \mathcal{E}'[P_0] \triangleright \Delta, \{s[r_i] : T_i\}_{i \in I}$, where $\text{pid}(G) = |I|$, $\forall i. G \upharpoonright r_i = T_i$ and $\text{safe}(\{s[r_i] : T_i\}_{i \in I})$. By applying rule (CTXT), $\mathcal{E}'[P_0] \rightarrow_{p'} \mathcal{E}'[P'_0]$ implies $\mathcal{E}[P_0] \rightarrow_p \mathcal{E}[P'_0]$, where $p = p' \cdot \text{nextProc}(\mathcal{E}'[P_0]) / \text{nextProc}(\mathcal{E}[P_0])$. By using the rule (TRES), we obtain $\Gamma \vdash (\nu s)\mathcal{E}'[P'_0] \triangleright \Delta'$, where $\Delta' = \Delta$ or $\Delta \Rightarrow_\iota \Delta'$, and $p \cdot \text{nextProc}((\nu s)\mathcal{E}'[P_0]) = p' \cdot \text{nextProc}(\mathcal{E}'[P_0]) = p_0 \cdot \text{nextProc}(P_0) \in \iota$, as required.

- Case $\mathcal{E} = \text{def } D \text{ in } \mathcal{E}'$. By assumption, $\Gamma \vdash \text{def } D \text{ in } \mathcal{E}'[P_0] \triangleright \Delta$. By inverting rule (TDEF), we obtain $\Gamma, x : \tilde{S}, X : \tilde{S}T_1 \dots T_n \vdash Q \triangleright c_1 : T_1, \dots, c_n : T_n$ and $\Gamma, X : \tilde{S}T_1 \dots T_n \vdash \mathcal{E}'[P_0] \triangleright \Delta$, where $D = \{X(\tilde{x}c_1 \dots c_n) = Q\}$. By applying rule (CTXT), $P_0 \rightarrow_{p_0} P'_0$ implies $\mathcal{E}'[P_0] \rightarrow_{p'} \mathcal{E}'[P'_0]$, where $p' = p_0 \cdot \text{nextProc}(P_0) / \text{nextProc}(\mathcal{E}'[P_0])$. By induction, $\Gamma, X : \tilde{S}T_1 \dots T_n \vdash \mathcal{E}'[P'_0] \triangleright \Delta'$, where $\Delta' = \Delta$ or $\Delta \Rightarrow_{\iota} \Delta'$ and $p' \cdot \text{nextProc}(\mathcal{E}'[P'_0]) \in \iota$. By applying rule (CTXT), $\mathcal{E}'[P_0] \rightarrow_{p'} \mathcal{E}'[P'_0]$ implies $\mathcal{E}[P_0] \rightarrow_p \mathcal{E}[P'_0]$, where $p = p' \cdot \text{nextProc}(\mathcal{E}'[P_0]) / \text{nextProc}(\mathcal{E}[P_0])$. By using the rule (TDEF), we obtain $\Gamma \vdash \text{def } D \text{ in } \mathcal{E}'[P'_0] \triangleright \Delta'$, where $\Delta' = \Delta$ or $\Delta \Rightarrow_{\iota} \Delta'$, and $p \cdot \text{nextProc}(\text{def } D \text{ in } \mathcal{E}'[P_0]) = p' \cdot \text{nextProc}(\mathcal{E}'[P'_0]) = p_0 \cdot \text{nextProc}(P_0) \in \iota$, as required.
- Case $\mathcal{E} = \mathcal{E}' \mid P''$. By assumption, $\Gamma \vdash \mathcal{E}'[P_0] \mid P'' \triangleright \Delta$. By inverting rule (TCONC), we obtain $\Gamma \vdash \mathcal{E}'[P_0] \triangleright \Delta_1$ and $\Gamma \vdash P'' \triangleright \Delta_2$, where $\Delta_1, \Delta_2 = \Delta$. By applying rule (CTXT), $P_0 \rightarrow_{p_0} P'_0$ implies $\mathcal{E}'[P_0] \rightarrow_{p'} \mathcal{E}'[P'_0]$, where $p' = p_0 \cdot \text{nextProc}(P_0) / \text{nextProc}(\mathcal{E}'[P_0])$. By induction, $\Gamma \vdash \mathcal{E}'[P'_0] \triangleright \Delta'_1$, where $\Delta'_1 = \Delta_1$ or $\Delta_1 \Rightarrow_{\iota} \Delta'_1$ and $p \cdot \text{nextProc}(\mathcal{E}'[P'_0]) \in \iota$. By applying rule (CTXT), $\mathcal{E}'[P_0] \rightarrow_{p'} \mathcal{E}'[P'_0]$ implies $\mathcal{E}[P_0] \rightarrow_p \mathcal{E}[P'_0]$, where $p = p' \cdot \text{nextProc}(\mathcal{E}'[P_0]) / \text{nextProc}(\mathcal{E}[P_0])$. By using the rule (TCONC), we obtain $\Gamma \vdash \mathcal{E}'[P'_0] \mid P'' \triangleright \Delta'$, where $\Delta' = \Delta$ or $\Delta \Rightarrow_{\iota} \Delta' = \Delta'_1, \Delta_2$, and $p \cdot \text{nextProc}(\mathcal{E}'[P_0] \mid P'') = p' \cdot \text{nextProc}(\mathcal{E}'[P'_0]) = p_0 \cdot \text{nextProc}(P_0) \in \iota$, as required.

– The remaining cases are proved in a similar manner.

Corollary 1 (Type safety). *If $\emptyset \vdash P \triangleright \emptyset$ and $P \rightarrow_p^* P'$, then P' has no error.*

As in [10], an *annotated* process P is the result of annotating the bound names of P , e.g. $(\nu s : G)P$ and $s?(x : S)P$. It is worth mentioning that some rules which do not involve variables are the same as the ones of Table 5. Note that type checking of annotated processes is decidable, whenever the safety property is decidable. This follows because typings have finite-state transition systems.

Theorem 3. *Given an annotated process $(\nu s : G)P$ and a sorting Γ , it is decidable whether $\Gamma \vdash (\nu s : G)P \triangleright \emptyset$ is derivable or not.*

As in the most approaches on multiparty session types, our approach ensures that a typed ensemble of processes interacting on a single annotated session (i.e. a typed $(\nu s : G)(P_1 \mid \dots \mid P_n)$ with each P_i ($1 \leq i \leq n$) interacting only on $s[r_i]$) is deadlock-free. The deadlock freedom property in the presence of multiple interleaved sessions was also studied in [6]. In what follows, $Q \not\vdash$ means that the process Q is not able to evolve by means of any rule (we say that Q is a stuck process).

Theorem 4 (deadlock freedom). *Let $\emptyset \vdash P \triangleright \emptyset$, where $P \equiv (\nu s : G)(P_1 \mid \dots \mid P_n)$ and each P_i ($1 \leq i \leq n$) interacts only on $s[r_i]$ of type T_i , where $T_i = G \upharpoonright r_i$. Then P is deadlock-free: i.e., $P \rightarrow_p^* P' \not\vdash$ implies $P' \equiv \mathbf{0}$.*

As it can be noticed from the rules of Table 5, the obtained types are not unique. This is due to the fact that we use probability intervals allowing the

processes to be considered well-typed within the interval defined by lower and upper probabilities. Having several types for the same process, we can obtain some refinements of these typings. Inspired by [4], we use an *erase* function that, when applied to a process or a type, removes its probability annotations. The operational semantics without probabilities is similar with the one from Table 2. As probabilities do not influence the evolution, the following two results hold.

Proposition 1. *If $P \rightarrow_p P'$, then $\text{erase}(P) \rightarrow \text{erase}(P')$.*

Proof. By induction on the derivation $P \rightarrow_p P'$. There is a case for each operational semantics rule from Table 2.

– Case (COM):

$$s[r_1][r_2] \oplus_{j \in J} \langle p_j : l_j(v_j); P_j \rangle \mid s[r_2][r_1] \&_{i \in I} \{l_i(x_i); P'_i\} \rightarrow_{p_k} P_k \mid P'_k \{\tilde{v}_k / \tilde{x}_k\}.$$

Using the definition of the *erase* function, we get that

$$\begin{aligned} \text{erase}(P) &= \text{erase}(s[r_1][r_2] \oplus_{j \in J} \langle p_j : l_j(v_j); P_j \rangle \mid s[r_2][r_1] \&_{i \in I} \{l_i(x_i); P'_i\}) \\ &= \text{erase}(s[r_1][r_2] \oplus_{j \in J} \langle p_j : l_j(v_j); P_j \rangle) \mid \text{erase}(s[r_2][r_1] \&_{i \in I} \{l_i(x_i); P'_i\}) \\ &= s[r_1][r_2] \oplus_{j \in J} \langle l_j(v_j); \text{erase}(P_j) \rangle \mid s[r_2][r_1] \&_{i \in I} \{l_i(x_i); \text{erase}(P'_i)\} \end{aligned}$$

By using rule (SCOM), it holds that

$$\text{erase}(P) \rightarrow \text{erase}(P_k) \mid \text{erase}(P'_k) \{\tilde{v}_k / \tilde{x}_k\}.$$

Due to the fact that $\text{erase}(P') = \text{erase}(P_k \mid P'_k \{\tilde{v}_k / \tilde{x}_k\}) = \text{erase}(P'_k) \{\tilde{v}_k / \tilde{x}_k\}$, then it results that $\text{erase}(P) \rightarrow \text{erase}(P')$ – as required.

The remaining cases are proved in a similar manner.

Proposition 2. *If $\text{erase}(P) \rightarrow \text{erase}(P')$, then exists p such that $P \rightarrow_p P'$.*

It holds also that a process remains well typed after removing probabilities in processes and probability intervals in types.

Proposition 3. *If $\Gamma \vdash P \triangleright \Delta$, then $\Gamma \vdash \text{erase}(P) \triangleright \text{erase}(\Delta)$.*

Proof. The proof follows by induction on the structure of process P , and by using the rules of Table 5.

Considering $\text{erase}(\Delta_1) = \text{erase}(\Delta_2)$, we define the intersection of two typings:

$$\begin{aligned} \Delta_1 \sqcap \Delta_2 &= \{s[r] : (T_1 \sqcap T_2) \mid s[r] : T_1 \in \Delta_1 \text{ and } s[r] : T_2 \in \Delta_2\}, \text{ where} \\ T_1 \sqcap T_2 &= \begin{cases} r_1 \oplus_{i \in I} (\iota_{1i} \cap \iota_{2i}) : !l_i \langle S_i \rangle . (T_{1i} \sqcap T_{2i}) & \text{if } T_j = r_1 \oplus_{i \in I} \iota_{ji} : !l_i \langle S_i \rangle . T_{ji} \\ & \iota_{1i} \cap \iota_{2i} \neq \emptyset, \text{ with } j \in \{1, 2\} \\ r_1 \&_{i \in I} ?l_i(S_i) . (T_{1i} \sqcap T_{2i}) & \text{if } T_j = r_1 \&_{i \in I} ?l_i(S_i) . T_{ji} \\ & \text{with } j \in \{1, 2\} \\ \mu t . (T_{1i} \sqcap T_{2i}) & \text{if } T_1 = \mu t . T_{1i} \text{ and } T_2 = \mu t . T_{2i} \\ T_1 & \text{if } T_1 = T_2 = t \text{ or } T_1 = T_2 = \text{end} \\ \text{undefined} & \text{otherwise.} \end{cases} \end{aligned}$$

Note that since the intersection of intervals is associative, then also the intersection of typings is associative.

Proposition 4. $(\Delta_1 \sqcap \Delta_2) \sqcap \Delta_3 = \Delta_1 \sqcap (\Delta_2 \sqcap \Delta_3)$.

Proof. By induction on the structure of the local types appearing in Δ_1 , Δ_2 and Δ_3 , considered whenever their intersection is defined. Assume that $s[r_i] : T_j \in \Delta_j$ for $1 \leq j \leq 3$. Note that if one of T_j (for $1 \leq j \leq 3$) is **end** or t , then we should have $T_1 = T_2 = T_3$, and so trivially conclude the proof. Otherwise, we have the following cases:

- $T_j = r \oplus_{i \in I} \iota_{ji} : !i \langle S_i \rangle . T_{ji}$, for $1 \leq j \leq 3$.

By induction and by the associativity of interval intersections, it holds that $(T_1 \sqcap T_2) \sqcap T_3 = r \oplus_{i \in I} ((\iota_{1i} \cap \iota_{2i}) \cap \iota_{3i}) : !i \langle S_i \rangle . ((T_{1i} \sqcap T_{2i}) \sqcap T_{3i}) = r \oplus_{i \in I} (\iota_{1i} \cap (\iota_{2i} \cap \iota_{3i})) : !i \langle S_i \rangle . (T_{1i} \sqcap (T_{2i} \sqcap T_{3i})) = T_1 \sqcap (T_2 \sqcap T_3)$, as requested.

- $T_j = r \&_{i \in I} ?l_i(S_i) . T_{ji}$, for $1 \leq j \leq 3$.

By induction, it holds that $(T_1 \sqcap T_2) \sqcap T_3 = r \&_{i \in I} l_i(S_i) . ((T_{1i} \sqcap T_{2i}) \sqcap T_{3i}) = r \&_{i \in I} l_i(S_i) . (T_{1i} \sqcap (T_{2i} \sqcap T_{3i})) = T_1 \sqcap (T_2 \sqcap T_3)$, as requested.

- $T_j = \mu t . T'_j$, for $1 \leq j \leq 3$.

By induction, it holds that $(T_1 \sqcap T_2) \sqcap T_3 = \mu t . (T'_1 \sqcap T'_2) \sqcap T'_3 = \mu t . T'_1 \sqcap (T'_2 \sqcap T'_3) = T_1 \sqcap (T_2 \sqcap T_3)$, as requested.

It is worth noting that a process well-typed with different types still remains well-typed whenever we consider the intersection of typings Δ_1 and Δ_2 . This means that if for an action with probability p there exists two well-formed types with different probability intervals ι_1 and ι_2 , then the action remains well-type whenever we consider the intersection of the intervals ι_1 and ι_2 .

Theorem 5. If $\Gamma \vdash P \triangleright \Delta_1$, $\Gamma \vdash P \triangleright \Delta_2$ and $\text{erase}(\Delta_1) = \text{erase}(\Delta_2)$, then $\Gamma \vdash P \triangleright \Delta_1 \sqcap \Delta_2$.

Proof. The proof follows by induction on the structure of process P and by using the rules of Table 5.

By defining the union operator $\sqcup$ similarly, we get the next two results.

Proposition 5. $(\Delta_1 \sqcup \Delta_2) \sqcup \Delta_3 = \Delta_1 \sqcup (\Delta_2 \sqcup \Delta_3)$

Theorem 6. If $\Gamma \vdash P \triangleright \Delta_1$, and exists Δ_2 such that $\text{erase}(\Delta_1) = \text{erase}(\Delta_2)$, then $\Gamma \vdash P \triangleright \Delta_1 \sqcup \Delta_2$.

In what follows we associate to any execution path (starting from a process P and leading to a process Q), an execution probability of going along this path. This is computed by using the operational semantics presented in Table 2.

Definition 4 (evolution paths). A sequence of evolution transitions provides an evolution path $ep = P_0 \rightarrow_{p_1} P_1 \dots \rightarrow_{p_k} P_k$. Two evolution paths $ep = P_0 \rightarrow_{p_1} P_1 \dots \rightarrow_{p_k} P_k$ and $ep' = P'_0 \rightarrow_{p'_1} P'_1 \dots \rightarrow_{p'_{k'}} P'_{k'}$ are identical if $k = k'$, and for all $t \in \{0, \dots, k\}$ we have $P_t \equiv P'_t$ and $p_t = p'_t$.

Definition 5 (evolution probability). The probability to reach a process P_j starting from a process P_i (with $i < j$) along the evolution path $ep = P_0 \rightarrow_{p_1} P_1 \dots \rightarrow_{p_k} P_k$ is denoted by $\text{prob}(ep, P_i, P_j) = p_{i+1} * \dots * p_j$. The evolution probability to reach P_j starting from a process P_i (with $i < j$) taking all the possible evolution paths (without considering identical paths) is denoted by $\text{prob}(P_i, P_j) = \sum_{ep} \text{prob}(ep, P_i, P_j)$.

For any process P we can define the sets $\text{Reach}_k(P)$ of processes reached in k steps starting from P by using the rules of Table 2.

Definition 6 (reachable sets). $\text{Reach}_1(P) = \{Q \mid P \rightarrow_p Q\}$. For $k \geq 2$, $\text{Reach}_k(P) = \{Q \mid Q \in \text{Reach}_{k-1}(P) \ \&\ Q \not\rightarrow\} \cup \{R \mid Q \in \text{Reach}_{k-1}(P) \ \&\ Q \rightarrow_p R\}$.

The following result ensures that for each well-typed restricted process, the sum of the probabilities of all possible transitions is 1. The restriction ensures that the types are derived from some well-formed global types.

Theorem 7. Given a well-typed process $(\nu s)P$ such that $\text{Reach}_k((\nu s)P) \neq \emptyset$ for $k \geq 1$, then $\sum_{Q \in \text{Reach}_k(P)} \text{prob}(P, Q) = 1$.

Proof. By induction on the number k of the evolution steps.

– Case $k = 1$.

By induction on the structure of P , namely on the value of $\text{nextProc}((\nu s)P)$.

- $\text{nextProc}((\nu s)P) = 0$. We have only one subcase:
 - * $P = \text{end}$. Since $(\nu s)P = \text{end}$, then $\text{Reach}_1((\nu s)P) = \emptyset$, and the sum is not computed.
 - * Since $(\nu s)P$ is well-typed this means that $P \neq c[r] \oplus_{i \in I} \langle p_i : l_i(v_i); P_i \rangle$ and also that $P \neq c[r] \&_{i \in I} \{l_i(x_i); P_i\}$.
- $\text{nextProc}((\nu s)P) = 1$. We have two subcases:
 - * $P = (\text{def } X(\tilde{x}) = P' \text{ in } (X(\tilde{v})|R))$ meaning that the rules (CALL) and (CTXT) are applied, and $(\nu s)P$ evolves with probability 1 to $(\nu s)(\text{def } X(\tilde{x}) = P' \text{ in } (P'\{\tilde{v}/\tilde{x}\}|R))$. This means that $\sum_{Q \in \text{Reach}_1((\nu s)P)} \text{prob}((\nu s)P, Q) = 1$ (as desired).
 - * $P = P_1 \mid P_2$, where $P_1 = s[r_1][r_2] \oplus_{j \in J} \langle p_j : l_j(v_j); P_j \rangle$ and $P_2 = s[r_2][r_1] \&_{i \in I} \{l_i(x_i); P'_i\}$ meaning that the rules (COM) and (CTXT) are applied, and $(\nu s)P$ evolves with probability p_k (with $k \in J$) to $(\nu s)(P_k \mid P'_k\{\tilde{v}_k/\tilde{x}_k\})$. From the definition of P_1 , it follows that $\sum_{k \in J} p_k = 1$. Thus, $\sum_{Q \in \text{Reach}_1((\nu s)P)} \text{prob}((\nu s)P, Q) = \sum_{k \in J} p_k = 1$ (as desired).
- $\text{nextProc}((\nu s)P) = l$, with $l > 1$. In this case exists P_1 such that $(\nu s)P = \mathcal{E}[P_1]$, $\text{nextProc}(\mathcal{E}) = l - 1$ and $\text{nextProc}(P_1) = 1$. Since $(\nu s)P$ is well-typed this means that $\text{Reach}_1((\nu s)P) = \text{Reach}_1(\mathcal{E}) + \text{Reach}_1(P_1)$. Thus,

$$\begin{aligned}
 & \sum_{Q \in \text{Reach}_1((\nu s)P)} \text{prob}((\nu s)P, Q) \\
 &= \sum_{Q \in \text{Reach}_1(\mathcal{E})} \text{prob}((\nu s)P, Q) + \sum_{Q \in \text{Reach}_1(P_1)} \text{prob}((\nu s)P, Q) \\
 &= \left(\sum_{1 \leq i \leq \text{nextProc}(\mathcal{E})} \frac{1}{\text{nextProc}((\nu s)P)} \right) + \frac{1}{\text{nextProc}((\nu s)P)} \\
 &= \frac{l-1}{l} + \frac{1}{l} = 1 \text{ (as desired). The normalization appears due to the application of the (CTXT) rule.}
 \end{aligned}$$

- Case $k > 1$. Since $\text{Reach}_k((\nu s)P) \neq \emptyset$, this means that $\text{Reach}_{k-1}((\nu s)P) \neq \emptyset$.

By considering the inductive step, we obtain that

$\sum_{Q \in \text{Reach}_{k-1}((\nu s)P)} \text{prob}((\nu s)P, Q) = 1$. Let us consider that another evolution step is performed, namely k steps starting from process $(\nu s)P$. This means that $\sum_{Q \in \text{Reach}_k((\nu s)P)} \text{prob}((\nu s)P, Q)$ can be broken into two parts: one computing the probability for the first $(k - 1)$ steps, and another one using the probabilities obtained in the additional step. Thus, it holds that $\sum_{Q \in \text{Reach}_k((\nu s)P)} \text{prob}((\nu s)P, Q) = \sum_{Q_1 \in \text{Reach}_{k-1}((\nu s)P)} (\text{prob}((\nu s)P, Q_1) * \sum_{Q \in \text{Reach}_1(Q_1)} \text{prob}(Q_1, Q))$.

Since we have $\sum_{Q \in \text{Reach}_1(Q_1)} \text{prob}(Q_1, Q) = 1$, then

$$\sum_{Q \in \text{Reach}_k((\nu s)P)} \text{prob}((\nu s)P, Q) = \sum_{Q_1 \in \text{Reach}_{k-1}((\nu s)P)} \text{prob}((\nu s)P, Q_1).$$

Also, since we have $\sum_{Q_1 \in \text{Reach}_{k-1}((\nu s)P)} \text{prob}((\nu s)P, Q_1) = 1$, then

$$\sum_{Q \in \text{Reach}_k((\nu s)P)} \text{prob}((\nu s)P, Q) = 1, \text{ as required.}$$

5 Conclusion and Related Work

In this paper we have introduced and studied interval probability for multiparty session types by extending and revising our previous approaches presented in [1] and [2]; comparing to [1], in this paper we use simpler syntax and semantics, and added new results. The current approach is a probabilistic extension of the process calculus presented in [16], allowing both nondeterministic external choices and probabilistic internal choices. We defined probabilistic multiparty session types able to describe the structure of the communications by using probability intervals given in terms of lower and upper probabilities. Moreover, we have defined and studied a typing system extending the multiparty session types with probability intervals. The new typing system has several properties and features. It preserves the classical typing properties; additionally, it is specified in such a way to satisfy the axioms of the standard probability theory and of the interval probability theory. To illustrate our calculus, we considered an example inspired by survey polls in which the probabilities are provided by a well-defined numerical scale for responses.

A probabilistic variant of binary session types that consider fixed probabilities in types was presented in [11]. We are not aware of other approaches using probabilities in types even though the probabilistic models in concurrency have been intensively investigated. There is an extensive literature on probabilistic process algebras (see [3, 15] and references therein). In [19], types are used in probabilistic π -calculus to identify a class of nondeterministic probabilistic behaviours which can preserve the compositionality of the parallel operator in the framework of event structures. The authors claim to perform an initial step towards a good typing discipline for probabilistic name passing by employing Segala automata [18] and probabilistic event structures. In comparison with that work, here we simplify the approach and work directly with processes, giving a probabilistic typing in the context of multiparty session types. Additionally, we introduce the interval probability in the framework of multiparty session types.

References

1. Aman, B., Ciobanu, G.: Probabilities in session types. In: Marin, M., Craciun, A. (eds.) *Proceedings Third Symposium on Working Formal Methods, FROM 2019*, Timișoara, Romania, 3–5 September 2019. EPTCS, vol. 303, pp. 92–106 (2019). <https://doi.org/10.4204/EPTCS.303.7>
2. Aman, B., Ciobanu, G.: Imprecise probability for multiparty session types in process algebra. *CoRR* abs/2002.08175 (2020)
3. Bernardo, M., De Nicola, R., Loretì, M.: Revisiting bisimilarity and its modal logic for nondeterministic and probabilistic processes. *Acta Informatica* **52**(1), 61–106 (2014). <https://doi.org/10.1007/s00236-014-0210-1>
4. Bocchi, L., Honda, K., Tuosto, E., Yoshida, N.: A theory of design-by-contract for distributed multiparty interactions. In: Gastin, P., Laroussinie, F. (eds.) *CONCUR 2010. LNCS*, vol. 6269, pp. 162–176. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15375-4_12
5. de Campos, L.M., Huete, J.F., Moral, S.: Probability intervals: a tool for uncertain reasoning. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* **2**(2), 167–196 (1994). <https://doi.org/10.1142/S0218488594000146>
6. Coppo, M., Dezani-Ciancaglini, M., Yoshida, N., Padovani, L.: Global progress for dynamically interleaved multiparty sessions. *Math. Struct. Comput. Sci.* **26**(2), 238–302 (2016). <https://doi.org/10.1017/S0960129514000188>
7. Dempster, A.P.: Upper and lower probabilities induced by a multivalued mapping. In: Yager, R.R., Liu, L. (eds.) *Classic Works of the Dempster-Shafer Theory of Belief Functions, Studies in Fuzziness and Soft Computing*, vol. 219, pp. 57–72. Springer (2008). https://doi.org/10.1007/978-3-540-44792-4_3
8. Ghilezan, S., Jaksic, S., Pantovic, J., Scalas, A., Yoshida, N.: Precise subtyping for synchronous multiparty sessions. *J. Logic Algebraic Meth. Program.* **104**, 127–173 (2019). <https://doi.org/10.1016/j.jlamp.2018.12.002>
9. Halpern, J.Y.: *Reasoning About Uncertainty*. MIT Press, Cambridge (2005)
10. Honda, K., Yoshida, N., Carbone, M.: Multiparty asynchronous session types. *J. ACM* **63**(1), 9:1–9:67 (2016). <https://doi.org/10.1145/2827695>
11. Inverso, O., Melgratti, H.C., Padovani, L., Trubiani, C., Tuosto, E.: Probabilistic analysis of binary sessions. In: Konnov, I., Kovács, L. (eds.) *31st International Conference on Concurrency Theory, CONCUR 2020. LIPIcs*, vol. 171, pp. 14:1–14:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2020). <https://doi.org/10.4230/LIPIcs.CONCUR.2020.14>
12. Keynes, J.M.: *A Treatise on Probability*. Macmillan And Co. (1921)
13. Kolmogorov, A.: *Foundations of the Theory of Probability*. Chelsea Pub Co (1950)
14. Levi, I.: On indeterminate probabilities. *J. Philos.* **71**(13), 391–418 (1974). <https://doi.org/10.2307/2025161>
15. López, N., Núñez, M.: An overview of probabilistic process algebras and their equivalences. In: Baier, C., Haverkort, B.R., Hermanns, H., Katoen, J.-P., Siegle, M. (eds.) *Validation of Stochastic Systems. LNCS*, vol. 2925, pp. 89–123. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24611-4_3
16. Scalas, A., Dardha, O., Hu, R., Yoshida, N.: A linear decomposition of multiparty sessions for safe distributed programming. In: Müller, P. (ed.) *31st European Conference on Object-Oriented Programming, ECOOP 2017. LIPIcs*, vol. 74, pp. 24:1–24:31. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2017). <https://doi.org/10.4230/LIPIcs.ECOOP.2017.24>

17. Scalas, A., Yoshida, N.: Less is more: multiparty session types revisited. *Proc. ACM Program. Lang.* **3**(POPL), 301–3029 (2019). <https://doi.org/10.1145/3290343>
18. Segala, R., Lynch, N.A.: Probabilistic simulations for probabilistic processes. *Nord. J. Comput.* **2**(2), 250–273 (1995)
19. Varacca, D., Yoshida, N.: Probabilistic pi-calculus and event structures. *Electron. Notes Theor. Comput. Sci.* **190**(3), 147–166 (2007). <https://doi.org/10.1016/j.entcs.2007.07.009>
20. Wallsten, T.S., Budescu, D.V., Zwick, R., Kemp, S.M.: Preferences and reasons for communicating probabilistic information in verbal or numerical terms. *Bull. Psychon. Soc.* **31**(2), 135–138 (1993). <https://doi.org/10.3758/BF03334162>
21. Weichselberger, K.: The theory of interval-probability as a unifying concept for uncertainty. *Int. J. Approximate Reasoning* **24**(2–3), 149–170 (2000). [https://doi.org/10.1016/S0888-613X\(00\)00032-3](https://doi.org/10.1016/S0888-613X(00)00032-3)
22. Wolfenson, M., Fine, T.L.: Bayes-like decision making with upper and lower probabilities. *J. Am. Stat. Assoc.* **77**(377), 80–88 (1982). <https://doi.org/10.1080/01621459.1982.10477769>



Combinatorial Flows as Bicolored Atomic Flows

Giti Omidvar^(✉) and Lutz Straßburger

Inria Saclay and École Polytechnique Paris, Palaiseau, France

giti.omidvar@inria.fr

Abstract. We introduce combinatorial flows as a graphical representation of proofs. They can be seen as a generalization of atomic flows on one side and of combinatorial proofs on the other side. From atomic flows, introduced by Guglielmi and Gundersen, they inherit the close correspondence with open deduction and the possibility of tracing the occurrences of atoms in a derivation. From combinatorial proofs, introduced by Hughes, they inherit the correctness criterion that allows to reconstruct the derivation from the flow. In fact, combinatorial flows form a proof system in the sense of Cook and Reckhow. We show how to translate between open deduction derivations and combinatorial flows, and we show how they are related to combinatorial proofs with cuts.

Keywords: Proof identity · Proof invariants · Combinatorial flows · Open deduction

1 Introduction

The question of when two proofs are the same is older than proof theory itself, as it has been posed by Hilbert [31] as early as 1900. Nowadays there are essentially two approaches to the problem. The first is to find suitable proof transformations and postulate that two proofs are the same if they can be transformed into each other using these transformations. This can be achieved via proof normalization [24] or rule permutations [21]. The second way is to define suitable canonical proof representations. The most prominent examples are λ -terms [10], proof nets [9], and combinatorial proofs [18].

A recurring theme in this second setting is the idea to trace the occurrences of formulas or atoms inside the proofs. It has first been used by Kelly and Maclane in [20] (via *coherence graphs*) to determine the identity of morphisms in a category. This idea has then been rediscovered in Girard's *proof nets* [7, 9] and is also used nowadays in *string diagrams* [27]. All these notions—proof nets, coherence graphs, string diagrams—work remarkably well in the linear setting, where no contraction or weakening is present. Indeed, proof nets form a canonical representation for multiplicative linear logic (MLL) [9].

In a classical setting, where contraction and weakening are present, the idea of tracing formulas in a derivation has first been used by Buss [3] in the form of *logical flow graphs*, which have later been studied by Carbone [4, 5] to investigate the relation between cuts and contractions and cycles in these flow graphs.

To get a more precise grip on these cycles—which do not exist in the linear setting—and to eliminate them, Guglielmi and Gundersen developed *atomic flows* [11] as a refinement of logical flow graphs, by only tracing the atoms and by completely detaching the flow from the derivation. In this way, atomic flows are visually closer to coherence graphs and string diagrams, they can be used as invariants for proofs and they can be used to devise normalization procedures for deep inference derivations [11, 13, 32]. However, atomic flows have two major drawbacks:

1. *We cannot read back a proof from an atomic flow.* They lose too much information about the proof. Not only do they *not* form a canonical proof representation, there is also no polynomial correctness criterion.¹ Therefore they do not form a proof system in the sense of Cook and Reckhow [6].
2. *Yanking is not possible in atomic flows.* One of the main advantages of coherence graphs or string diagrams is that they can abstract away from superfluous “bends”. This is indicated on the left below, where we simply can “pull” the ends of the wire. In MLL, this process corresponds to cut elimination [17].

$$\text{wavy blue line} = \text{vertical line} \quad \text{vs.} \quad \text{wavy black line} \neq \text{vertical line} \quad (1)$$

In atomic flows, this situation is represented as on the right above, where we cannot straighten the edge. The reason lies in the interference of contraction with cut elimination.

The purpose of this paper is to address these two problems. In fact, the first problem has already been solved by *combinatorial proofs* [18], but for too high a price: a total separation of the linear part and the resource management part of the proof. Performing this separation is as expensive as cut elimination, and therefore leads to a size explosion.

Our idea here is to use two colors in the atomic flows, to distinguish between the linear parts that can be yanked (blue) and the resource management parts that cannot be yanked (purple). This is similar to what happens in combinatorial proofs, but is less restrictive, as the two parts can be mixed—there is no global separation between the linear part and the resource management. In fact, what we present here can be seen as a merging of atomic flows and combinatorial proofs, and we use the term *combinatorial flows*².

Special care has to be taken with respect to the units, as atomic flows and combinatorial proofs are unit-free.³ However, in order to establish a proper correspondence with the deep inference proof system, units are necessary.

This paper is organized as follows. In the next two sections we recall basic notions of deep inference and show how derivations are translated into objects that we call *preflows*, as the correctness criterion is not yet established. To present the correctness

¹ Das has shown in [8] that no such criterion is possible, under the assumption that integer factoring is hard for $P/poly$.

² This term has already been used in [29] for compositions of combinatorial proofs. Here we are more general: the components of our combinatorial flows are not combinatorial proofs.

³ Combinatorial proofs can be presented with units [18] but then the units are treated as atoms.

criterion, we recall in Sect. 4 Retoré's *RB-cographs*. Then, in Sects. 5 and 6 we define the correctness for the linear parts and the resource management parts, respectively. Finally, in Sect. 7 we define combinatorial flows as correct preflows. In Sect. 8, we introduce *purification* which is a form of normalization via rewriting that simplifies the flow by eliminating irrelevant parts. Finally, in Sect. 9, we establish the relation between our combinatorial flows and Hughes' combinatorial proof with cuts, as they are presented in [19].

2 Preliminaries on Open Deduction

Open deduction [12] is a deep inference formalism [14], allowing to write derivations in a way such that the same operations that are used to build formulas from atoms are also used to build derivations from inference rules.

We define the *formulas* of classical propositional logic, denoted by $A, B, \dots$ to be generated from a countable set of propositional variables $\{a, b, \dots\}$ and their negations $\{\bar{a}, \bar{b}, \dots\}$ with the following grammar:

$$A, B := t \mid f \mid a \mid \bar{a} \mid A \vee B \mid A \wedge B$$

Negation can be also extended to all formulas via De Morgan laws:

$$\bar{t} = f \quad \bar{f} = t \quad \bar{\bar{a}} = a \quad \overline{A \wedge B} = \bar{A} \vee \bar{B} \quad \overline{A \vee B} = \bar{A} \wedge \bar{B}$$

We define the equivalence relation $\equiv$ on formulas:

$$\begin{array}{llll} A \wedge B \equiv B \wedge A & (A \wedge B) \wedge C \equiv A \wedge (B \wedge C) & A \wedge t \equiv A & t \vee t \equiv t \\ A \vee B \equiv B \vee A & (A \vee B) \vee C \equiv A \vee (B \vee C) & A \vee f \equiv A & f \wedge f \equiv f \end{array} \quad (2)$$

A formula A is a *unit* if $A = t$ or $A = f$. A formula A is *unit-free* if it does not contain any units. A formula A is *pure* if either $A \equiv t$ or $A \equiv f$ or $A \equiv A'$ for some unit-free formula A' . An *atom* is an element in $\mathcal{A} = \{a, b, \dots\} \cup \{\bar{a}, \bar{b}, \dots\}$, and a *sequent* $\Gamma = A_1, \dots, A_n$ is a finite non-empty multiset of formulas.

Figure 1 shows the inference rules of system SKS [2]. They have to be understood as rule schemas, where a can stand for an arbitrary atom and A, B, C, D for arbitrary formulas. We call an *inference system* any such set of inference rules. The rules can be composed to *derivations*, which are defined inductively below, and which are denoted

$\frac{A}{B} \mathcal{D} \parallel \mathbf{S}$ where A is the *premise* and B is the *conclusion* of the derivation $\mathcal{D}$, and $\mathbf{S}$ is the set of inference rules used in $\mathcal{D}$.

1. A formula A is a derivation with premise A and conclusion A .
2. Every inference rule $\frac{A}{B} \rho$ in $\mathbf{S}$, is a derivation with premise A and conclusion B .
3. If $\mathcal{D}_1 \parallel \mathbf{S}$ and $\mathcal{D}_2 \parallel \mathbf{S}$ are derivations, then the compositions $\mathcal{D}_1 \wedge \mathcal{D}_2$ and $\mathcal{D}_1 \vee \mathcal{D}_2$ are derivations and denoted as

$$\frac{\frac{B_1}{A_1} \mathcal{D}_1 \parallel \mathbf{S} \quad \frac{B_2}{A_2} \mathcal{D}_2 \parallel \mathbf{S}}{B_1 \wedge B_2} \quad \text{and} \quad \frac{\frac{B_1}{A_1} \mathcal{D}_1 \parallel \mathbf{S} \quad \frac{B_2}{A_2} \mathcal{D}_2 \parallel \mathbf{S}}{B_1 \vee B_2} \quad \text{respectively.}$$

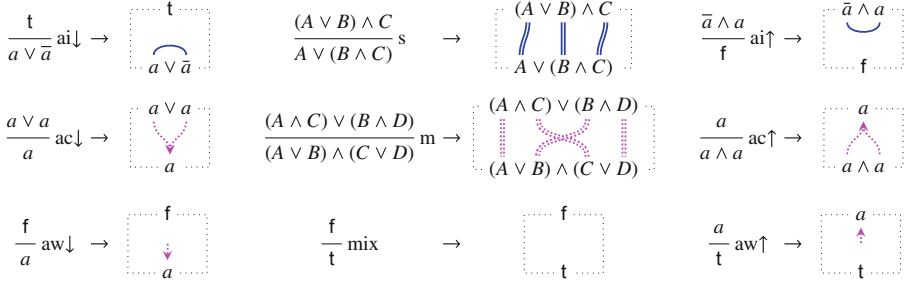


Fig. 1. Inference rules of system SKS and their translation into flowboxes

4. If $\mathcal{D}_1$ is a derivation with premise A_1 and conclusion B_1 , and $\mathcal{D}_2$ is a derivation with premise A_2 and conclusion B_2 , and $A_2 \equiv B_1$, we can compose $\mathcal{D}_1$ and $\mathcal{D}_2$ directly to $\mathcal{D}_1 \circ \mathcal{D}_2$ denoted as:

$$\begin{array}{ccccc}
 A_1 & & A_1 & & A_1 \\
 \mathcal{D}_1 \parallel s & & \mathcal{D}_1 \parallel s & & \mathcal{D}_1 \parallel s \\
 B_1 & & A_2 & & B_1 \\
 \frac{}{A_2} \equiv & \text{or} & & \text{or} & \\
 \mathcal{D}_2 \parallel s & & \mathcal{D}_2 \parallel s & & \mathcal{D}_2 \parallel s \\
 B_2 & & B_2 & & B_2
 \end{array} \quad (3)$$

Figure 2 shows on the left an example of an SKS derivation.

3 From Derivations to Flows

Definition 3.1. We say that a set X is $\mathcal{A}$ -labelled if it is equipped with a **labelling function** $\ell_X: X \rightarrow \mathcal{A}$, mapping each element $x \in X$ to an atom. A binary relation $\mathbb{B} \subseteq X \times X$ on an $\mathcal{A}$ -labelled set X is **well-matched** if it is symmetric and we have that $x\mathbb{B}y$ implies $\ell_X(x) = \ell_X(y)$. We say that $\mathbb{B}$ is **perfectly matched** if it is well-matched and for every $x \in X$ there is exactly one $y \in X$ with $x\mathbb{B}y$.

Definition 3.2. Let $\mathbb{R}$ and $\mathbb{S}$ be binary relations on $X \uplus Y$ and $Y \uplus Z$ where $X \cap Z = \emptyset$. The composition of the two relations $\mathbb{R}$ and $\mathbb{S}$ (denoted as $\mathbb{R} \circ \mathbb{S}$) is a binary relation on $X \uplus Z$ where $(a, c) \in \mathbb{R} \circ \mathbb{S}$ if and only if there exists a sequence $v_1, v_2, \dots, v_n$ of elements in $X \uplus Y \uplus Z$ such that $a = v_1$, $c = v_n$, and for every $i \in \{1, \dots, n-1\}$, we have $v_i \mathbb{R} v_{i+1}$ if i is odd and $v_i \mathbb{S} v_{i+1}$ if i is even.

For a formula A , we write $[A]$ to denote the set of leaves of the formula tree of A . This set is $\mathcal{A}$ -labelled, with the labelling function $\ell_{[A]}: [A] \rightarrow \mathcal{A}$ mapping each leaf of the formula tree to the atom occurring in that position. Note that $A \equiv A'$ implies $[A] = [A']$.

Definition 3.3. A **flowbox** is a triple $\phi = \langle A, B, \mathbb{B}_\phi \rangle$, where A and B are formulas and $\mathbb{B}_\phi$ is a well-matched relation on $[\bar{A}] \uplus [B]$. For flowboxes $\phi = \langle A, B, \mathbb{B}_\phi \rangle$ and $\psi = \langle C, D, \mathbb{B}_\psi \rangle$, we define $\phi \wedge \psi = \langle A \wedge C, B \wedge D, \mathbb{B}_\phi \uplus \mathbb{B}_\psi \rangle$ and $\phi \vee \psi = \langle A \vee C, B \vee D, \mathbb{B}_\phi \uplus \mathbb{B}_\psi \rangle$, which are also flowboxes. If $\phi = \langle A, B_1, \mathbb{B}_\phi \rangle$ and $\psi = \langle B_2, C, \mathbb{B}_\psi \rangle$ are flowboxes with $B_1 \equiv B_2$, then $\phi \circ \psi = \langle A, C, \mathbb{B}_\phi \circ \mathbb{B}_\psi \rangle$ is also a flowbox.

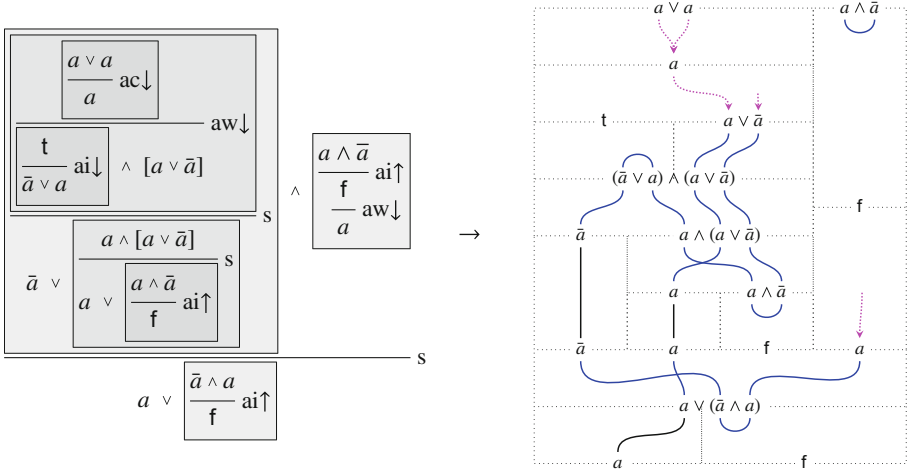


Fig. 2. A derivation from $(a \vee a) \wedge (a \wedge \bar{a})$ to $a \vee f$ and its translation to a preflow.

Example 3.4. Here are some examples of flowboxes. We denote them by writing the formulas on top of each other and indicating the relation $\mathbb{B}$ by edges between the atom occurrences.



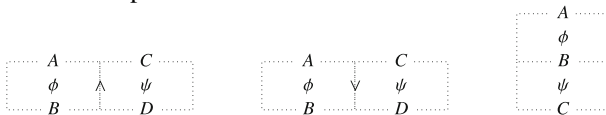
The vertical composition of the first two flowboxes results in the third flowbox.

Observe for each formula there is an *identity* flowbox $\text{id}_A = \langle A, A, \mathbb{B}_{\text{id}} \rangle$, and that the operation $\circ$ on flowboxes is associative with id as unit. In fact, flowboxes form a category and are essentially the same as the B-nets of [22], and the operation $\circ$ corresponds to the cut elimination of B-nets. As already observed in [22], the composition $\circ$ forgets too much information. In general, we cannot recover a derivation from a flowbox. To gain more control, we define a formal composition that keeps the structure.

Definition 3.5. We define the set of *preflows* to be inductively constructed as follows:

1. A flowbox $\phi = \langle A, B, \mathbb{B}_\phi \rangle$ is a preflow $\phi: A \vdash B$ with *premise* A and *conclusion* B .
2. If $\phi: A \vdash B$ and $\psi: C \vdash D$ are preflows, Then their *horizontal compositions* $\phi \otimes \psi: A \wedge C \vdash B \wedge D$ and $\phi \oplus \psi: A \vee C \vdash B \vee D$ are preflows.
3. Let $\phi: A \vdash B_1$ and $\psi: B_2 \vdash C$ be preflows with $B_1 \equiv B_2$, then the *vertical composition* $\phi \odot \psi: A \vdash C$ is a preflow.

These compositions are depicted as follows:



As suggested by this graphical notation, we consider preflows to be equivalent modulo

$$\begin{aligned} (\phi \otimes \psi) \otimes \xi &= \phi \otimes (\psi \otimes \xi) & (\phi \otimes \psi) \odot (\xi \otimes \pi) &= (\phi \odot \xi) \otimes (\psi \odot \pi) \\ (\phi \vee \psi) \vee \xi &= \phi \vee (\psi \vee \xi) & (\phi \vee \psi) \odot (\xi \vee \pi) &= (\phi \odot \xi) \vee (\psi \odot \pi) \end{aligned} \quad (5)$$

We can now translate derivations into preflows by translating inference rules into flowboxes as indicated in Fig. 1—the colors and arrows used in the figure will be explained later—and then use the operations $\otimes$, $\vee$, and $\odot$ to compose them.

Definition 3.6. The *translation* T of a derivation $\mathcal{D}$, denoted as $T(\mathcal{D})$, is the preflow inductively obtained as follows. If $\mathcal{D}$ is a formula A , then its translation is id_A . If $\mathcal{D}$ is a rule instance, then the translation is shown in Fig. 1. Finally, $T(\mathcal{D}_1 \wedge \mathcal{D}_2) = T(\mathcal{D}_1) \otimes T(\mathcal{D}_2)$ and $T(\mathcal{D}_1 \vee \mathcal{D}_2) = T(\mathcal{D}_1) \vee T(\mathcal{D}_2)$ and $T(\mathcal{D}_1 \circ \mathcal{D}_2) = T(\mathcal{D}_1) \odot T(\mathcal{D}_2)$.

Definition 3.7. We define the *collapse* $\llbracket \phi \rrbracket$ of a preflow ϕ to be the flowbox inductively obtained via $\llbracket \phi \otimes \psi \rrbracket = \llbracket \phi \rrbracket \wedge \llbracket \psi \rrbracket$ and $\llbracket \phi \vee \psi \rrbracket = \llbracket \phi \rrbracket \vee \llbracket \psi \rrbracket$ and $\llbracket \phi \odot \psi \rrbracket = \llbracket \phi \rrbracket \circ \llbracket \psi \rrbracket$.

In other words, the collapse “executes” the operations that are used to define preflows.

Example 3.8. The collapse of the preflow in Fig. 2 is the rightmost flowbox in (4)

Clearly, preflows contain too much information—essentially the same as the open deduction derivation—and the collapse forgets too much of it. Atomic flows are somewhere in the middle. However they forget too much (we still cannot reconstruct the derivation) and too little (we cannot yank) at the same time.

To better control the information that can be removed, we propose here a solution that assigns colors (blue and purple) to flowboxes and only allows to collapse then if they have the same color. The idea is that the blue flowboxes encode linear (or multiplicative) behaviour, and therefore blue wires can be yanked. The purple flowboxes encode the resource management (or the additive behaviour) of the proof, and therefore purple wires cannot be yanked.⁴

Furthermore, both kinds of flowboxes obey different correctness criteria. To define these, we need to recall Retoré’s RB-cographs [25, 26].

4 Preliminaries on RB-cographs

An (*undirected*) *graph* $\mathcal{G} = \langle V_{\mathcal{G}}, R_{\mathcal{G}} \rangle$ is a set of vertices $V_{\mathcal{G}}$ accompanied with a binary edge relation $R_{\mathcal{G}} \subseteq V_{\mathcal{G}} \times V_{\mathcal{G}}$ which is irreflexive and symmetric. We omit the index $\mathcal{G}$ in $V_{\mathcal{G}}$ and $R_{\mathcal{G}}$ when it is clear from the context. All graphs in this paper have an *$\mathcal{A}$ -labelled* vertex set. For two graphs $\mathcal{G}$ and $\mathcal{H}$ we define their *disjoint union* as $\mathcal{G} \uplus \mathcal{H} = \langle V_{\mathcal{G}} \cup V_{\mathcal{H}}, R_{\mathcal{G}} \cup R_{\mathcal{H}} \rangle$ and their *join* as $\mathcal{G} \bowtie \mathcal{H} = \langle V_{\mathcal{G}} \cup V_{\mathcal{H}}, R_{\mathcal{G}} \cup R_{\mathcal{H}} \cup \{(u, v), (v, u) \mid u \in V_{\mathcal{G}}, v \in V_{\mathcal{H}}\} \rangle$. The *complement* of $\mathcal{G}$ is $\overline{\mathcal{G}} = \langle V_{\mathcal{G}}, \{(v, u) \mid (v, u) \notin R_{\mathcal{G}} \text{ and } v \neq u\} \rangle$. The labels are preserved for $\mathcal{G} \uplus \mathcal{H}$ and $\mathcal{G} \bowtie \mathcal{H}$ and negated in $\overline{\mathcal{G}}$. A graph $\mathcal{G}$ is a *cograph* if and only if it is constructed from single vertices using the operations $\uplus$ and complement.

⁴ To indicate the color in a flowbox with empty $\mathbb{B}$ (because of weakening), we sometimes draw half arrows to the atoms, as done in Fig. 1.

Definition 4.1. The *graph* a formula A , denoted as $\mathcal{G}(A)$, is defined inductively via $\mathcal{G}(\mathbf{t}) = \mathcal{G}(\mathbf{f}) = \langle \emptyset, \emptyset \rangle$ (the empty graph), $\mathcal{G}(a) = \langle \{\bullet_a\}, \emptyset \rangle$ (a single vertex graph whose vertex is labelled by a), and $\mathcal{G}(B \vee C) = \mathcal{G}(B) \uplus \mathcal{G}(C)$, and $\mathcal{G}(B \wedge C) = \mathcal{G}(B) \bowtie \mathcal{G}(C)$.

This means that for every formula A , we have that $V_{\mathcal{G}(A)} = [A]$, with the same labelling function. We immediately have the following properties of this translation:

Proposition 4.2. A graph $\mathcal{G}$ is a cograph if and only if it is isomorphic to $\mathcal{G}(A)$ for some pure formula A . And for all pure formulas A , we have $\overline{\mathcal{G}(A)} = \mathcal{G}(\overline{A})$.

Proposition 4.3. If A is a unit-free formula, then $A \equiv B$ iff $\mathcal{G}(A) = \mathcal{G}(B)$.

Definition 4.4. An **RB-graph** is a triple $\mathcal{G} = \langle V_{\mathcal{G}}, R_{\mathcal{G}}, B_{\mathcal{G}} \rangle$ such that $\langle V_{\mathcal{G}}, R_{\mathcal{G}} \rangle$ is a labelled graph and $B_{\mathcal{G}}$ is a perfectly matched binary relation on $V_{\mathcal{G}}$. An **RB-cograph** is an RB-graph $\mathcal{G}$ where $\langle V_{\mathcal{G}}, R_{\mathcal{G}} \rangle$ is a cograph.

Definition 4.5. An *alternating elementary cycle* ($\mathfrak{x}$ -cycle) in a RB-graph is a sequence of vertices $x_1, \dots, x_n$ with n even, such that $x_i \neq x_j$ for all $i, j \in \{1, \dots, n\}$ with $i \neq j$, and we have $x_1 R x_2 B x_3 R x_4 \dots x_n B x_1$. A *chord* in an $\mathfrak{x}$ -cycle is an edge $x_i R x_j$ for $i, j \in \{1, \dots, n\}$ that does not participate in the cycle. A *chordless $\mathfrak{x}$ -cycle* is an $\mathfrak{x}$ -cycle without any chords. An RB-graph $\mathcal{G}$ is $\mathfrak{x}$ -acyclic if it has no chordless $\mathfrak{x}$ -cycle.

5 Multiplicative Flows

The basic idea of the correctness criterion for combinatorial flows is the distinction between multiplicative and additive behavior of flowboxes. We begin in this section with the multiplicative part, which is based on Retoré's work on handsome proof nets [26].

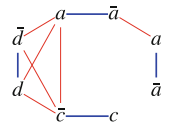
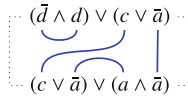
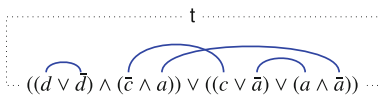
Let $\phi = \langle A, B, B_{\phi} \rangle$ be a flowbox, where B_{ϕ} is perfectly matched. Then we can associate to ϕ the RB-cograph $\mathcal{G}(\phi) = \langle V_{\phi}, R_{\phi}, B_{\phi} \rangle$, where $\langle V_{\phi}, R_{\phi} \rangle$ is $\mathcal{G}(\overline{A} \vee B)$. Observe that $V_{\phi} = V_{\mathcal{G}(\overline{A} \vee B)} = [\overline{A} \vee B] = [\overline{A}] \uplus [B]$, allowing us to use B_{ϕ} in the graph.

Definition 5.1. A flowbox $\phi = \langle A, B, B_{\phi} \rangle$ is *pure* if A and B are pure. A *multiplicative flow* (or *m-flow*) is a pure flowbox $\phi = \langle A, B, B_{\phi} \rangle$, where B_{ϕ} is perfectly matched and $\mathcal{G}(\phi)$ is $\mathfrak{x}$ -acyclic, and we do not have both $A \equiv \mathbf{t}$ and $B \equiv \mathbf{f}$.⁵

Remark 5.2. Note that for each flowbox ϕ with B_{ϕ} being perfectly matched, the graph $\mathcal{G}(\phi)$ is uniquely determined. However, the converse is not true. In particular, if A and B are unit-free formulas and $\phi = \langle A, B, B_{\phi} \rangle$ and $\phi' = \langle \mathbf{t}, \overline{A} \vee B, B_{\phi} \rangle$, then $\mathcal{G}(\phi) = \mathcal{G}(\phi')$.

Proposition 5.3. Let A and B be unit-free formulas. Then $\phi = \langle A, B, B_{\phi} \rangle$ is an m-flow iff $\phi' = \langle \mathbf{t}, \overline{A} \vee B, B_{\phi} \rangle$ is an m-flow.

Example 5.4. Below on the left are two flowboxes that are m-flows and that have the same RB-cograph which is shown on the right below.



⁵ Observe that we do allow $A \equiv \mathbf{t}$ or $B \equiv \mathbf{f}$, just not both at the same time.

When we compose two flowboxes ϕ and ψ with the operations $\wedge, \vee, \circ$ defined in Definition 3.3, then it is clear that the result is perfectly matched if ϕ and ψ are. From the work of Retoré [26] it follows that also the property of being $\mathfrak{x}$ -acyclic is preserved. However, the result does not need to be pure even if ϕ and ψ are (see Example 5.7 below). In particular, if we have m-flows $\phi = \langle t, B, \mathbb{B}_\phi \rangle$ and $\psi = \langle C, D, \mathbb{B}_\psi \rangle$, where C is unit-free, then $\phi \vee \psi$ is not an m-flow because $t \vee C$ is not pure. However, we have the following result:

A

Theorem 5.5. *Let $\mathcal{D} \parallel \{ai\downarrow, ai\uparrow, s, mix\}$ be a derivation. If A and B are pure, then $\llbracket T(\mathcal{D}) \rrbracket$ is an m-flow.*

B

Proof. Every flowbox occurring in $T(\mathcal{D})$ is an m-flow, because $\mathcal{D}$ only contains instances of $ai\downarrow, ai\uparrow, s, mix$. By construction, $\mathbb{B}_{\llbracket T(\mathcal{D}) \rrbracket}$ is perfectly matched, and as already observed above, no $\mathfrak{ae}$ -cycle is introduced in $\mathcal{G}(\llbracket T(\mathcal{D}) \rrbracket)$. Since A and B are pure, it follows that $\llbracket T(\mathcal{D}) \rrbracket$ is an m-flow. $\square$

We also have the converse, which is a consequence of sequentialization of linear logic proof nets.

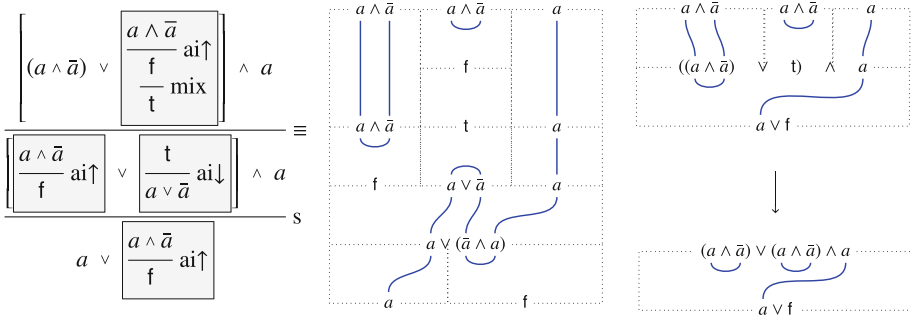
A

Theorem 5.6. *Let $\phi = \langle A, B, \mathbb{B}_\phi \rangle$ be an m-flow. Then there is a derivation $\mathcal{D} \parallel \{ai\downarrow, ai\uparrow, s, mix\}$ with $\llbracket T(\mathcal{D}) \rrbracket = \phi$.*

B

Proof. The simplest way to prove this is using the sequentialization result of Retoré [26] together with the correspondence between sequent calculus and deep inference [14]. A direct sequentialization from proof nets to deep inference derivations can be found in [30]. $\square$

Example 5.7. Below we show on the left a derivation, whose translation to combinatorial flows is shown in the middle. We can naively compose the lower half, but then would obtain flowboxes that are not pure, as shown on the upper right below. Nonetheless, the complete composition is pure and an m-flow (shown on the lower right).



This example also shows a case of yanking.

6 Additive Flows

In the previous section we looked at flows generated by the rules $\text{ai}\downarrow$, $\text{ai}\uparrow$, s , mix . Now we investigate the flows generated by the rules $\text{ac}\downarrow$, $\text{ac}\uparrow$, $\text{aw}\downarrow$, $\text{aw}\uparrow$, m . The main difference is that we do no longer demand that the relation of the flowbox is perfectly matched, but that it is a function with certain properties.

A flowbox $\phi = \langle A, B, \mathbb{B}_\phi \rangle$ is **function-like** if $x\mathbb{B}_\phi y$ implies that either $x \in [A]$ and $y \in [B]$ or $x \in [B]$ and $y \in [A]$, and for every $x \in [A]$, there is a unique y with $x\mathbb{B}_\phi y$. Then $\mathbb{B}_\phi$ defines a function $f_\phi^\downarrow: [A] \rightarrow [B]$, and we write ϕ as $\langle A, B, f_\phi^\downarrow \rangle$.

Similarly, we say that ϕ is **cofunction-like** if $x\mathbb{B}_\phi y$ implies that either $x \in [A]$ and $y \in [B]$ or $x \in [B]$ and $y \in [A]$, and for every $y \in [B]$ there is exactly one $x \in [A]$ with $x\mathbb{B}_\phi y$. In this case we interpret the function $[B] \rightarrow [A]$ defined by $\mathbb{B}_\phi$ as function $f_\phi^\uparrow: [\bar{B}] \rightarrow [\bar{A}]$, and we write ϕ as $\langle A, B, f_\phi^\uparrow \rangle$.

For specifying the desired properties of those functions, let us recall the notion of *skew fibration* [18].

Definition 6.1. Let $\mathcal{G}$ and $\mathcal{H}$ be graphs. A **graph homomorphism** $f: \mathcal{G} \rightarrow \mathcal{H}$ is a mapping $f: V_{\mathcal{G}} \rightarrow V_{\mathcal{H}}$ such that for every $v, w \in V_{\mathcal{G}}$, if $vR_{\mathcal{G}}w$ then $f(v)R_{\mathcal{H}}f(w)$, and for every $v \in V_{\mathcal{G}}$ we have $\ell_{\mathcal{G}}(v) = \ell_{\mathcal{H}}(f(v))$. A **skew fibration** is a graph homomorphism $f: \mathcal{G} \rightarrow \mathcal{H}$ where $\mathcal{H}$ is non-empty, and for every $v \in V_{\mathcal{G}}$ and $w \in V_{\mathcal{H}}$ with $f(v)R_{\mathcal{H}}w$, there exists a vertex z in $\mathcal{G}$ with $vR_{\mathcal{G}}z$ and $(f(z), w) \notin R_{\mathcal{H}}$.

For graph homomorphisms $f_1: \mathcal{G}_1 \rightarrow \mathcal{H}_1$ and $f_2: \mathcal{G}_2 \rightarrow \mathcal{H}_2$ we can define their horizontal compositions $f_1 + f_2: \mathcal{G}_1 \uplus \mathcal{G}_2 \rightarrow \mathcal{H}_1 \uplus \mathcal{H}_2$ and $f_1 \times f_2: \mathcal{G}_1 \bowtie \mathcal{G}_2 \rightarrow \mathcal{H}_1 \bowtie \mathcal{H}_2$, acting componentwise on the vertex set $V_{\mathcal{G}_1} \uplus V_{\mathcal{G}_2}$.

Lemma 6.2. If $f: \mathcal{G} \rightarrow \mathcal{H}$ and $g: \mathcal{I} \rightarrow \mathcal{J}$ are skew fibrations, then so is $f + g: \mathcal{G} \uplus \mathcal{H} \rightarrow \mathcal{I} \uplus \mathcal{J}$. Furthermore, if either $\mathcal{G}_1 \neq \emptyset \neq \mathcal{G}_2$ or $\mathcal{G}_1 = \emptyset = \mathcal{G}_2$ then $f \times g: \mathcal{G} \bowtie \mathcal{H} \rightarrow \mathcal{I} \bowtie \mathcal{J}$ is also a skew fibration.

Proof. Immediate from the definition. $\square$

Lemma 6.3. Let $f: \mathcal{G} \rightarrow \mathcal{H}_1 \uplus \mathcal{H}_2$ be a skew fibration. Then $f = f_1 + f_2$ for skew fibrations $f_1: \mathcal{G}_1 \rightarrow \mathcal{H}_1$ and $f_2: \mathcal{G}_2 \rightarrow \mathcal{H}_2$ with $\mathcal{G} = \mathcal{G}_1 \uplus \mathcal{G}_2$.

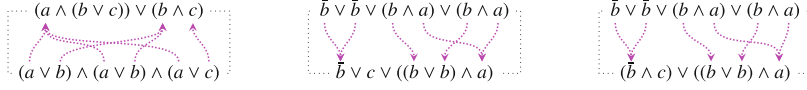
Proof. Immediate from the definition. $\square$

Lemma 6.4. If $f: \mathcal{G} \rightarrow \mathcal{H}$ and $g: \mathcal{H} \rightarrow \mathcal{I}$ are skew fibrations and $\mathcal{G}, \mathcal{H}, \mathcal{I}$ are cographs, then the composition $f \circ g: \mathcal{G} \rightarrow \mathcal{I}$ is a skew fibration.

Proof. See [19] or [28]. $\square$

Definition 6.5. An **$a^\downarrow$ -flow** is a function-like flowbox $\phi = \langle A, B, f_\phi^\downarrow \rangle$ where A and B are pure and $A \neq \mathbf{t}$ and $f_\phi^\downarrow$ is a skew fibration $f_\phi^\downarrow: \mathcal{G}(A) \rightarrow \mathcal{G}(B)$. Similarly, an **$a^\uparrow$ -flow** is a cofunction-like flowbox $\phi = \langle C, D, f_\phi^\uparrow \rangle$ where C and D are pure and $D \neq \mathbf{f}$ and $f_\phi^\uparrow$ is a skew fibration $f_\phi^\uparrow: \mathcal{G}(\bar{D}) \rightarrow \mathcal{G}(\bar{C})$. We call $a^\downarrow$ -flows and $a^\uparrow$ -flows also **additive flows**.

Example 6.6. When drawing flowboxes that are $a^\downarrow$ -flows or $a^\uparrow$ -flows, we use purple arrows to indicate the direction of the functions. Below are three examples. The first one is an $a^\uparrow$ -flow. The second one is an $a^\downarrow$ -flow. The third example is not a skew fibration because in the lower graph there is an edge between the $\bar{b}$ and the c , violating the skew lifting.



When composing flowboxes that are additive flows, we are in a similar situation as for multiplicative flows in the previous section. In the general case, the horizontal composition of skew fibrations is not a skew fibration (see side condition in Lemma 6.2). However, we have an analogous result as for m-flows:

Theorem 6.7. $\begin{matrix} A \\ B \end{matrix}$ Let $\mathcal{D} \parallel \{aw^\downarrow, ac^\downarrow, m\}$ be a derivation. If A and B are pure, then $\llbracket T(\mathcal{D}) \rrbracket$ is an $a^\downarrow$ -flow. Dually, if A and B are pure in $\mathcal{D} \parallel \{aw^\uparrow, ac^\uparrow, m\}$ then $\llbracket T(\mathcal{D}) \rrbracket$ is an $a^\uparrow$ -flow.

Proof. Every flowbox occurring in $T(\mathcal{D})$ is an $a^\downarrow$ -flow (resp. $a^\uparrow$ -flow). We can conclude by a similar argument as for Theorem 5.5. $\square$

As before, we also have the converse.

Theorem 6.8. $\begin{matrix} A \\ B \end{matrix}$ Let $\phi = \langle A, B, f_\phi^\downarrow \rangle$ be an $a^\downarrow$ -flow. Then there is a derivation $\mathcal{D} \parallel \{aw^\downarrow, ac^\downarrow, m\}$ with $\llbracket T(\mathcal{D}) \rrbracket = \phi$. Dually, for every $a^\uparrow$ -flow ψ we have $\mathcal{D} \parallel \{aw^\uparrow, ac^\uparrow, m\}$ with $\llbracket T(\mathcal{D}) \rrbracket = \psi$.

Proof. This follows from [19] together with [2], or more directly from [28]. $\square$

7 Combinatorial Flows

Definition 7.1. A **combinatorial flow** is a preflow $\phi: A \vdash B$ where every flowbox that occurs in ϕ is either a multiplicative flow or an additive flow.

We immediately have the following:

Theorem 7.2. Let $\mathcal{D}$ be a derivation in SKS. Then $T(\mathcal{D})$ is a combinatorial flow.

We can simplify combinatorial flows by “executing” the operations in Definition 3.5. More precisely, a combinatorial flow $\phi: A \vdash B$ is a **simplification** of a $\psi: A \vdash B$ if ϕ is obtained from ψ by collapsing subflows in which all flowboxes have the same type (m-flow, $a^\downarrow$ -flow, or $a^\uparrow$ -flow).

Note that the difference between the collapse (which is uniquely determined) and a simplification (which is not uniquely determined) is that in the former everything is simplified, whereas in the latter the property of being a combinatorial flow is preserved. With this, we can state the converse of Theorem 7.2

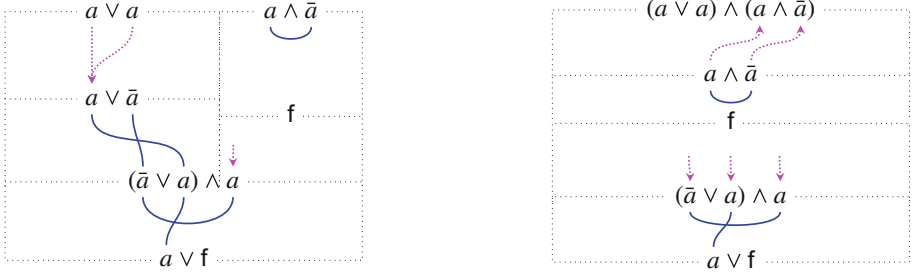


Fig. 3. A simplification of the combinatorial flow in Fig. 2 and its purification

Theorem 7.3. Let $\phi: A \vdash B$ be a combinatorial flow. Then there is a derivation $\mathcal{D} \parallel_{\text{SKS}}$ such that ϕ is a simplification of $\mathbf{T}(\mathcal{D})$. A
B

Proof. This follows immediately from Theorems 5.6 and 6.8. □

The combinatorial flow on the left in Fig. 3 is a simplification of the one in Fig. 2.

Corollary 7.4. Combinatorial flows are sound and complete for classical logic.

Corollary 7.5. Combinatorial flows form a proof system in the sense of Cook and Reckhow [6].

Proof. It can be checked in polynomial time if a preflow is a combinatorial flow. □

Example 7.6. Figure 4 shows a series of combinatorial flows, where we apply “Currying”: flipping some premises to the conclusion or vice versa. Because blue wires can be yanked, this is easily possible, and by Theorem 7.3 there is always a corresponding derivation. However, in the last combinatorial flow in that figure, there are two blue bends that are connected by a purple edge, and that therefore cannot be yanked:



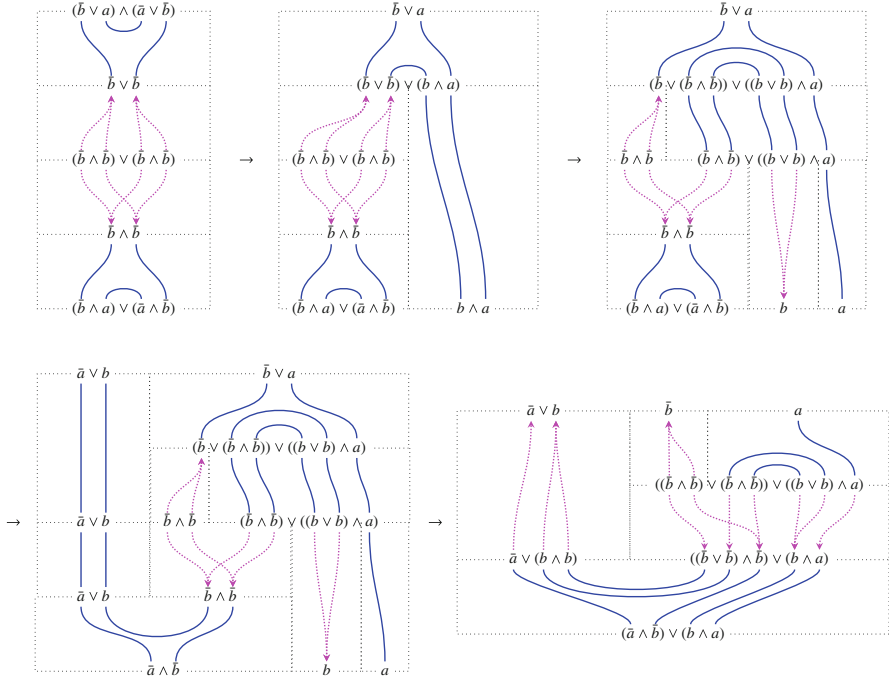


Fig. 4. Example of applying “Currying” to a combinational flow.

8 Purification

We introduced the notion of *pure* because the correctness criteria for additive and multiplicative flows do not work if the formulas are not pure. In the multiplicative case, we do no longer have a canonical representation of an m-flow (see e.g. [1, 17, 23]) and checking equivalence is PSPACE-complete [15]. In the additive case, the horizontal composition can break the skew fibration property, when units are involved. This is the reason for the side condition in Lemma 6.2. With the equivalence $\equiv$, some units occurring in formulas can be removed, but not all. And the presence of these units can block further simplification in combinational flows.

We have $A \wedge f \not\equiv f$ and $A \vee t \not\equiv t$ because otherwise $\equiv$ would change the number of atoms in a formula, and therefore break Propositions 4.2 and 4.3. However, we have the logical equivalences $A \wedge f \iff f$ and $A \vee t \iff t$. Using these equivalences, we can remove all units from a combinational flow, and we call this process *purification*.

Definition 8.1. The *purification* of a formula A , denoted as $\mathbf{p}(A)$ is defined to be the normal form of the rewriting relation $\rightsquigarrow$ below:

$$\begin{array}{llll}
 A \wedge t \rightsquigarrow A & t \wedge A \rightsquigarrow A & A \vee t \rightsquigarrow t & t \vee A \rightsquigarrow t \\
 A \vee f \rightsquigarrow A & f \vee A \rightsquigarrow A & A \wedge f \rightsquigarrow f & f \wedge A \rightsquigarrow f
 \end{array} \tag{6}$$

It is easy to see that this rewriting relation is terminating and confluent, and therefore the purification of a formula is well-defined. The interesting observation is that this rewriting relation can be extended from formulas to combinatorial flows.

Definition 8.2. A *slice* of a combinatorial flow $\phi: A \vdash B$ is a formula C such that $\phi = \phi_1 \odot \phi_2$ for some combinatorial flows $\phi_1: A \vdash C_1$ and $\phi_2: C_2 \vdash B$ with $C_1 \equiv C \equiv C_2$. A combinatorial flow ϕ is *pure* if every slice of ϕ is pure.

Theorem 8.3. For every combinatorial flow $\phi: A \vdash B$, there is a pure combinatorial flow $p(\phi): p(A) \vdash p(B)$.

Proof. First, observe that every flowbox occurring in ϕ is pure, i.e., premise and conclusion can be written as a unit-free formula or a unit. If a slice (recall the equivalences in (5)) is not pure, then it must have a subformula of the shape $A \wedge f$ or $A \vee t$, which is the consequence of the horizontal compositions of flowboxes. We can list all possible cases and design a rewrite system that extends (6) to combinatorial flows. A list of these cases can be found in Fig. 5. Below are two representative cases:

$$\begin{array}{ccc}
 \begin{array}{|c|c|} \hline f & B \\ \hline \varphi & \oplus \\ \hline C & D \\ \hline \end{array} & \rightsquigarrow & \begin{array}{|c|} \hline f \\ \hline \downarrow \\ \hline C \wedge D \\ \hline \end{array}
 \end{array}
 \quad
 \begin{array}{ccc}
 \begin{array}{|c|c|} \hline A & B \\ \hline \varphi & \oplus \\ \hline C & f \\ \hline \end{array} & \rightsquigarrow & \begin{array}{|c|} \hline A \wedge B \\ \hline \downarrow \\ \hline B \\ \hline \downarrow \\ \hline f \\ \hline \end{array}
 \end{array}
 \quad (7)$$

Each step reduces the number of flowboxes in ϕ that have a unit as premise or conclusion. Therefore the rewriting is terminating. Furthermore, the result is pure. $\square$

The combinatorial flow on the right in Fig. 3 is a purification of the one on the left.

9 Combinatorial Flows and Combinatorial Proofs

Definition 9.1. A *combinatorial proof* [18] of a pure formula A is a skew fibration $f: \mathcal{H} \rightarrow \mathcal{G}(A)$ from an $\mathfrak{a}$ -acyclic RB-cograph $\mathcal{H}$ to the graph of A .⁶

Translated to the setting of this paper, a combinatorial proof is the composition $\phi \odot \psi$ of an m -flow $\phi = \langle t, H, \mathbb{B}_\phi \rangle$ and an $a^\downarrow$ -flow $\psi = \langle H, A, f_\psi^\downarrow \rangle$. In other words, combinatorial proofs are a special case of combinatorial flows that make a global separation between the multiplicative and the additive parts of a cut-free proof. In order to deal with cuts, the notion of combinatorial proof has been extended to sequents in [19].

Definition 9.2. A *combinatorial proof with cuts* of a sequent $\Gamma = A_1, \dots, A_n$ of unit-free formulas is a skew fibration $f: \mathcal{H} \rightarrow \mathcal{G}(A_1 \vee \dots \vee A_n \vee (\overline{C_1} \wedge C_1) \vee \dots \vee (\overline{C_k} \wedge C_k))$ from an $\mathfrak{a}$ -acyclic RB-cograph $\mathcal{H}$ to the graph of $\Gamma, C_1 \wedge C_1, \dots, \overline{C_k} \wedge C_k$, where $C_1, \dots, C_k$ are arbitrary unit-free formulas and are called the *cut formulas* of the proof.

⁶ In [18], the units t and f are treated like atoms in the translation to graphs, so that the restriction to pure formulas was not needed. However, this would make composition difficult to define, and for this reason in [19] combinatorial proofs have been restricted to the unit-free setting.

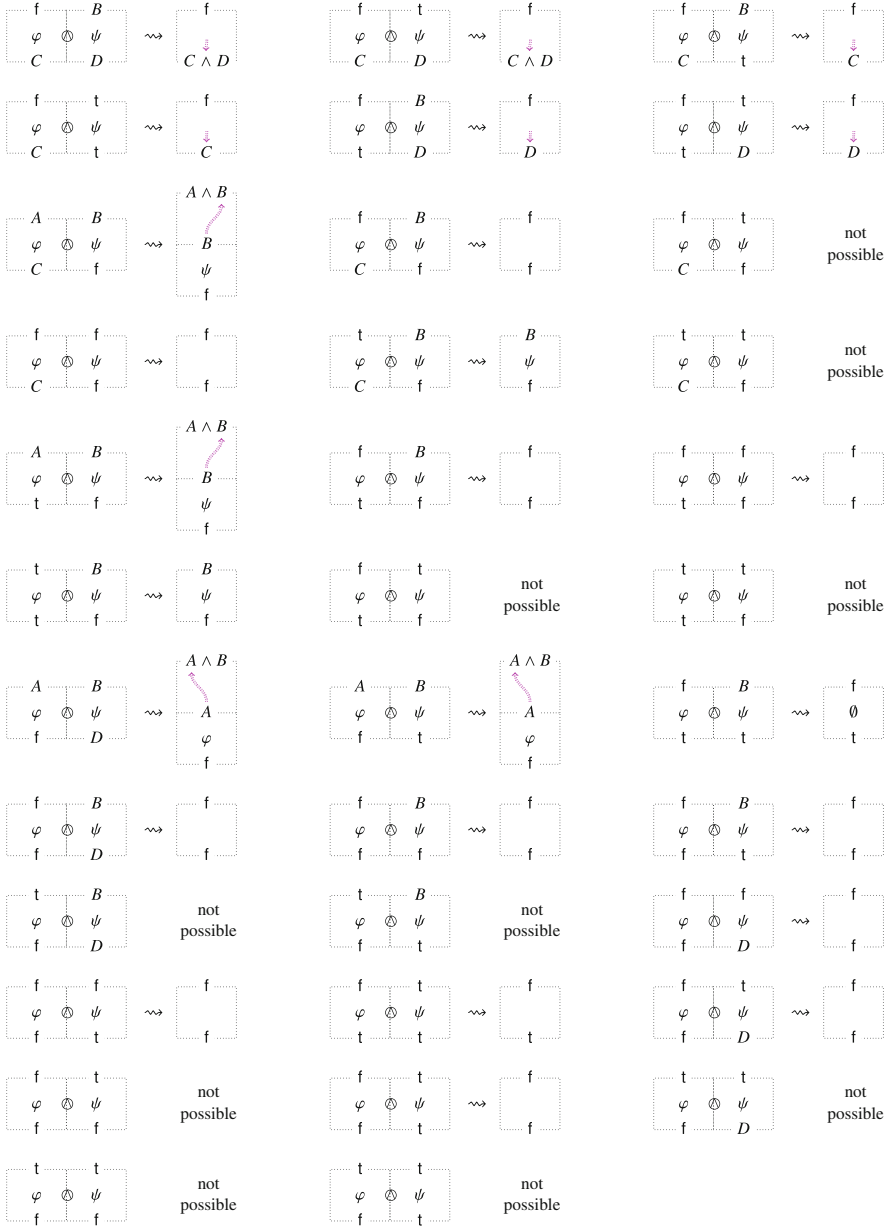


Fig. 5. Purification cases for conjunction. The cases for disjunction are dual.

These cuts can be simulated by an m-flow in combinatorial flows. More precisely, assume we have a combinatorial proof with cuts for a sequent $\Gamma = \overline{A}, B$, as shown on the left in Fig. 6. We have that $H = H_A \vee H_B$ and $\psi = \psi_A \odot \psi_{BC}$ because of Lemma 6.3.

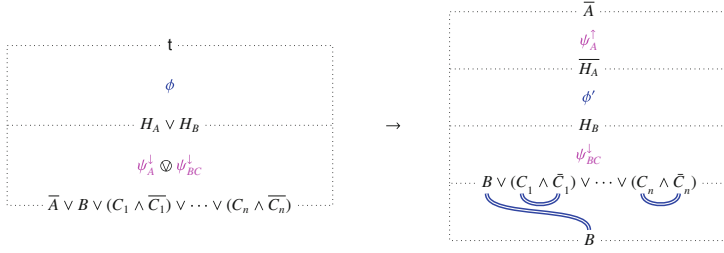


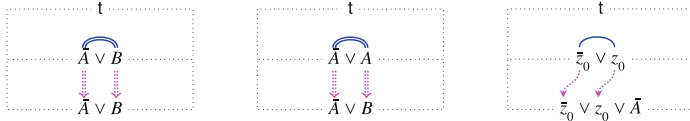
Fig. 6. Translating combinatorial proofs with cuts into combinatorial flows.

We can translate this into a combinatorial flow $\phi: A \vdash B$, as shown on the right in Fig. 6. There, the m-flow ϕ' exists by Proposition 5.3.

We are now going to show the converse, i.e., we will give a polynomial translation from combinatorial flows to combinatorial proofs with cuts. For this, we have to be a bit more careful with the units, as combinatorial proofs with cuts are only defined for unit-free formulas. We let z_0 be a fresh propositional variable, and we define the function $(\cdot)^0$ on pure formulas as follows: If $A \equiv B$ for some unit-free formula B , then $A^0 = B$. If $A \equiv t$, then $A^0 = \bar{z}_0 \vee z_0$. If $A \equiv f$, then $A^0 = \bar{z}_0 \wedge z_0$.

Now assume we have a combinatorial flow $\phi: A \vdash B$. We can translate this inductively into a combinatorial proof with cuts of the sequent $\Gamma = \bar{A}^0, B^0$.

1. First, every m-flow, a[↓]-flow, and a[↑]-flow can be immediately translated into a combinatorial proof. For the cases $\langle A, B, \mathbb{B}_\phi \rangle$ and $\langle A, B, f_\psi^\downarrow \rangle$ and $\langle A, t, f_\psi^\uparrow \rangle$ these are shown below. The others are similar.



2. If $\phi = \phi_1 \otimes \phi_2$ with $\phi_1: A_1 \vdash B_1$ and $\phi_2: A_2 \vdash B_2$ then we have by induction hypothesis combinatorial proofs with cuts of $\Gamma_1 = \bar{A}_1^0, B_1^0$ and $\Gamma_2 = \bar{A}_2^0, B_2^0$. By the construction in [19], we get one with conclusion $\Gamma = \bar{A}_1^0, \bar{A}_2^0, B_1^0 \wedge B_2^0$ which is equivalent to $(\bar{A}_1 \vee \bar{A}_2)^0, (B_1 \wedge B_2)^0$. The case for $\phi = \phi_1 \odot \phi_2$ is similar.
3. If $\phi = \phi_1 \odot \phi_2$ with $\phi_1: A \vdash D_1$ and $\phi_2: D_2 \vdash B$ with $D_1 \equiv D_2$, we proceed similarly, with the difference that we add a new cut formula D_1^0 . Note that $\mathcal{G}(D_1^0) = \mathcal{G}(D_2^0)$ and $\bar{D}_1^0 \wedge D_1^0$ is added to the conclusion sequent.

10 Conclusion and Future Work

We have defined combinatorial flows as a graphical representation of classical proofs, merging features from atomic flows and combinatorial proofs. Unlike atomic flows, they allow to reconstruct a proof, and unlike combinatorial proofs they allow a more flexible mixture of additive and multiplicative parts of a proof.

In future work, we would like to use combinatorial flows to define more flexible normalization procedures. Observe that atomic flows allow local reductions [11] that have to be preceded by a global cycle removing step [13,32], and that combinatorial proofs perform global cut reductions [16,19] similar to the sequent calculus. We hope that with combinatorial flows we can merge the advantages of both, via a “semi-local” normalization procedure that works on the level of flowboxes, similar to the purification procedure that we presented in Sect. 8. In fact, note that this purification can be seen as a normalization for weakening, and what remains to be done is a similar procedure for contraction.

References

1. Blute, R., Cockett, R., Seely, R., Trimble, T.: Natural deduction and coherence for weakly distributive categories. *J. Pure Appl. Algebra* **113**, 229–296 (1996)
2. Brünnler, K., Tiu, A.F.: A local system for classical logic. In: Nieuwenhuis, R., Voronkov, A. (eds.) *LPAR 2001. LNCS (LNAI)*, vol. 2250, pp. 347–361. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45653-8_24
3. Buss, S.R.: The undecidability of k -provability. *Ann. Pure Appl. Log.* **53**(1), 72–102 (1991)
4. Carbone, A.: Interpolants, cut elimination and flow graphs for the propositional calculus. *Ann. Pure Appl. Log.* **83**, 249–299 (1997)
5. Carbone, A.: Turning cycles into spirals. *Ann. Pure Appl. Log.* **96**(1–3), 57–73 (1999)
6. Cook, S.A., Reckhow, R.A.: The relative efficiency of propositional proof systems. *J. Symb. Log.* **44**(1), 36–50 (1979)
7. Danos, V., Regnier, L.: The structure of multiplicatives. *Arch. Math. Log.* **28**(3), 181–203 (1989)
8. Das, A.: Rewriting with linear inferences in propositional logic. In: van Raamsdonk, F. (ed.) *24th International Conference on Rewriting Techniques and Applications (RTA)*. *Leibniz International Proceedings in Informatics (LIPIcs)*, vol. 21, pp. 158–173. Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2013)
9. Girard, J.-Y.: Linear logic. *Theor. Comput. Sci.* **50**, 1–102 (1987)
10. Girard, J.-Y., Lafont, Y., Taylor, P.: *Proofs and Types*. *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press (1989)
11. Guglielmi, A., Gundersen, T.: Normalisation control in deep inference via atomic flows. *Logical Methods Comput. Sci.* **4**(1:9), 1–36 (2008)
12. Guglielmi, A., Gundersen, T., Parigot, M.: A proof calculus which reduces syntactic bureaucracy. In: Lynch, C. (ed.) *21st International Conference on Rewriting Techniques and Applications, RTA 2010*. *Leibniz International Proceedings in Informatics (LIPIcs)*, UK, vol. 6, pp. 135–150. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik (2010)
13. Guglielmi, A., Gundersen, T., Straßburger, L.: Breaking paths in atomic flows for classical logic. In: *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010*, Edinburgh, UK, 11–14 July 2010, pp. 284–293. IEEE Computer Society (2010)
14. Guglielmi, A., Straßburger, L.: Non-commutativity and MELL in the calculus of structures. In: Fribourg, L. (ed.) *CSL 2001. LNCS*, vol. 2142, pp. 54–68. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44802-0_5
15. Heijltjes, W., Houston, R.: Proof equivalence in MLL is PSPACE-complete. *Log. Methods Comput. Sci.* **12**(1) (2016)
16. Heijltjes, W., Hughes, D., Straßburger, L.: Proof nets for first-order additive linear logic. *Research Report RR-9201*, Inria (2018)

17. Hughes, D.: Simple multiplicative proof nets with units (2005, preprint)
18. Hughes, D.: Proofs without syntax. *Ann. Math.* **164**(3), 1065–1076 (2006)
19. Hughes, D.: Towards Hilbert’s 24th problem: combinatorial proof invariants: (preliminary version). *Electr. Notes Theor. Comput. Sci.* **165**, 37–63 (2006)
20. Kelly, G.M., Lane, S.M.: Coherence in closed categories. *J. Pure Appl. Algebra* **1**, 97–140 (1971)
21. Kleene, S.C.: Permutability of inferences in Gentzen’s calculi LK and LJ, memoirs of the American mathematical society. *J. Symb. Log.* **19**(1), 62–63 (1954)
22. Lamarche, F., Straßburger, L.: Naming proofs in classical propositional logic. In: Urzyczyn, P. (ed.) *TLCA 2005. LNCS*, vol. 3461, pp. 246–261. Springer, Heidelberg (2005). <https://doi.org/10.1007/11417170.19>
23. Lamarche, F., Straßburger, L.: From proof nets to the free *-autonomous category. *Logical Methods Comput. Sci.* **2**(4:3), 1–44 (2006)
24. Prawitz, D.: *Natural Deduction: A Proof-Theoretical Study*. Almqvist and Wiksell (1965)
25. Retoré, C.: Pomset logic as a calculus of directed cographs. In: Abrusci, V.M., Casadio, C. (eds.) *Dynamic Perspectives in Logic and Linguistics*, Bulzoni, Roma, pp. 221–247 (1999). Also available as INRIA Rapport de Recherche RR-3714
26. Retoré, C.: Handsome proof-nets: perfect matchings and cographs. *Theor. Comput. Sci.* **294**(3), 473–488 (2003)
27. Selinger, P.: A survey of graphical languages for monoidal categories. In: Coecke, B. (ed.) *New Structures for Physics*, pp. 289–355. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-12821-9_4
28. Straßburger, L.: A characterization of medial as rewriting rule. In: Baader, F. (ed.) *RTA 2007. LNCS*, vol. 4533, pp. 344–358. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-73449-9_26
29. Straßburger, L.: Combinatorial flows and their normalisation. In: Miller, D. (ed.) *FSCD 2017. Leibniz International Proceedings in Informatics (LIPIcs)*, vol. 84, pp. 31:1–31:17 (2017)
30. Straßburger, L.: Deep inference and expansion trees for second-order multiplicative linear logic. *Math. Struct. Comput. Sci.* **29**(8), 1030–1060 (2019)
31. Thiele, R.: Hilbert’s twenty-fourth problem. *Amer. Math. Monthly* **110**, 1–24 (2003)
32. Tubella, A.A., Guglielmi, A., Ralph, B.: Removing cycles from proofs. In: Goranko, V., Dam, M. (eds.) *26th EACSL Annual Conference on Computer Science Logic, CSL 2017, Stockholm, Sweden, 20–24 August 2017. LIPIcs*, vol. 82, pp. 9:1–9:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2017)



A Logic of “Black Box” Classifier Systems

Xinghan Liu¹(✉) and Emiliano Lorini²

¹ IRIT, University of Toulouse, Toulouse, France
xinghan.liu@univ.toulouse.fr

² IRIT, CNRS, University of Toulouse, Toulouse, France

Abstract. Binary classifiers are traditionally studied by propositional logic (PL). PL can only represent them as white boxes, under the assumption that the underlying Boolean function is fully known. Binary classifiers used in practical applications and trained by machine learning are however opaque. They are usually described as black boxes. In this paper, we provide a product modal logic called PLC (Product modal Logic for binary input Classifier) in which the notion of “black box” is interpreted as the uncertainty over a set of classifiers. We give results about axiomatics and complexity of satisfiability checking for our logic. Moreover, we present a dynamic extension in which the process of acquiring new information about the actual classifier can be represented.

1 Introduction

The notions of explanation and explainability have been extensively investigated by philosophers [11, 14] and are key aspects of AI-based systems given the importance of explaining the behavior and prediction of an artificial intelligent system. A variety of notions of explanation for classifier systems have been discussed in the area of explainable AI (XAI). Since systems trained by machine learning are increasingly opaque, instead of studying specific models, the *model-agnostic* approach comes into focus. Namely, given a black box system or algorithm, we know nothing about how it works inside. Without opening the black box, we can query some (but not all) inputs and have some partial information about the system. Initially there were *global* model-agnostic explanations like partial dependence plots and global surrogate models. Recently LIME [21] and its followers e.g. SHAP [16] and Anchors [22] have raised a *local* model-agnostic explanation approach, namely explaining why a given input is classified in a certain way. For a comprehensive overview of the research in this area see, e.g., [18].

At the mathematical level, a binary classifier can be viewed as a Boolean function and is traditionally studied by propositional logic. Recent years have witnessed several logic-based approaches to local explanation of classifier systems [2, 6, 12, 13, 23], e.g., computing prime implicants and abductive explanations of a given classification, and detecting biases in the classification process by means of the notion counterfactual explanation. But, all these logic-based approaches deal with “white box” classifiers, i.e., specific transparent models representable by propositional formulas. A limitation is that given a Boolean function f and a

propositional formula φ , φ either fully expresses f or does not express f at all. This all-or-nothing nature makes it impossible to give a *partial* description of f , which is a natural way to represent a black box classifier.

The central idea of this paper is that a product modal logic is the proper way to represent a “black box” classifier. As we have shown in [15], it is natural to think of a classifier with binary inputs as a partition of an S5 Kripke model, where each possible state stands for an input instance. However, this only represents “white box” classifiers. We extend this semantics with a second dimension universally ranging over a set of possible classifiers, which results in a proper extension of the product modal logic $S5 \times S5 = S5^2$ [8] we call PLC (Product modal Logic of binary-input Classifiers). The notion of black box is interpreted as an agent’s uncertainty among those (white box) classifiers, as illustrated in Fig. 1.

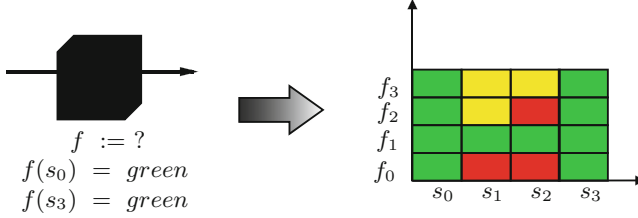


Fig. 1. A classifier associating color labels in $\{\text{red, yellow, green}\}$ to input instances. We do not know its Boolean formula, since f_0, f_1, f_2, f_3 are all compatible with our partial knowledge of it. However, we know that the two input instances s_0 and s_3 are both classified as green. (Color figure online)

The paper is structured as follows. Section 2 introduces the modal language and semantic model of PLC which we name multi-classifier model (MCM). Its axiomatics along with the completeness and complexity results for the satisfiability checking problem are given in Sect. 3. In Sect. 4, we will exemplify the logic’s application by using it to represent the notion of black box and to formalize different notions of classifier explanation. A dynamic extension is given in Sect. 6 to capture the process of acquisition of new knowledge about the classifier. Some non-routine proofs are given in a technical annex at the end of the paper.

2 Language and Semantics

Let $Atm_0 = \{p, q, \dots\}$ be a countable set of atomic propositions which intend to denote input variables (features) of a classifier. We introduce a finite set Val to denote the possible output values (classifications, decisions) of the classifier.

Elements of Val are noted $x, y, \dots$. For any $x \in Val$, we call $t(x)$ a decision atom, and have $Dec = \{t(x) : x \in Val\}$ ¹. Finally let $Atm = Atm_0 \cup Dec$.

The modal language $\mathcal{L}$ is defined by the following grammar:

$$\varphi ::= p \mid t(x) \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \Box_I \varphi \mid \Box_F \varphi,$$

where p ranges over Atm_0 and x ranges over Val .

Definition 1. A multi-classifier model (MCM) is a pair $\Gamma = (S, \Phi)$ where $S \subseteq 2^{Atm_0}$ and $\Phi \subseteq F_S$, with $F_S = Val^S$ the set of functions with domain S and codomain Val . A pointed MCM is a triple (Γ, s, f) where $\Gamma = (S, \Phi)$ is an MCM, $s \in S$ and $f \in \Phi$. The class of all multi-classifier models is noted **MCM**.

Formulas in $\mathcal{L}$ are interpreted relative to a pointed MCM as follows.

Definition 2 (Satisfaction relation). Let $\Gamma = (S, \Phi)$ be an MCM, $s \in S$ and $f \in \Phi$. Then,

$$\begin{aligned} (\Gamma, s, f) \models p &\iff p \in s, \\ (\Gamma, s, f) \models t(x) &\iff f(s) = x, \\ (\Gamma, s, f) \models \neg\varphi &\iff (\Gamma, s, f) \not\models \varphi, \\ (\Gamma, s, f) \models \varphi \wedge \psi &\iff (\Gamma, s, f) \models \varphi \text{ and } (\Gamma, s, f) \models \psi, \\ (\Gamma, s, f) \models \Box_I \varphi &\iff \forall s' \in S : (\Gamma, s', f) \models \varphi, \\ (\Gamma, s, f) \models \Box_F \varphi &\iff \forall f' \in \Phi : (\Gamma, s, f') \models \varphi. \end{aligned}$$

Both $\Box_I \varphi$ and $\Box_F \varphi$ have standard modal reading but range over different sets. $\Box_I \varphi$ has to be read “ φ necessarily holds for the actual function, regardless of the input instance”, while its dual $\Diamond_I \varphi =_{def} \neg \Box_I \neg \varphi$ has to be read “ φ possibly holds for the actual function, regardless of the input instance”. Similarly, $\Box_F \varphi$ has to be read “ φ necessarily holds for the actual input instance, regardless of the function” and its dual $\Diamond_F \varphi$ has to be read “ φ possibly holds for the actual input instance, regardless of the function”.

Let X be a finite subset of Atm_0 . An important abbreviation is the following:

$$[X]\varphi =_{def} \bigwedge_{Y \subseteq X} ((\bigwedge_{p \in Y} \wedge \bigwedge_{p \in X \setminus Y} \neg p) \rightarrow \Box_I ((\bigwedge_{p \in Y} \wedge \bigwedge_{p \in X \setminus Y} \neg p) \rightarrow \varphi)).$$

Complex as it seems, $[X]\varphi$ means nothing but “ φ necessarily holds, regardless of the values of the input variables outside X ” or “ φ necessarily holds, if the values of the input variables in X are kept fixed”. It can be justified by checking that $(\Gamma, s, f) \models [X]\varphi$, if and only if $\forall s' \in S$, if $s \cap X = s' \cap X$ then $(\Gamma, s', f) \models \varphi$. Its dual $\langle X \rangle \varphi =_{def} \neg [X] \neg \varphi$ has to be read “ φ possibly holds, if the values of the input variables in X are kept fixed”. These modalities have a *ceteris paribus* reading and were first introduced in [9]. Similar modalities are used in existing logics of functional dependence [3, 25].

¹ Notice that p denotes an input *variable*, while x is an output *value* rather than the output *variable*, which makes sense of the symbolic difference between p and $t(x)$.

A formula φ of $\mathcal{L}$ is said to be satisfiable relative to the class **MCM** if there exists a pointed multi-classifier model (Γ, s, f) with $\Gamma \in \mathbf{MCM}$ such that $(\Gamma, s, f) \models \varphi$. We say that φ is valid in the multi-classifier model $\Gamma = (S, \Phi)$, noted $\Gamma \models \varphi$, if $(\Gamma, s, f) \models \varphi$ for every $s \in S, f \in \Phi$. It is said to be valid relative to **MCM**, noted $\models_{\mathbf{MCM}} \varphi$, if $\neg\varphi$ is not satisfiable relative to **MCM**.

3 Axiomatics and Complexity

In this section, we are going to present two axiomatics for the language $\mathcal{L}$ by distinguishing the finite-variable from the infinite-variable case. We will moreover give complexity results for satisfiability checking. Before, we are going to introduce an alternative Kripke semantics for the interpretation of the language $\mathcal{L}$. It will allow us to use the standard canonical model technique for proving completeness. Indeed, this technique cannot be directly applied to MCMs in the infinite-variable case.

3.1 Alternative Kripke Semantics

The crucial concept of the alternative semantics is multi-decision model (MDM).

Definition 3. An MDM is a tuple $M = (W, \sim_{\square_I}, \sim_{\square_F}, V)$ where:

- W is a set of worlds,
- $\sim_{\square_I}$ and $\sim_{\square_F}$ are equivalence relations on W ,
- $V : W \longrightarrow 2^{Atm}$ is a valuation function,

and which satisfies the following constraints, $\forall w, v \in W, \forall x, y \in Val$:

- (C1) $\sim_{\square_I} \circ \sim_{\square_F} = \sim_{\square_F} \circ \sim_{\square_I}$,
- (C2) if $V_{Atm_0}(w) = V_{Atm_0}(v)$ then $V_{Dec}(w) = V_{Dec}(v)$,
- (C3) if $w \sim_{\square_F} v$ then $V_{Atm_0}(w) = V_{Atm_0}(v)$,
- (C4) if $\mathbf{t}(x) \in V(w)$ and $x \neq y$ then $\mathbf{t}(y) \notin V(w)$,
- (C5) $\exists x \in Val$ such that $\mathbf{t}(x) \in V(w)$,

with $V_Y(w) = (V(w) \cap Y)$ for every $w \in W$ and for every $Y \subseteq Atm$, and $\circ$ the standard composition operator for binary relations.

The class of multi-decision models is noted **MDM**. An MDM $M = (W, \sim_{\square_I}, \sim_{\square_F}, V)$ is called finite if W is finite. The class of finite MDMs is noted finite-**MDM**. Interpretation of formulas in $\mathcal{L}$ relative to a pointed MDM goes as follows. (We omit interpretations for $\neg$ and $\wedge$ which are defined as usual.)

Definition 4 (Satisfaction Relation). Let $M = (W, \sim_{\square_I}, \sim_{\square_F}, V)$ be an MDM and let $w \in W$. Then,

$$\begin{aligned} (M, w) \models q &\iff q \in V(w) \text{ for } q \in Atm, \\ (M, w) \models \square_I \varphi &\iff \forall v \in W, \text{ if } w \sim_{\square_I} v \text{ then } v \models \varphi, \\ (M, w) \models \square_F \varphi &\iff \forall v \in W, \text{ if } w \sim_{\square_F} v \text{ then } v \models \varphi. \end{aligned}$$

Validity and satisfiability of formulas in $\mathcal{L}$ relative to class **MDM** (resp. **finite-MDM**) are defined in the usual way.

The most important result in this subsection is the semantic equivalence between **MCM** and **MDM**, regardless of Atm_0 being finite or infinite. Although a pointed MDM (M, w) looks like a pointed MCM (Γ, s, f) , it only approximates it. Indeed, unlike an MCM, an MDM M may be redundant, that is, (i) a classifier in M (i.e., a $\sim_{\square_I}$ -equivalence class) may include multiple copies of the same input instance (i.e., of the same valuation for the atoms in Atm_0), or (ii) M may contain multiple copies of the same classifier (i.e., two identical $\sim_{\square_F}$ -equivalence classes). Moreover, an MDM M may be “defective” insofar as (iii) the intersection between a classifier in M (i.e., a $\sim_{\square_I}$ -equivalence class) and the set of all possible classifications of a given input instance by the classifiers in M (i.e., a $\sim_{\square_F}$ -equivalence class) is not a singleton. What makes the proof of the following theorem non-trivial is transforming a possibly redundant or defective MDM into a non-redundant and non-defective one by preserving truth of formulas. A non-redundant and non-defective MDM is then isomorphic to an MCM.

Theorem 1. *Let $\varphi \in \mathcal{L}$. Then, φ is satisfiable relative to the class **MCM** if and only if it is satisfiable relative to the class **MDM**.*

Proof. We start with the left-to-right direction of the proof. Let (Γ, s_0, f_0) be a pointed MCM with $\Gamma = (S, \Phi)$, $S \subseteq 2^{Atm_0}$ and $\Phi \subseteq F_S$ such that $(\Gamma, s_0, f_0) \models \varphi$. We define the tuple $M = (W, \sim_{\square_I}, \sim_{\square_F}, V)$ as follows:

- $W = \{(s, f) : s \in S \text{ and } f \in \Phi\}$,
- $\forall (s, f), (s', f') \in W, (s, f) \sim_{\square_I} (s', f') \text{ iff } f = f'$
- $\forall (s, f), (s', f') \in W, (s, f) \sim_{\square_F} (s', f') \text{ iff } s = s'$,
- $\forall (s, f) \in W, V(s, f) = s \cup \{t(f(s))\}$.

It is routine exercise to verify that M so defined is an MDM. Moreover, by induction on the structure of φ , it is easy to prove that “ $(\Gamma, s, f) \models \varphi$ iff $(M, (s, f)) \models \varphi$ ” for every $s \in S$ and $f \in \Phi$. Thus, $(M, (s_0, f_0)) \models \varphi$ since $(\Gamma, s_0, f_0) \models \varphi$.

Let us now prove the right-to-left direction. Let $M = (W, \sim_{\square_I}, \sim_{\square_F}, V)$ be an MDM and $w_0 \in W$ such that $(M, w_0) \models \varphi$. Given $v \in W$, let $|v|_{Atm} = \{u \in W : v \sim_{\square_I} u \text{ and } V_{Atm}(v) = V_{Atm}(u)\}$. We transform the MDM M into a tuple $M' = (W', \sim'_{\square_I}, \sim'_{\square_F}, V')$ such that:

- $W' = \{|v|_{Atm} : v \in W\}$,
- $\forall |v|_{Atm}, |u|_{Atm} \in W', |v|_{Atm} \sim'_{\square_I} |u|_{Atm} \text{ iff } \exists v \in |v|_{Atm}, u \in |u|_{Atm} \text{ such that } v \sim_{\square_I} u$,
- $\forall |v|_{Atm}, |u|_{Atm} \in W', |v|_{Atm} \sim'_{\square_F} |u|_{Atm} \text{ iff } \exists v \in |v|_{Atm}, u \in |u|_{Atm} \text{ such that } v \sim_{\square_F} u$,
- $\forall |v|_{Atm} \in W', V'(|v|_{Atm}) = V(v)$.

It is routine exercise to verify that M' is an MDM and, by induction on the structure of φ , that “ $(M, v) \models \varphi$ iff $(M', |v|_{Atm}) \models \varphi$ ” for every $v \in W$. Thus,

$(M, |w_0|_{Atm}) \models \varphi$ since $(M, w_0) \models \varphi$. Finally, because of Constraints **C2** and **C3** in Definition 3, the following property holds:

$$(C6) \quad (\sim'_{\square_I} \cap \sim'_{\square_F}) = id_{W'},$$

where $id_{W'}$ is the identity relation on W' .

Let $W'/\sim'_{\square_I}$ be the quotient set of W' by the equivalence relation $\sim'_{\square_I}$. We note $\tau, \tau', \dots$ its elements. Given $\tau, \tau' \in W'/\sim'_{\square_I}$, we write $\tau \approx_F \tau'$ if and only if, $\forall |v|_{Atm} \in \tau, \forall |u|_{Atm} \in \tau', \text{ if } V'_{Atm_0}(|v|_{Atm}) = V'_{Atm_0}(|u|_{Atm}) \text{ then } V'_{Dec}(|v|_{Atm}) = V'_{Dec}(|u|_{Atm})$. Given $|v|_{Atm}, |u|_{Atm} \in W'$, we write $|v|_{Atm} \simeq |u|_{Atm}$ if and only if $\exists \tau, \tau' \in W'/\sim'_{\square_I}$ such that $|v|_{Atm} \in \tau, |u|_{Atm} \in \tau', \tau \approx_F \tau'$ and $V'_{Atm_0}(|v|_{Atm}) = V'_{Atm_0}(|u|_{Atm})$. Clearly, $\approx_F$ and $\simeq$ are equivalence relations.

We are going transform the MDM M' into an MDM which does not contain multiple copies of the same function and which satisfies the same formulas as M' . We define it to be a tuple $M'' = (W'', \sim''_{\square_I}, \sim''_{\square_F}, V'')$ such that:

- $W'' = \{ \simeq(|v|_{Atm}) : |v|_{Atm} \in W' \},$
- $\forall \simeq(|v|_{Atm}), \simeq(|u|_{Atm}) \in W'', \simeq(|v|_{Atm}) \sim''_{\square_I} \simeq(|u|_{Atm}) \text{ iff } \exists |v|_{Atm} \in \simeq(|v|_{Atm}), |u|_{Atm} \in \simeq(|u|_{Atm}) \text{ such that } |v|_{Atm} \sim'_{\square_I} |u|_{Atm},$
- $\forall \simeq(|v|_{Atm}), \simeq(|u|_{Atm}) \in W'', \simeq(|v|_{Atm}) \sim''_{\square_F} \simeq(|u|_{Atm}) \text{ iff } \exists |v|_{Atm} \in \simeq(|v|_{Atm}), |u|_{Atm} \in \simeq(|u|_{Atm}) \text{ such that } |v|_{Atm} \sim'_{\square_F} |u|_{Atm},$
- $\forall \simeq(|v|_{Atm}) \in W'', V''(\simeq(|v|_{Atm})) = V'(|v|_{Atm}).$

Again, it is routine to verify that M'' is an MDM which satisfies the previous Constraint **C6**. Moreover, by induction on the structure of φ , it is easy to prove that “ $(M', |v|_{Atm}) \models \varphi$ iff $(M'', \simeq(|v|_{Atm})) \models \varphi$ ” for every $|v|_{Atm} \in W'$. Thus, $(M'', \simeq(|w_0|_{Atm})) \models \varphi$ since $(M', |w_0|_{Atm}) \models \varphi$.

We can easily build an MCM isomorphic to M'' . □

3.2 Finite-Variable Case

We first consider the variant of the logic with finitely many propositional atoms in Atm_0 . For every finite $X, Y \subseteq Atm_0$ we define:

$$cn_{X,Y} =_{def} \bigwedge_{p \in X} p \wedge \bigwedge_{p \in (Y \setminus X)} \neg p.$$

Definition 5 (Logic PLC). Let Atm_0 be finite. We define PLC as the extension of classical propositional logic given by axioms and rules of inference in Table 1.

Axioms **AtLeast**_{t(x)}, **AtMost**_{t(x)} and **Funct** guarantee that every input $Y \subseteq Atm_0$, whose syntactic counterpart is cn_{Y, Atm_0} , has only one decision atom as output. Axioms **K**_■, **T**_■, **4**_■ and **5**_■ together with the rule of inference **Nec**_■ indicate that both modal operators $\square_F$ and $\square_I$ satisfy the principles of the modal logic S5. According to Axiom **Comm**, they moreover commute. This makes the logic meet the requirement of a product of two S5 modal logics, i.e., S5² [8]. Nevertheless, the existence of the two “independence” Axioms **Indep**_{□_F, p} and **Indep**_{□_F, ¬p} indicates that PLC is stronger than S5² in general.

Table 1. Axioms and rules of inference, with $\blacksquare \in \{\Box_I, \Box_F\}$

$(\blacksquare\varphi \wedge \blacksquare(\varphi \rightarrow \psi)) \rightarrow \blacksquare\psi$	(K_■)
$\blacksquare\varphi \rightarrow \varphi$	(T_■)
$\blacksquare\varphi \rightarrow \blacksquare\blacksquare\varphi$	(4_■)
$\neg\blacksquare\varphi \rightarrow \blacksquare\neg\blacksquare\varphi$	(5_■)
$\Box_F\Box_I\varphi \leftrightarrow \Box_I\Box_F\varphi$	(Comm)
$\bigvee_{x \in Val} t(x)$	(AtLeast_{t(x)})
$t(x) \rightarrow \neg t(y)$ if $x \neq y$	(AtMost_{t(x)})
$(\text{cn}_{X, \text{Atm}_0} \wedge t(x)) \rightarrow \Box_I(\text{cn}_{X, \text{Atm}_0} \rightarrow t(x))$	(Func_t)
$p \rightarrow \Box_F p$	(Indep_{□_F, p})
$\neg p \rightarrow \Box_F \neg p$	(Indep_{□_F, ¬p})
$\frac{\varphi}{\blacksquare\varphi}$	(Nec_■)

Soundness of PLC relative to **MCM** is a simple exercise. To prove the completeness result, we first need to show that PLC is complete relative to **MDM**, which is proven by the canonical model construction.

Theorem 2. *Let Atm_0 be finite. Then, the logic PLC is sound and complete relative to the class **MDM**.*

Our main result of this subsection becomes a corollary of Theorems 1 and 2.

Corollary 1. *Let Atm_0 be finite. Then, the logic PLC is sound and complete relative to the class **MCM**.*

3.3 Infinite-Variable Case

We now move to the infinite-variable variant of our logic, under the assumption that the set Atm_0 is countably infinite. In order to obtain an axiomatics we just need to drop the “functionality” Axiom **Func_t** of Table 1. Indeed, when Atm_0 is infinite, the construction $\text{cn}_{X, \text{Atm}_0}$ cannot be expressed in a finitary way.

Definition 6 (Logic WPLC). *We define WPLC (Weak PLC) to be the extension of classical propositional logic given by Axioms **K_■**, **T_■**, **4_■**, **5_■**, **Comm**, **AtLeast_{t(x)}**, **AtMost_{t(x)}**, **Indep_{□_F, p}** and **Indep_{□_F, ¬p}**, and the rule of inference **Nec_■** in Table 1.*

Soundness of the logic WPLC is a straightforward exercise. For completeness, we need to distinguish MDMs from quasi-MDMs that are obtained by removing the “functionality” Constraint **C2** from Definition 3.

Definition 7 (Quasi-MDM). *A quasi-MDM is a tuple $M = (W, \sim_{\Box_I}, \sim_{\Box_F}, V)$ where W , $\sim_{\Box_I}$, $\sim_{\Box_F}$ and V are defined as in Definition 3 and which satisfies all constraints of Definition 3 except **C2**.*

The class of quasi-MDMs is noted **QMDM**. A quasi-MDM $M = (W, \sim_{\square_I}, \sim_{\square_F}, V)$ is said to be finite if W is finite. The class of finite quasi-MDMs is noted finite-**QMDM**. Semantic interpretation of formulas in $\mathcal{L}$ relative to quasi-MDMs is analogous to semantic interpretation relative to MDMs given in Definition 4. Moreover, validity and satisfiability of formulas in $\mathcal{L}$ relative to class **QMDM** (resp. finite-**QMDM**) is again defined in the usual way.

The first crucial result of this subsection is that when Atm_0 is infinite the language $\mathcal{L}$ cannot distinguish finite MDMs from finite quasi-MDMs.

Theorem 3. *Let $\varphi \in \mathcal{L}$ with Atm_0 infinite. Then, φ is satisfiable relative to the class finite-MDM if and only if it is satisfiable relative to the class finite-QMDM.*

Proof. The left-to-right direction is trivial. We prove the right-to-left direction. Suppose Atm_0 is infinite. Moreover, let $M = (W, \sim_{\square_I}, \sim_{\square_F}, V)$ be a finite quasi-MDM and $w_0 \in W$ such that $(M, w_0) \models \varphi$. Since Atm_0 is infinite and W is finite, we can define an injection $g : W \rightarrow Atm_0 \setminus Atm(\varphi)$. We define the tuple $M' = (W', \sim'_{\square_I}, \sim'_{\square_F}, V')$ as follows:

- $W' = W$;
- $\sim'_{\square_I} = \sim_{\square_I}$;
- $\sim'_{\square_F} = \sim_{\square_F}$;
- for every $w \in W'$,

$$V'(w) = (V(w) \setminus \{g(v) : v \in W \text{ and } w \neq v\}) \cup \{g(w)\}.$$

It is routine to verify that M' is a finite MDM. Indeed, $V'_{Atm_0}(w) \neq V'_{Atm_0}(v)$ for all $w, v \in W'$ such that $w \neq v$. This guarantees that M' satisfies the “functionality” constraint **C2**. Moreover, by induction on the structure of φ , it is straightforward to prove that “ $(M, v) \models \varphi$ iff $(M', v) \models \varphi$ ” for every $v \in W$. The crucial point of the proof is that $\sim'_{\square_I} = \sim_{\square_I}$ and $\sim'_{\square_F} = \sim_{\square_F}$. Thus, $(M', w_0) \models \varphi$ since $(M, w_0) \models \varphi$. $\square$

The second result is that satisfiability for formulas in $\mathcal{L}$ relative to the class **QMDM** is equivalent to satisfiability relative to the class finite-**QMDM**.

Theorem 4. *Let $\varphi \in \mathcal{L}$. Then, φ is satisfiable relative to the class **QMDM** if and only if it is satisfiable relative to the class finite-**QMDM**.*

Proof. The right-to-left direction is clear. We are going to prove the left-to-right direction by a filtration argument.

Let $M = (W, \sim_{\square_I}, \sim_{\square_F}, V)$ be a quasi-MDM and $w_0 \in W$ such that $(M, w_0) \models \varphi$. It is routine to verify that $(\sim_{\square_I} \cup \sim_{\square_F})^* = \sim_{\square_I} \circ \sim_{\square_F} = \sim_{\square_F} \circ \sim_{\square_I}$. Thus, we can define $M' = (W', \sim'_{\square_I}, \sim'_{\square_F}, V')$ to be the submodel of M generated from w_0 through the relation $\sim_{\square_I} \circ \sim_{\square_F}$. M' is a quasi-MDM and $(M', w_0) \models \varphi$.

Let $sf(\varphi)$ be the set of all subformulas of φ and let $sf^+(\varphi) = sf(\varphi) \cup Dec$. Moreover, for every $v \in W'$, let $\Theta(v) = \{\psi \in sf^+(\varphi) : (M', v) \models \psi\}$. For every $v, u \in W'$, we define

$$v \simeq u \text{ iff } \Theta(v) = \Theta(u).$$

Moreover, we define $[v] = \{u \in W' : v \simeq u\}$.

We construct a new model $M'' = (W'', \sim''_{\Box_I}, \sim''_{\Box_F}, V'')$ where:

- $W'' = \{[v] : v \in W'\};$
- $[v] \sim''_{\Box_I} [u]$ iff

$$\forall \Box_I \psi \in sf(\varphi), ((M', v) \models \Box_I \psi \text{ iff } (M', u) \models \Box_I \psi);$$

- $[v] \sim''_{\Box_F} [u]$ iff

$$\forall \Box_F \psi \in sf(\varphi), ((M', v) \models \Box_F \psi \text{ iff } (M', u) \models \Box_F \psi) \text{ and}$$

$$\forall p \in sf(\varphi) \cap Atm_0, ((M', v) \models p \text{ iff } (M', u) \models p);$$

- $V''([v]) = V'_{sf(\varphi) \cap Atm_0}(v) \cup V'_{Dec}(v).$

M'' is indeed a filtration, for it satisfies that if $v \sim_{\blacksquare} u$, then $[v] \sim_{\blacksquare} [u]$; and if $\blacksquare \psi \in sf(\varphi)$ and $(M', v) \models \blacksquare \psi$, then $(M', u) \models \psi$, for $\blacksquare \in \{\Box_I, \Box_F\}$. Additionally, the valuation function is defined in the standard way.

To check that M'' is a finite quasi-MDM, we go through all constraints. For **C1** a crucial fact is that M' generated from w_0 through $\sim_{\Box_I} \circ \sim_{\Box_F}$, viz. $\forall v, u \in W', v \sim_{\Box_I} \circ \sim_{\Box_F} u$ and $v \sim_{\Box_F} \circ \sim_{\Box_I} u$. **C3** holds because of the definition of $\sim''_{\Box_F}$. **C4, C5** hold, since V'' not only considers $sf(\varphi) \cap Atm_0$ but also Dec .

It is routine to verify that $M'' = (W'', \sim''_{\Box_I}, \sim''_{\Box_F}, V'')$ is a filtration of M' and is a finite quasi-MDM. Therefore, $(M'', [w_0]) \models \varphi$. $\square$

The following theorem is provable by standard canonical model argument. Note that like Theorems 1 and 4, it does not rely on Atm_0 being infinite or finite.

Theorem 5. *The logic WPLC is sound and complete relative to the class QMDM.*

The fact that the logic WPLC is sound and complete relative to the class MCM is a direct corollary of Theorems 1, 3, 4 and 5.

Corollary 2. *Let Atm_0 be infinite. Then, the logic WPLC is sound and complete relative to the class MCM.*

3.4 Complexity Results

We now move to complexity of satisfiability checking. As for the axiomatics, we distinguish the finite-variable from the infinite-variable case. When Atm_0 is finite, the problem of verifying whether a formula is satisfiable is polynomial. The latter problem mirrors the satisfiability checking problem for the finite-variable modal logic S5 which is also known to be polynomial [10].

Theorem 6. *Let Atm_0 be finite. Then, checking satisfiability of $\mathcal{L}$ -formulas relative to the class MCM can be done in polynomial time.*

Proof. Suppose $|Atm_0|$ is finite. Then, the class **MCM** is bounded by some integer k . So, in order to determine whether a formula φ is satisfiable for the class **MCM**, it is sufficient to verify whether φ is satisfied in one of these MCMs. This verification takes a polynomial time in the size of φ since it is a repeated model checking and model checking in the product modal logic $S5^2$ is polynomial. $\square$

We know that when moving from the finite-variable to the infinite-variable case complexity of satisfiability checking is in NEXPTIME.

Theorem 7. *Let Atm_0 be infinite. Then, checking satisfiability of $\mathcal{L}$ -formulas relative to the class **MCM** is in NEXPTIME.*

Proof. We know that satisfiability checking for the product modal logic $S5^2$ with two $S5$ modalities $\Box_1$ and $\Box_2$ is NEXPTIME-complete [8]. We have a polynomial reduction of satisfiability checking for $\mathcal{L}$ -formulas relative to the class **MCM** to the latter problem. In particular, given a formula $\varphi \in \mathcal{L}$, we can translate it into a formula $tr(\varphi)$ of $S5^2$ where the translation tr is defined as follows: (i) $tr(q) = q$ for $q \in Atm$, (ii) $tr(\neg\varphi) = \neg tr(\varphi)$, (iii) $tr(\varphi_1 \wedge \varphi_2) = tr(\varphi_1) \wedge tr(\varphi_2)$, (iv) $tr(\Box_I \varphi) = \Box_I tr(\varphi)$, (v) $tr(\Box_F \varphi) = \Box_2 tr(\varphi)$. We have that φ is satisfiable for the class **MCM** if and only $\bigwedge_{\chi \in \Delta} \Box_1 \Box_2 \chi \wedge tr(\varphi)$ is a satisfiable formula of the product modal logic $S5^2$, where Δ is the following finite theory corresponding to the Axioms **Indep** $_{\Box_F, p}$, **Indep** $_{\Box_F, \neg p}$, **AtMost** $_{t(x)}$ and **AtLeast** $_{t(x)}$ of the logic WPLC:

$$\begin{aligned} \Delta = \{ & \bigvee_{x \in Val} t(x) \} \cup \{ t(x) \rightarrow \neg t(y) : x \neq y \} \cup \{ p \rightarrow \Box_2 p : p \in Atm_0(\varphi) \} \cup \\ & \{ \neg p \rightarrow \Box_2 \neg p : p \in Atm_0(\varphi) \}, \end{aligned}$$

and $Atm_0(\varphi)$ is the set of atoms in Atm_0 which occur in φ . $\square$

In [4] (see also [5]) it is proved that all proper normal extensions of the product modal logic $S5^2$ are in NP. In future work, we plan to verify whether these results are applicable to our setting in order to improve our complexity upper bound. The problem is that Axioms **Indep** $_{\Box_F, p}$, **Indep** $_{\Box_F, \neg p}$, **AtMost** $_{t(x)}$ and **AtLeast** $_{t(x)}$ are not axiom schemata in the proper sense.

4 Application

As mentioned, the $\Box_F$ operator is interpreted as partial knowledge about the classifier properties². In this section, we are going to exemplify how to use it for representing abductive explanations of a black box classifier.

² In the real world, partial knowledge may come from the data set as well as from the training process. For example, through learning, we may acquire knowledge that certain input features behave monotonically [26].

4.1 An Example of Classification Task

Consider a selection function which specifies whether a paper submitted to a conference is acceptable for presentation (1) or not (0) depending on its feature profile composed of four input features: significance (si), originality (or), clarity of the presentation (cl) and fulfillment of the anonymity requirement (an). For the sake of simplicity, we assume each feature in a paper profile is binary: si means the paper is significant while $\neg$ si means the paper is not significant, or means the paper is original while $\neg$ or means the paper is not original, and so on. We say that a first paper profile dominates a second paper profile, if all conditions satisfied by the second profile are satisfied by the first profile, and there exists a condition satisfied by the first profile which is not satisfied by the second profile. For example if the first profile is $si \wedge \neg or \wedge cl \wedge an$ and the second profile is $si \wedge \neg or \wedge \neg cl \wedge an$, then the first dominates the second.

The selection function is implemented in a classifier system that has to automatically split papers into two sets, the set of acceptable papers and the set of non-acceptable ones. We assume a certain agent (e.g., the author of a paper submitted to the conference) has only partial knowledge of the classifier system. In particular, she only knows that the classifier complies with the following three constraints: (1) submissions that satisfy the four conditions should be automatically accepted, (2) if a first paper profile dominates a second paper profile and the second paper profile is acceptable, then the first paper profile should also be acceptable, and (3) submissions that violate the anonymity requirement should be automatically rejected. In this case, the classifier is a black box for the agent.

Example 1. The multi-classifier model (MCM) representing the previous situation is the tuple $\Gamma = (S, \Phi)$ such that $S = 2^{\{si, or, cl, an\}}$ and

$$\begin{aligned} \forall f \in F_S, f \in \Phi \text{ iff } & (i) \forall s \in S, \text{ if } \{si, or, cl, an\} \subseteq s \text{ then } f(s) = 1, \\ & (ii) \forall s, s' \in S, \text{ if } s \subset s' \text{ and } f(s) = 1 \text{ then } f(s') = 1. \\ & (iii) \forall s \in S, \text{ if } an \notin s \text{ then } f(s) = 0. \end{aligned}$$

The agent does not know which function in Φ corresponds to the actual classifier, i.e., they are epistemically indistinguishable for her.

4.2 Explanations

Given space constraints, we exemplify explanations for white and black box classifiers by showing the dichotomy global vs. local explanation and the notion of *abductive explanation* based on *prime implicant*. Some notations and abbreviations are needed to formally represent them. Let λ denote a conjunction of literals, where a literal is an atom p or its negation $\neg p$. We write $\lambda \subseteq \lambda'$, call λ a part (subset) of λ' , if all literals in λ also occur in λ' ; and $\lambda \subset \lambda'$ if $\lambda \subseteq \lambda'$ but not $\lambda' \subseteq \lambda$. In the glossary of Boolean classifiers, s is called an *instance*, λ is called a *term* or *property* (of the instance). The set of terms is noted *Term*. Moreover, let $Atm(\varphi)$ denote the atoms occurring in φ . Finally, notice that the abbreviations $[X]\varphi$ and $\langle X \rangle \varphi$ introduced in Sect. 2 will be used.

Let us start with *prime implicant*, a key concept in the theory of Boolean functions since [20]. It can be presented in the language $\mathcal{L}(Atm)$ as follows:

$$\text{PImp}(\lambda, x) =_{\text{def}} \Box_I \left(\lambda \rightarrow \left(\mathbf{t}(x) \wedge \bigwedge_{p \in \text{Atm}(\lambda)} \langle \text{Atm}(\lambda) \setminus \{p\} \rangle \neg \mathbf{t}(x) \right) \right).$$

The abbreviation $\text{PImp}(\lambda, x)$ has to be read “ λ is a prime implicant for the classification x ”. Roughly speaking, the latter means that (i) λ necessarily leads to the classification x (why λ is an *implicant*), and (ii) for any of its proper subsets λ' , possibly there is a state where λ' holds but the classification is different from x (why λ is *prime*).

Prime implicant counts as a “global” explanation, in the sense that it is a property of the classifier and holds at *all* its input instances. Partially, as a response to the local approach in model-agnostic methods, researchers from logic-based approaches in XAI focus on the “localized” prime implicant namely *abductive explanation* (AXp)³. An abductive explanation is not only a prime implicant, but also a *property of the actual instance*. The notion of abductive explanation is expressed in $\mathcal{L}$ as follows:

$$\text{AXp}(\lambda, x) =_{\text{def}} \lambda \wedge \text{PImp}(\lambda, x).$$

$\text{AXp}(\lambda, x)$ just means that λ is an abductive explanation of the actual classification x . Let us instantiate the notions of prime implicant and abductive explanation in the paper example we introduced in Sect. 4.1.

Example 2. Take the MCM $\Gamma = (S, \Phi)$ in Example 1, and let $s_1 = \{\text{si}, \text{or}, \text{an}\} \in S$. Consider the function f_1 s.t. $\forall s \in S : f_1(s) = 1$ iff $\text{an} \in s$ and $\{\text{or}, \text{cl}\} \cap s \neq \emptyset$. The function f_1 is syntactically expressed by the formula $\Box_I (\mathbf{t}(1) \leftrightarrow ((\text{or} \wedge \text{an}) \vee (\text{cl} \wedge \text{an})))$. Clearly $f_1 \in \Phi$ for it satisfies the three constraints. Hence, we have:

$$(\Gamma, s_1, f_1) \models \text{AXp}(\text{or} \wedge \text{an}, 1) \wedge \text{PImp}(\text{or} \wedge \text{an}, 1) \wedge \text{PImp}(\text{cl} \wedge \text{an}, 1).$$

Meanwhile $(\Gamma, s_1, f_1) \not\models \text{AXp}(\text{cl} \wedge \text{an}, 1)$, because $(\Gamma, s_1, f_1) \not\models \text{cl} \wedge \text{an}$. But consider $s_2 = \{\text{si}, \text{cl}, \text{an}\} \in S$. We have $(\Gamma, s_2, f_1) \models \text{AXp}(\text{cl} \wedge \text{an}, 1)$.

Now we investigate what happens when facing a black box model $\Gamma = (S, \Phi)$. The agent has uncertainty about the actual classifier’s properties. Therefore, it is interesting to draw the distinction between objective and subjective (or epistemic) explanation. Objective explanation coincides with the notion of explanation in the context of white box classifiers defined above. Subjective explanation refers to the agent’s interpretation of the classifier and her explanation of the classifier’s decision in the light of her partial knowledge.

We say the term λ is a *subjective* prime implicant for x , noted $\text{SubPImp}(\lambda, x)$, if the agent knows that λ is a prime implicant for x , that is:

$$\text{SubPImp}(\lambda, x) =_{\text{def}} \Box_F \text{PImp}(\lambda, x).$$

³ It has many names in literature: PI explanation [23], sufficient reason [6]. We adopt the one from [13] for its nice correspondence to contrastive explanation in [12].

Similarly, we say λ is a *subjective* abductive explanation of the actual classification x , noted $\text{SubAXp}(\lambda, x)$, if the agent knows that λ is an abductive explanation of the actual classification x , that is:

$$\text{SubAXp}(\lambda, x) =_{\text{def}} \Box_{\text{F}} \text{AXp}(\lambda, x).$$

It is worth noting that in the case of a white box classifier, if the set of input instances S is finite, we can always find an abductive explanation of the actual classification. That is, for every $\Gamma = (S, \Phi) \in \mathbf{MCM}$, $s \in S$ and $f \in \Phi$:

$$\text{if } S \text{ is finite then } \exists \lambda \in \text{Term such that } (\Gamma, s, f) \models \text{AXp}(\lambda, f(s)).$$

Nonetheless, this result cannot be generalized to the black box case. Indeed, as the following example shows, there is no guarantee for the existence of a subjective explanation of the actual classification. The problem is that the minimality condition can collapse when moving from objective to subjective explanation, since the agent can have more than one classifier in her epistemic state.

Example 3. Let $\Gamma = (S, \Phi)$, f_1 and s_1 be the same as in Example 2. There is no λ such that $(\Gamma, s_1, f_1) \models \Box_{\text{F}} \text{AXp}(\lambda, 1)$. To see this, consider f_2 s.t. $\forall s \in S : f_2(s) = 1$ iff $\{\text{si}, \text{an}\} \subseteq s$. The function f_2 is syntactically expressed by the formula $\Box_{\text{I}}(\text{t}(1) \leftrightarrow (\text{si} \wedge \text{an}))$. Clearly $f_2 \in \Phi$ for it satisfies the three constraints. We have $(\Gamma, s_1, f_2) \models \text{AXp}(\text{si} \wedge \text{an}, 1)$. But there is no term which minimally explains both $f_1(s_1)$ and $f_2(s_1)$. Indeed, $\text{or} \wedge \text{an}$ is not enough for explaining $f_2(s_1)$, $\text{si} \wedge \text{an}$ is not enough for explaining $f_1(s_1)$, and $\text{si} \wedge \text{or} \wedge \text{an}$ fails the minimality condition for both. Therefore, we have

$$(\Gamma, s_1, f_1) \models \text{AXp}(\text{or} \wedge \text{an}, 1) \wedge \bigwedge_{\lambda \in \text{Term}(\{\text{si}, \text{or}, \text{cl}, \text{an}\})} \neg \text{SubAXp}(\lambda, 1).$$

However, this does not mean that the agent knows nothing about the classifier. For instance, she knows that violating the anonymity requirement is a prime implicant for rejection, that is, $(\Gamma, s_1, f_1) \models \text{SubAXp}(\neg \text{an}, 0)$.

To sum up, the four notions of explanation we introduced can be organized in Table 2 along the two dimensions objective vs subjective and local vs global.

Table 2. Notions of prime implicant and abductive explanation

	Local	Global
Objective	$\text{AXp}(\lambda, x)$	$\text{PImp}(\lambda, x)$
Subjective	$\text{SubAXp}(\lambda, x)$	$\text{SubPImp}(\lambda, x)$

5 Dynamic Extension

Before concluding, we are going to present a simple dynamic extension of the language $\mathcal{L}$ by operators of the form $[\varphi]$. They describe the consequences of removing from the actual model all classifiers that do not *globally* satisfy the constraint φ . More generally, they allow us to model the process of gaining new knowledge about the classifier’s properties. The extended modal language $\mathcal{L}^{dyn}$ is defined by the following grammar:

$$\varphi ::= p \mid \mathbf{t}(x) \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \Box_I \varphi \mid \Box_F \varphi \mid [\varphi]\psi,$$

where p ranges over Atm_0 and x ranges over Val .

The new formula $[\varphi]\psi$ has to be read “ ψ holds after having discarded all classifiers that do not globally satisfy the property φ ”. Notice the similar but different notations $[X]$ and $[\varphi]$. For example, $[\{p\}]$, $[\{p, q\}]$ are abbreviations with *ceteris paribus* meaning, while $[p]$, $[p \wedge \neg q]$ are dynamic operators.

The interpretation of the operators $[\varphi]$ relative to a pointed MCM (Γ, s, f) with $\Gamma = (S, \Phi)$, $s \in S$ and $f \in \Phi$ goes as follows:

$$(\Gamma, s, f) \models [\varphi]\psi \iff \text{if } (\Gamma, s, f) \models \Box_I \varphi \text{ then } (\Gamma^\varphi, s, f) \models \psi,$$

where $\Gamma^\varphi = (S^\varphi, \Phi^\varphi)$ is the MCM such that:

$$\begin{aligned} S^\varphi &= S, \\ \Phi^\varphi &= \{f' \in \Phi : \forall s' \in S, (\Gamma, s', f') \models \varphi\}. \end{aligned}$$

The previous update semantics for the operator $[\varphi]$ is reminiscent of the semantics of public announcement logic (PAL) [7, 19]. However, there is an important difference. While PAL has a one-dimensional state elimination semantics, our update semantics operates on a single dimension of the product in an MCM. In particular, it only removes classifiers that do not globally satisfy the constraint φ , without modifying the set S of input instances.

The logics D-PLC and D-WPLC (Dynamic PLC and D-WPLC) extend the logic PLC and WPLC by the dynamic operators $[\varphi]$. They are defined as follows.

Definition 8 (Logics D-PLC and D-WPLC). *We define D-PLC (resp. D-WPLC) to be the extension of PLC (resp. WPLC) of Definition 5 (resp. Definition 6) generated by the following reduction axioms for the dynamic operators $[\varphi]$:*

$$\begin{aligned} [\varphi]p &\leftrightarrow (\Box_I \varphi \rightarrow p) \\ [\varphi]\mathbf{t}(x) &\leftrightarrow (\Box_I \varphi \rightarrow \mathbf{t}(x)) \\ [\varphi]\neg\psi &\leftrightarrow (\Box_I \varphi \rightarrow \neg[\varphi]\psi) \\ [\varphi](\psi_1 \wedge \psi_2) &\leftrightarrow ([\varphi]\psi_1 \wedge [\varphi]\psi_2) \\ [\varphi]\Box_I \psi &\leftrightarrow (\Box_I \varphi \rightarrow \Box_I [\varphi]\psi) \\ [\varphi]\Box_F \psi &\leftrightarrow (\Box_I \varphi \rightarrow \Box_F [\varphi]\psi) \end{aligned}$$

and the following rule of inference:

$$\frac{\varphi_1 \leftrightarrow \varphi_2}{\psi \leftrightarrow \psi[\varphi_1/\varphi_2]} \quad (\mathbf{RE})$$

It is a routine exercise to verify that the equivalences in Definition 8 are valid for the class **MCM** and that the rule of replacement of equivalents (**RE**) preserves validity. We show the validity of the sixth equivalence as an example:

$$\begin{aligned} (\Gamma, s, f) \models [\varphi]\Box_F\psi &\iff \text{if } (\Gamma, s, f) \models \Box_I\varphi \text{ then } (\Gamma^\varphi, s, f) \models \Box_F\psi; \\ &\iff \text{if } (\Gamma, s, f) \models \Box_I\varphi \text{ then } \forall f' \in \Phi^\varphi, (\Gamma^\varphi, s, f') \models \psi; \\ &\iff \text{if } (\Gamma, s, f) \models \Box_I\varphi \text{ then } \forall f' \in \Phi, \\ &\quad (\text{if } \forall s' \in S, (\Gamma, s', f') \models \psi \text{ then } (\Gamma^\varphi, s, f') \models \psi); \\ &\iff \text{if } (\Gamma, s, f) \models \Box_I\varphi \text{ then } \forall f' \in \Phi, \\ &\quad (\text{if } (\Gamma, s, f') \models \Box_I\psi \text{ then } (\Gamma^\varphi, s, f') \models \psi); \\ &\iff \text{if } (\Gamma, s, f) \models \Box_I\varphi \text{ then } \forall f' \in \Phi, (\Gamma, s, f') \models [\varphi]\psi; \\ &\iff (\Gamma, s, f) \models \Box_I\varphi \rightarrow \Box_F[\varphi]\psi. \end{aligned}$$

The completeness of D-PLC and D-WPLC for this class of models follows from Theorem 2 and Corollary 1, in view of the fact that the reduction axioms and the rule of replacement of proved equivalents can be used to find, for any $\mathcal{L}^{dyn}$ -formula, a provably equivalent $\mathcal{L}$ -formula.

Theorem 8. *Let Atm_0 be finite. Then, the logic D-PLC is sound and complete relative to the class **MCM**.*

Theorem 9. *Let Atm_0 be infinite. Then, the logic D-WPLC is sound and complete relative to the class **MCM**.*

The following decidability result is a consequence of Theorem 7 and the fact that via the reduction axioms in Definition 8 we can find a reduction of satisfiability checking of $\mathcal{L}^{dyn}$ -formulas to satisfiability checking of $\mathcal{L}$ -formulas.

Theorem 10. *Checking satisfiability of $\mathcal{L}^{dyn}$ -formulas relative to **MCM** is decidable.*

Let us end up with the paper example to illustrate to expressive power of our dynamic extension.

Example 4. Let $\Gamma = (S, \Phi)$, f_1 and s_1 be the same as in Example 2. We have

$$(\Gamma, s_1, f_1) \models [(\text{or} \wedge \text{an}) \rightarrow \text{t}(1)]\Box_F \bigvee_{\lambda \subseteq (\text{or} \wedge \text{an})} \mathbf{Axp}(\lambda, 1).$$

This means that after having discarded all classifiers which do not take $(\text{or} \wedge \text{an})$ as an implicant for acceptance of a paper, the agent knows that there must be a part of $\text{or} \wedge \text{an}$ that abductively explains the acceptance of the paper s_1 .

6 Conclusion

We have presented a product modal logic which supports reasoning about (i) partial knowledge and uncertainty of a classifier’s properties and, (ii) objective and subjective explanations of a classifier’s decision. Moreover, we have studied a dynamic extension of the logic which allows us to represent the event of gaining new knowledge about the classifier’s properties.

Our logic is intrinsically single-agent: it models the uncertainty of one agent about the actual classifier’s properties. In future work, we plan to generalize our framework to the multi-agent setting. The extension would result in a multi-relational product semantics in which every agent has her own epistemic indistinguishability relation which commutes with the input instance dimension (the equivalence relation $\sim_{\square_I}$ in Definition 3 of MDM). We also plan to enrich this semantics with a knowledge update mechanism in the spirit of Section 5. This would allow us to represent exchange of information between agents with an explanatory purpose, which is named dialogical explanation by philosophers [24] and interactive explanation by researchers in the XAI domain [1, 17].

Acknowledgments. Support from the ANR-3IA Artificial and Natural Intelligence Toulouse Institute (ANITI) is gratefully acknowledged.

References

1. Amershi, S., Cakmak, M., Knox, W., Kulesza, T.: Power to the people: the role of humans in interactive machine learning. *AI Mag.* **35**(4), 105–120 (2014)
2. Audemard, G., Bellart, S., Bounia, L., Koriche, F., Lagniez, J.M., Marquis, P.: On the computational intelligibility of boolean classifiers. In: *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, vol. 18, pp. 74–86 (2021)
3. Baltag, A., van Benthem, J.: A simple logic of functional dependence. *J. Philos. Log.* **50**(5), 939–1005 (2021). <https://doi.org/10.1007/s10992-020-09588-z>
4. Bezhanishvili, N., Hodkinson, I.M.: All normal extensions of S5-squared are finitely axiomatizable. *Stud. Logica.* **78**(3), 443–457 (2004)
5. Bezhanishvili, N., Marx, M.: All proper normal extensions of S5-square have the polynomial size model property. *Stud. Logica.* **73**(3), 367–382 (2003). <https://doi.org/10.1023/A:1023383112908>
6. Darwiche, A., Hirth, A.: On the reasons behind decisions. In: *ECAI 2020–24th European Conference on Artificial Intelligence. Frontiers in Artificial Intelligence and Applications*, vol. 325, pp. 712–720. IOS Press (2020)
7. van Ditmarsch, H., van Der Hoek, W., Kooi, B.: *Dynamic Epistemic Logic*, Synthese Library, vol. 337 Springer (2007). <https://doi.org/10.1007/978-1-4020-5839-4>
8. Gabbay, D.M., Kurucz, A., Wolter, F., Zakharyashev, M.: *Many-dimensional modal logics: theory and applications*. Elsevier (2003)
9. Grossi, D., Lorini, E., Schwarzentruher, F.: The ceteris paribus structure of logics of game forms. *J. Artif. Intell. Res.* **53**, 91–126 (2015)
10. Halpern, J.Y.: The effect of bounding the number of primitive propositions and the depth of nesting on the complexity of modal logic. *Artif. Intell.* **75**(2), 361–372 (1995)

11. Hempel, C.G., Oppenheim, P.: Studies in the logic of explanation. *Philos. sci.* **15**(2), 135–175 (1948)
12. Ignatiev, A., Narodytska, N., Asher, N., Marques-Silva, J.: From contrastive to abductive explanations and back again. In: Baldoni, M., Bandini, S. (eds.) *AIxIA 2020. LNCS (LNAI)*, vol. 12414, pp. 335–355. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-77091-4_21
13. Ignatiev, A., Narodytska, N., Marques-Silva, J.: Abduction-based explanations for machine learning models. In: *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 1511–1519 (2019)
14. Kment, B.: Counterfactuals and explanation. *Mind* **115**(458), 261–310 (2006)
15. Liu, X., Lorini, E.: A logic for binary classifiers and their explanation. In: Baroni, P., Benz Müller, C., Wáng, Y.N. (eds.) *CLAR 2021. LNCS (LNAI)*, vol. 13040, pp. 302–321. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-89391-0_17
16. Lundberg, S., Lee, S.I.: A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874* (2017)
17. Miller, T.: Explanation in artificial intelligence: insights from the social sciences. *Artif. Intell.* **267**, 1–38 (2019)
18. Molnar, C.: *Interpretable Machine Learning*. Lulu. com (2020)
19. Plaza, J.: Logics of public communications. *Synthese* **158**(2), 165–179 (2007). <https://doi.org/10.1007/s11229-007-9168-7>
20. Quine, W.V.: A way to simplify truth functions. *Am. Math. Mon.* **62**(9), 627–631 (1955)
21. Ribeiro, M.T., Singh, S., Guestrin, C.: Why should i trust you? Explaining the predictions of any classifier. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144 (2016)
22. Ribeiro, M.T., Singh, S., Guestrin, C.: Anchors: high-precision model-agnostic explanations. In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, vol. 32 (2018)
23. Shih, A., Choi, A., Darwiche, A.: Formal verification of bayesian network classifiers. In: *International Conference on Probabilistic Graphical Models*, pp. 427–438. PMLR (2018)
24. Walton, D.: A new dialectical theory of explanation. *Philos. Explor.* **7**(1), 71–89 (2004)
25. Yang, F., Väänänen, J.: Propositional logics of dependence. *Ann. Pure Appl. Logic* **167**(7), 557–589 (2016)
26. You, S., Ding, D., Canini, K., Pfeifer, J., Gupta, M.: Deep lattice networks and partial monotonic functions. *Advances in neural information processing systems* 30 (2017)



What Kinds of Connectives Cause the Difference Between Intuitionistic Predicate Logic and the Logic of Constant Domains?

Naosuke Matsuda¹ and Kento Takagi²(✉) 

¹ Department of Engineering, Niigata Institute of Technology,
Fujihashi, Kashiwazaki City, Niigata 945-1195, Japan

² Department of Computer Science, Tokyo Institute of Technology,
Meguro-ku, Tokyo, Ookayama 152-8522, Japan
`kento.takagi.aa@gmail.com`

Abstract. It is known that intuitionistic Kripke semantics can be generalized so that it can treat arbitrary propositional connectives characterized by truth functions. Our previous work studied how the choice of propositional connectives changes the relation between classical and intuitionistic propositional logics, and showed that the set of valid sequents in intuitionistic propositional logic coincides with the set of valid sequents in classical propositional logic if and only if the connectives we choose are all monotonic. In this paper, we extend the generalized Kripke semantics to first-order logic and study how the choice of connectives changes the relation between intuitionistic predicate logic and the logic of constant domains. In the case of the usual connectives $\neg$, $\wedge$, $\vee$, and $\rightarrow$, it is well-known that the presence of disjunction causes the difference between these two logics. Generalizing this result to general propositional connectives, we give a simple necessary and sufficient condition for the set of valid sequents in intuitionistic predicate logic to coincide with the set of valid sequents in the logic of constant domains.

Keywords: Kripke semantics · Propositional connective · Intuitionistic predicate logic · The logic of constant domains

1 Introduction

1.1 Background—Generalized Kripke Semantics

In [4], Kripke provided the intuitionistic interpretation for formulas built out of the usual propositional connectives $\neg$, $\rightarrow$, $\wedge$, and $\vee$. The notion of validity in intuitionistic logic can be defined with this interpretation. Rousseau [8] and Geuvers and Hurkens [2] extended the intuitionistic interpretation so that it can treat arbitrary propositional connectives characterized by truth functions. Their idea is very simple: when c is a propositional connective and t_c is the

truth function associated with c , then the value $\|c(\alpha_1, \dots, \alpha_n)\|_w$ of the formula $c(\alpha_1, \dots, \alpha_n)$ at world w is defined as follows:

$$\|c(\alpha_1, \dots, \alpha_n)\|_w = 1 \text{ if and only if } t_c(\|\alpha_1\|_v, \dots, \|\alpha_n\|_v) = 1 \text{ for every } v \succeq w.$$

This definition generalizes the definition of the interpretation for formulas with the usual connectives. For example, in the case where c is implication, $t_{\rightarrow}(\|\alpha_1\|_v, \|\alpha_2\|_v) = 1$ holds for every $v \succeq w$ if and only if either $\|\alpha_1\|_v = 0$ or $\|\alpha_2\|_v = 1$ holds for every $v \succeq w$. In the case where c is conjunction, it can be easily verified that the condition that $t_{\wedge}(\|\alpha_1\|_v, \|\alpha_2\|_v) = 1$ holds for every $v \succeq w$, that is, both $\|\alpha_1\|_v = 1$ and $\|\alpha_2\|_v = 1$ hold for every $v \succeq w$, is equivalent to the usual condition that both $\|\alpha_1\|_w = 1$ and $\|\alpha_2\|_w = 1$ hold. The same holds for disjunction.

It is well-known that the relation between intuitionistic logic and classical logic changes depending on what kinds of propositional connectives are used. We consider an important relation between classical logic and intuitionistic logic—the inclusion relationship between the set of classically valid sequents and the set of intuitionistically valid sequents. For a given set $\mathcal{C}$ of propositional connectives, let $\text{CLS}(\mathcal{C})$ denote the set of classically valid sequents built out of connectives in $\mathcal{C}$ and $\text{ILS}(\mathcal{C})$ the set of Kripke-valid sequents built out of connectives in $\mathcal{C}$. Then, $\text{ILS}(\mathcal{C}) \subseteq \text{CLS}(\mathcal{C})$ always holds because classical models can be regarded as one-world Kripke models. However, the converse inclusion depends on $\mathcal{C}$. For example, $\text{ILS}(\{\neg\}) \subsetneq \text{CLS}(\{\neg\})$ and $\text{ILS}(\{\wedge, \vee\}) = \text{CLS}(\{\wedge, \vee\})$ hold¹. Then, there arises a natural question: for what $\mathcal{C}$, does $\text{ILS}(\mathcal{C}) = \text{CLS}(\mathcal{C})$ hold? We answered this question in [3]:

Theorem. $\text{ILS}(\mathcal{C}) = \text{CLS}(\mathcal{C})$ if and only if every $c \in \mathcal{C}$ is monotonic².

1.2 Predicate Logic with General Propositional Connectives

Generalized Kripke semantics introduced in Sect. 1.1, as well as classical semantics, can be extended to first-order logic by adding $\forall$ and $\exists$ with the usual interpretations. Furthermore, in the case of predicate logic, there is an important intermediate logic between classical logic and intuitionistic logic, called the *logic of constant domains*, or CD. CD is semantically characterized by Kripke models with constant domains.

This paper analyzes how the choice of connectives changes the relation between classical first-order logic, intuitionistic first-order logic and the logic of constant domains in terms of validity of sequents. Let $\text{FOCLS}(\mathcal{C})$ be the set of classically valid *predicate* sequents built out of connectives in $\mathcal{C}$, $\text{FOILS}(\mathcal{C})$, the set of Kripke-valid *predicate* sequents built out of connectives in $\mathcal{C}$, and $\text{FOCDS}(\mathcal{C})$, the set of CD-valid *predicate* sequents built out of connectives in $\mathcal{C}$ ³. Then, by the definition of each kind of model, $\text{FOILS}(\mathcal{C}) \subseteq \text{FOCDS}(\mathcal{C}) \subseteq$

¹ For example, $\neg\neg p \Rightarrow p \in \text{CLS}(\{\neg\}) \setminus \text{ILS}(\{\neg\})$.

² For the definition of monotonicity, see Sect. 2.1-2.2.

³ CD-valid sequents are those sequents that are valid in all constant domain Kripke models.

FOCLS($\mathcal{C}$) always holds. However, the converse inclusion relationships depend on $\mathcal{C}$, and hence we have two questions:

- (i) For what $\mathcal{C}$, does FOILS($\mathcal{C}$) = FOCDS($\mathcal{C}$) hold?
- (ii) For what $\mathcal{C}$, does FOCDS($\mathcal{C}$) = FOCLS($\mathcal{C}$) hold?

For (ii), we obtain the following theorem by simply extending the theorem in Sect. 1.1.

Theorem. FOCDS($\mathcal{C}$) = FOCLS($\mathcal{C}$) if and only if every connective in $\mathcal{C}$ is monotonic.

The proof is easily obtained by extending that of the theorem in Sect. 1.1. The reader who wants to see a detailed definition of classical first-order semantics with general propositional connectives and a detailed proof of this theorem is referred to [5].

In this paper, we consider the remaining question, (i). It is well-known that, in the case of the usual connectives, the presence of disjunction causes the difference between FOILS($\mathcal{C}$) and FOCDS($\mathcal{C}$) (cf., e.g., [1]). That is, for $\mathcal{C} \subseteq \{\neg, \wedge, \vee, \rightarrow\}$, FOILS($\mathcal{C}$) = FOCDS($\mathcal{C}$) if and only if $\vee \notin \mathcal{C}$. However, this result does not tell us what *property* of disjunction causes the difference. By considering general connectives, our main result clarifies what property of connectives cause the difference between FOILS($\mathcal{C}$) and FOCDS($\mathcal{C}$):

Theorem. FOILS($\mathcal{C}$) = FOCDS($\mathcal{C}$) if and only if every $c \in \mathcal{C}$ is supermultiplicative⁴.

Furthermore, combining the preceding two theorems, we obtain a necessary and sufficient condition for FOILS($\mathcal{C}$) to coincide with FOCLS($\mathcal{C}$):

Corollary. FOILS($\mathcal{C}$) = FOCLS($\mathcal{C}$) if and only if every connective $c \in \mathcal{C}$ is both monotonic and supermultiplicative.

Remark 1. Although we shall not describe it explicitly, we can easily see from the proofs that the above results also hold if we use sequents with a single succedent instead of those with multi-succedents. For example, the proof of the second theorem also shows that FOILS₁($\mathcal{C}$) = FOCDS₁($\mathcal{C}$) if and only if every connective in $\mathcal{C}$ is supermultiplicative, where FOILS₁($\mathcal{C}$) = $\{\Gamma \Rightarrow \Delta \in \text{FOILS}(\mathcal{C}) \mid \Delta \text{ consists of a single formula}\}$ and FOCDS₁($\mathcal{C}$) = $\{\Gamma \Rightarrow \Delta \in \text{FOCDS}(\mathcal{C}) \mid \Delta \text{ consists of a single formula}\}$.

2 Preliminaries

2.1 Truth Functions

We introduce some notions concerning truth functions briefly. For more detailed presentations, see Sect. 1.1 in [3].

⁴ For the definition of supermultiplicativity, see Sect. 2.1-2.2.

For a sequence $\mathbf{a} \in \{0, 1\}^n$ of truth values of length $n \geq 0$, we denote by $\mathbf{a}[i]$ the i -th component of $\mathbf{a}$. We denote $\langle 1, \dots, 1 \rangle$ and $\langle 0, \dots, 0 \rangle$ by $\mathbf{1}$ and $\mathbf{0}$, respectively. (The lengths of $\mathbf{1}$ and $\mathbf{0}$ as sequences depend on the context.) An n -ary *truth function* is a function from $\{0, 1\}^n$ to $\{0, 1\}$.

We define a partial order $\sqsubseteq$ on $\{0, 1\}^n$ by $\mathbf{a} \sqsubseteq \mathbf{b} \iff \mathbf{a}[i] \leq \mathbf{b}[i]$ for all $1 \leq i \leq n$. For $\mathbf{a}, \mathbf{b} \in \{0, 1\}^n$, $\mathbf{a} \sqcap \mathbf{b}$ denotes the infimum of $\{\mathbf{a}, \mathbf{b}\}$, so that $(\mathbf{a} \sqcap \mathbf{b})[i] = \min(\mathbf{a}[i], \mathbf{b}[i])$ for all $1 \leq i \leq n$. An n -ary truth function f is said to be *monotonic* if for any $\mathbf{a}, \mathbf{b} \in \{0, 1\}^n$, $\mathbf{a} \sqsubseteq \mathbf{b}$ implies $f(\mathbf{a}) \leq f(\mathbf{b})$. An n -ary truth function f is said to be *supermultiplicative* if for any $\mathbf{a}, \mathbf{b} \in \{0, 1\}^n$, $f(\mathbf{a}) = f(\mathbf{b}) = 1$ implies $f(\mathbf{a} \sqcap \mathbf{b}) = 1$ ⁵.

2.2 Formulas and Sequents

A (*propositional*) *connective* is a symbol with an associated truth function. For a propositional connective c , we denote by t_c the truth function associated with c and by $\text{ar}(c)$ the arity of t_c . A connective is said to be *monotonic* if its truth function is monotonic. A connective is said to be *supermultiplicative* if its truth function is supermultiplicative. Note that it can be easily verified that for a connective c with $\text{ar}(c) \leq 2$, c is supermultiplicative if and only if c is neither disjunction $\vee$ nor exclusive disjunction $\oplus$ ⁶.

Fix a set $\mathcal{C}$ of propositional connectives. An *atomic formula* is an expression of the form $p(x_1, \dots, x_n)$, where p is an n -ary predicate symbol and $x_1, \dots, x_n$ are individual variables⁷. The set $\text{FOFml}(\mathcal{C})$ of (*predicate*) *formulas* with propositional connectives in $\mathcal{C}$ is defined inductively as follows:

- if α is an atomic formula, then $\alpha \in \text{FOFml}(\mathcal{C})$;
- if $c \in \mathcal{C}$ and $\alpha_1, \dots, \alpha_{\text{ar}(c)} \in \text{FOFml}(\mathcal{C})$, then $c(\alpha_1, \dots, \alpha_{\text{ar}(c)}) \in \text{FOFml}(\mathcal{C})$;
- if $\alpha \in \text{FOFml}(\mathcal{C})$ and x is an individual variable, then $\forall x \alpha \in \text{FOFml}(\mathcal{C})$ and $\exists x \alpha \in \text{FOFml}(\mathcal{C})$.

We denote by $\text{FV}(\alpha)$ the set of free variables of α . In particular, $\text{FV}(c(\alpha_1, \dots, \alpha_{\text{ar}(c)}))$ is $\text{FV}(\alpha_1) \cup \dots \cup \text{FV}(\alpha_{\text{ar}(c)})$.

A (*predicate*) *sequent* is an expression of the form $\Gamma \Rightarrow \Delta$, where Γ and Δ are sets of formulas. We denote by $\text{FOSqt}(\mathcal{C})$ the set $\{\Gamma \Rightarrow \Delta \mid \Gamma, \Delta \subseteq \text{FOFml}(\mathcal{C})\}$. If $\Gamma = \{\alpha_1, \dots, \alpha_n\}$ and $\Delta = \{\beta_1, \dots, \beta_m\}$, we often omit the braces and simply write $\alpha_1, \dots, \alpha_n \Rightarrow \beta_1, \dots, \beta_m$.

2.3 Kripke Semantics

Kripke models are defined in the same way as in the case of first-order intuitionistic logic with the usual propositional connectives. That is, a Kripke model is

⁵ Note that $f(\mathbf{a}) = f(\mathbf{b}) = 1 \Rightarrow f(\mathbf{a} \sqcap \mathbf{b}) = 1$ if and only if $f(\mathbf{a}) \sqcap f(\mathbf{b}) \leq f(\mathbf{a} \sqcap \mathbf{b})$.

⁶ The truth function of exclusive disjunction is defined as follows: $t_{\oplus}(x, y) = 1$ if and only if $x \neq y$.

⁷ Although we assume the language has no function symbols and no constant symbols, all arguments in this paper work as well in case the language has function symbols and constant symbols.

a tuple $\langle W, \preceq, D, I \rangle$, where $\langle W, \preceq \rangle$ is a non-empty pre-ordered set; D assigns to each $w \in W$ a non-empty set $D(w)$, called the *individual domain at w* ; and I is a function, called the *interpretation function*, that assigns to each pair $\langle w, p \rangle$ of a world and an n -ary predicate symbol a function $I(w, p)$ from $D(w)^n$ to $\{0, 1\}$. As usual, D is required to be monotonic and I should satisfy the hereditary condition. That is, we assume that for any $w, v \in W$, if $w \preceq v$, then $D(w) \subseteq D(v)$ and $I(w, p)(a_1, \dots, a_n) \leq I(v, p)(a_1, \dots, a_n)$ for any n -ary predicate symbol and any $a_1, \dots, a_n \in D(w)$. An *assignment* in $D(w)$ is a function which assigns to each individual variable an element in $D(w)$. As usual, a Kripke model $\mathcal{K} = \langle W, \preceq, D, I \rangle$ is said to be *constant domain* if $D(w) = D(v)$ for all $w, v \in W$.

Let $\mathcal{K} = \langle W, \preceq, D, I \rangle$ be a Kripke model. The *value* $\|\alpha\|_{\mathcal{K}, w}^\rho \in \{0, 1\}$ of a formula $\alpha \in \text{FOFml}(\mathcal{C})$ at a world $w \in W$ with respect to an assignment ρ in $D(w)$ is defined as follows:

- $\|p(x_1, \dots, x_n)\|_{\mathcal{K}, w}^\rho = I(w, p)(\rho(x_1), \dots, \rho(x_n))$;
- $\|c(\alpha_1, \dots, \alpha_{\text{ar}(c)})\|_{\mathcal{K}, w}^\rho = 1$ if and only if $t_c(\|\alpha_1\|_{\mathcal{K}, v}^\rho, \dots, \|\alpha_{\text{ar}(c)}\|_{\mathcal{K}, v}^\rho) = 1$ for all $v \succeq w$;
- $\|\forall x \alpha\|_{\mathcal{K}, w}^\rho = 1$ if and only if $\|\alpha\|_{\mathcal{K}, v}^{\rho[x \mapsto a]} = 1$ for all $v \succeq w$ and all $a \in D(v)$;
- $\|\exists x \alpha\|_{\mathcal{K}, w}^\rho = 1$ if and only if $\|\alpha\|_{\mathcal{K}, w}^{\rho[x \mapsto a]} = 1$ for some $a \in D(w)$,

where $\rho[x \mapsto a]$ is the assignment in $D(w)$ which maps x to a and is equal to ρ everywhere else. As in the case of the usual connectives, the hereditary condition easily extends to any formula:

Lemma 2. *For any $\alpha \in \text{FOFml}(\mathcal{C})$, any Kripke model $\mathcal{K} = \langle W, \preceq, D, I \rangle$, any $w, v \in W$, and any assignment ρ in $D(w)$, if $w \preceq v$ then $\|\alpha\|_{\mathcal{K}, w}^\rho \leq \|\alpha\|_{\mathcal{K}, v}^\rho$.*

For a Kripke model $\mathcal{K} = \langle W, \preceq, D, I \rangle$, a possible world $w \in W$, an assignment ρ in $D(w)$ and a sequent $\Gamma \Rightarrow \Delta \in \text{FOSqt}(\mathcal{C})$, the *value* $\|\Gamma \Rightarrow \Delta\|_{\mathcal{K}, w}^\rho \in \{0, 1\}$ of $\Gamma \Rightarrow \Delta$ at w with respect to ρ is defined by

$$\|\Gamma \Rightarrow \Delta\|_{\mathcal{K}, w}^\rho = \begin{cases} 0 & \text{if } \|\alpha\|_{\mathcal{K}, w}^\rho = 1 \text{ for all } \alpha \in \Gamma \text{ and } \|\beta\|_{\mathcal{K}, w}^\rho = 0 \text{ for all } \beta \in \Delta \\ 1 & \text{otherwise.} \end{cases}$$

For a Kripke model $\mathcal{K} = \langle W, \preceq, D, I \rangle$, a sequent $\Gamma \Rightarrow \Delta \in \text{FOSqt}(\mathcal{C})$ is *valid* in $\mathcal{K}$ (notation: $\mathcal{K} \models \Gamma \Rightarrow \Delta$) if $\|\Gamma \Rightarrow \Delta\|_{\mathcal{K}, w}^\rho = 1$ for all $w \in W$ and all assignment ρ in $D(w)$. A sequent $\Gamma \Rightarrow \Delta \in \text{FOSqt}(\mathcal{C})$ is *Kripke-valid* (resp. *CD-valid*) if it is valid in all Kripke models (resp. constant domain Kripke models). We denote by $\text{FOILS}(\mathcal{C})$ the set $\{\Gamma \Rightarrow \Delta \in \text{FOSqt}(\mathcal{C}) \mid \Gamma \Rightarrow \Delta \text{ is Kripke-valid}\}$ and by $\text{FOCDS}(\mathcal{C})$ the set $\{\Gamma \Rightarrow \Delta \in \text{FOSqt}(\mathcal{C}) \mid \Gamma \Rightarrow \Delta \text{ is CD-valid}\}$. Then, it immediately follows that $\text{FOILS}(\mathcal{C}) \subseteq \text{FOCDS}(\mathcal{C})$ for any $\mathcal{C}$.

Here, for later use, we establish some notations. The value $\|\alpha\|_{\mathcal{K}, w}^\rho$ depends only on the values of ρ on $\text{FV}(\alpha)$. Hence, for a partial function ρ from the set of individual variables to $D(w)$, $\|\alpha\|_{\mathcal{K}, w}^\rho$ can be defined if $\rho(x)$ is defined for all $x \in \text{FV}(\alpha)$. For such (partial) assignments, we also define $\rho[x \mapsto a]$ to be the

function which maps x to a and is equal to ρ on $\text{dom}(\rho) \setminus \{x\}$. We use $\emptyset$ to denote the empty assignment $\emptyset \rightarrow D(w)$. For example, for a Kripke model $\mathcal{K} = \langle W, \preceq, D, I \rangle$, $w \in W$ and $a, b \in D(w)$, we have $\|p(x, y)\|_{\mathcal{K}, w}^{\emptyset[x \mapsto a][y \mapsto b]} = I(w, p)(a, b)$.

If $\vec{\alpha}$ denotes a sequence of formulas $\alpha_1, \dots, \alpha_n$, then we denote by $\|\vec{\alpha}\|_{\mathcal{K}, w}^\rho$ the sequence $\langle \|\alpha_1\|_{\mathcal{K}, w}^\rho, \dots, \|\alpha_n\|_{\mathcal{K}, w}^\rho \rangle$. For example, if $\vec{\beta} = \beta_1, \dots, \beta_n$, then $t_c(\|\vec{\beta}\|_{\mathcal{K}, w}^\rho) = t_c(\|\beta_1\|_{\mathcal{K}, w}^\rho, \dots, \|\beta_n\|_{\mathcal{K}, w}^\rho)$.

3 Condition for $\text{FOILS}(\mathcal{C}) = \text{FOCLS}(\mathcal{C})$

In this section, we show the following main theorem:

Theorem 3. $\text{FOILS}(\mathcal{C}) = \text{FOCLS}(\mathcal{C})$ if and only if every connective in $\mathcal{C}$ is supermultiplicative.

We show the “if” part in Sect. 3.1 and the “only if” part in Sect. 3.2.

3.1 The “if” Part

Here, we show that if every connective in $\mathcal{C}$ is supermultiplicative, then $\text{FOILS}(\mathcal{C}) = \text{FOCLS}(\mathcal{C})$. First, for later use, we remark that the supermultiplicativity can be extended to any number of arguments:

Lemma 4. *If a connective c is supermultiplicative, then c satisfies the following condition: for all $n \geq 1$ and all $\mathbf{a}_1, \dots, \mathbf{a}_n \in \{0, 1\}^{\text{ar}(c)}$, if $t_c(\mathbf{a}_1) = \dots = t_c(\mathbf{a}_n) = 1$ then $t_c(\mathbf{a}_1 \sqcap \dots \sqcap \mathbf{a}_n) = 1$.*

Proof. This lemma can be easily proved by induction on n .

Assume that every connective in $\mathcal{C}$ is supermultiplicative. Since $\text{FOILS}(\mathcal{C}) \subseteq \text{FOCLS}(\mathcal{C})$ holds, in order to prove $\text{FOILS}(\mathcal{C}) = \text{FOCLS}(\mathcal{C})$, it suffices to show the converse inclusion, and hence it suffices to show the following claim: if $\mathcal{K} \not\models \Gamma \Rightarrow \Delta$ for some $\Gamma \Rightarrow \Delta \in \text{FOSqt}(\mathcal{C})$ and some Kripke model $\mathcal{K} = \langle W, \preceq, D, I \rangle$, then there is a constant domain Kripke model $\mathcal{K}''$ such that $\mathcal{K}'' \not\models \Gamma \Rightarrow \Delta$. We show this claim by generalizing the method in [1], which is used to prove the claim for the usual connectives: for $\mathcal{C} \subseteq \{\neg, \wedge, \rightarrow\}$, $\text{FOILS}(\mathcal{C}) = \text{FOCLS}(\mathcal{C})$ holds.

Before describing the proof, we introduce some definitions. Let $\langle A, \preceq \rangle$ be a pre-ordered set. A *path* from $a \in A$ is a maximal linear subset of $\{b \in A \mid b \succeq a\}$. For $a \in A$ and $B \subseteq A$, we say B *bars* a (notation: $a \dashv B$) if, for any path $\mathcal{P}$ from a , $B \cap \mathcal{P} \neq \emptyset$ holds.

Here, before formal discussions, we give an overview of how to transform $\mathcal{K}$ into $\mathcal{K}''$. The transformation is divided into two steps:

- (1) Transform the Kripke model $\mathcal{K} = \langle W, \preceq, D, I \rangle$ into a Kripke model $\mathcal{K}' = \langle W', \preceq', D', I' \rangle$ such that
 - $\mathcal{K}' \not\models \Gamma \Rightarrow \Delta$;
 - $\mathcal{K}'$ is a tree;

- for any $\alpha \in \text{FOFml}(\mathcal{C})$, any $w' \in W'$, and any assignment ρ' in $D'(w')$, $\|\alpha\|_{\mathcal{K}', w'}^{\rho'} = 1$ if and only if $w' \vdash \{v' \succeq' w' \mid \|\alpha\|_{\mathcal{K}', v'}^{\rho'} = 1\}$.

Step (1) is the same as the first step in the transformation of a Kripke model into a Beth model in [4].

- (2) Transform the Kripke model $\mathcal{K}' = \langle W', \preceq', D', I' \rangle$ into a constant domain Kripke model $\mathcal{K}'' = \langle W', \preceq', D'', I'' \rangle$ in which the values of formulas in $\mathcal{K}'$ are preserved in some sense (that is, in the sense explained in the next paragraph), so that $\mathcal{K}'' \not\models \Gamma \Rightarrow \Delta$ holds. We define the domain D'' of $\mathcal{K}''$ to be a certain set of partial functions from W' into the union of all domains of $\mathcal{K}'$. By this definition, quantifications over individuals in the variable domains of $\mathcal{K}'$ can be translated into quantifications over individuals in the constant domains.

Here, we outline a proof of $\mathcal{K}'' \not\models \Gamma \Rightarrow \Delta$. Just for the sake of simplicity, we consider the case in which $\Gamma = \emptyset$ and $\Delta = \{\alpha\}$, where α has only one free variable, say, x . Since $\mathcal{K}' \not\models \Gamma \Rightarrow \Delta$, that is, $\mathcal{K}' \not\models \Rightarrow \alpha$, if we denote the root of W' by w'_* , then $\|\alpha\|_{\mathcal{K}', w'_*}^{\emptyset[x \mapsto a]} = 0$ for some $a \in D'(w'_*)$. Hence, in order to prove that $\mathcal{K}'' \not\models \Gamma \Rightarrow \Delta$, it suffices to show that there is some $\mathcal{F}$ in D'' such that $\|\alpha\|_{\mathcal{K}'', w'_*}^{\emptyset[x \mapsto \mathcal{F}]} = \|\alpha\|_{\mathcal{K}', w'_*}^{\emptyset[x \mapsto a]}$. To prove this, we shall show that the values of formulas in $\mathcal{K}'$ are preserved in $\mathcal{K}''$ in the following sense: α is true at w' in $\mathcal{K}''$ with x interpreted as a function $\mathcal{F} \in D''$ if and only if for any $v' \succeq' w'$ such that $v' \in \text{dom}(\mathcal{F})$ (where $\text{dom}(\mathcal{F})$ denotes the domain of $\mathcal{F}$), α is true at v' in $\mathcal{K}'$ with x interpreted as $\mathcal{F}(v') \in D'(v')$. (Lemma 8 below states this preservation formally.) The interpretation function I'' of $\mathcal{K}''$ is defined so that this preservation property holds in the case where α is atomic. Using the second and third properties of $\mathcal{K}'$ in step (1) and the assumption that every connective is supermultiplicative, we can extend the preservation property to any formula.

Step (2) is a simplification of the method in [1] for transforming a Beth model into a constant domain Kripke model⁸.

Now, we describe how to transform $\mathcal{K}$ into $\mathcal{K}''$ formally. Since $\mathcal{K} \not\models \Gamma \Rightarrow \Delta$, there exist some $w_* \in W$ and some assignment ρ_* in $D(w_*)$ such that $\|\Gamma \Rightarrow \Delta\|_{\mathcal{K}, w_*}^{\rho_*} = 0$. First, we transform $\mathcal{K}$ into a tree Kripke model $\mathcal{K}' = \langle W', \preceq', D', I' \rangle$.

Definition 5. Let *Last* denote the function which assigns to each non-empty finite sequence of elements of W its last component, so that $\text{Last}(w_0, \dots, w_n) = w_n$. Let the Kripke model $\mathcal{K}' = \langle W', \preceq', D', I' \rangle$ be such that

- $W' = \{\langle w_*, w_1, \dots, w_n \rangle \mid n \geq 0, w_1, \dots, w_n \in W, w_* \preceq w_1 \preceq \dots \preceq w_n\}$;

⁸ In [1], $\mathcal{K}$ is transformed into a Beth model, and then the Beth model is transformed into a constant domain Kripke model. In contrast, we do not introduce a Beth model because it is not necessary for the proof, and instead transform $\mathcal{K}$ into Kripke model $\mathcal{K}'$, which plays essentially the same role as the Beth model.

- $w' \preceq' v'$ if and only if w' is an initial segment of v' , that is, $\langle w_*, w_1, \dots, w_n \rangle \preceq' \langle w_*, v_1, \dots, v_m \rangle$ if and only if $n \leq m$ and $w_i = v_i$ for all $i = 1, \dots, n$;
- $D'(w') = D(\text{Last}(w'))$;
- $I'(w') = I(\text{Last}(w'), p)$.

We denote by w'_* the minimum element $\langle w_* \rangle$.

Then, it can be shown that $\mathcal{K}'$ has the following two properties (cf. [4]):

- (I) for any $\alpha \in \text{FOFml}(\mathcal{C})$, any $w' \in W'$, and any assignment ρ' in $D'(w') = D(\text{Last}(w'))$, $\|\alpha\|_{\mathcal{K}', w'}^{\rho'} = \|\alpha\|_{\mathcal{K}, \text{Last}(w')}^{\rho'}$;
- (II) for any $\alpha \in \text{FOFml}(\mathcal{C})$, any $w' \in W'$, and any assignment ρ' in $D'(w')$, $\|\alpha\|_{\mathcal{K}', w'}^{\rho'} = 1$ if and only if $w' \dashv \{v' \succeq' w' \mid \|\alpha\|_{\mathcal{K}', v'}^{\rho'} = 1\}$.

In particular, by (I), it holds that $\|\Gamma \Rightarrow \Delta\|_{\mathcal{K}', w'_*}^{\rho_*} = \|\Gamma \Rightarrow \Delta\|_{\mathcal{K}, w_*}^{\rho_*} = 0$.

Next, we transform $\mathcal{K}'$ into the constant domain Kripke model $\mathcal{K}''$.

Definition 6. Let the constant domain Kripke model $\mathcal{K}'' = \langle W', \preceq', D'', I'' \rangle$ be such that

- the pre-ordered set $\langle W', \preceq' \rangle$ is the same as that of $\mathcal{K}'$;
- D'' is the set of those partial functions $\mathcal{F}$ from W' to $\bigcup_{w' \in W'} D'(w')$ which satisfy the following conditions:
 - $w'_* \dashv \text{dom}(\mathcal{F})$ (where $\text{dom}(\mathcal{F})$ denotes the domain of $\mathcal{F}$);
 - $\text{dom}(\mathcal{F})$ is an upward-closed subset of W' ;
 - $\mathcal{F}(w') \in D'(w')$ for any $w' \in \text{dom}(\mathcal{F})$;
 - if $w' \in \text{dom}(\mathcal{F})$ and $w' \preceq' v'$ then $\mathcal{F}(w') = \mathcal{F}(v')$.
- $I''(w', p)(\mathcal{F}_1, \dots, \mathcal{F}_n) = 1$ if and only if for any $v' \succeq' w'$ such that $v' \in \bigcap_{i=1, \dots, n} \text{dom}(\mathcal{F}_i)$, $I'(v', p)(\mathcal{F}_1(v'), \dots, \mathcal{F}_n(v')) = 1$.

Then, the following lemma can be easily shown:

Lemma 7. Let $\mathcal{F}_1, \dots, \mathcal{F}_n \in D''$. Then, the following statements hold.

- (i) $\bigcap_{i=1, \dots, n} \text{dom}(\mathcal{F}_i)$ is an upward-closed subset of W' .
- (ii) For any $w' \in W'$, $w' \dashv \bigcap_{i=1, \dots, n} \text{dom}(\mathcal{F}_i)$.

The following lemma ensures that the values of formulas in $\mathcal{K}'$ are preserved in $\mathcal{K}''$ in some sense. Before describing the lemma, we establish the λ -notation. Let V be a set of individual variables and $\mathfrak{E}(x)$ an expression of our meta-language that denotes some value for each $x \in V$. Then $\lambda x \in V. \mathfrak{E}(x)$ denotes the function whose domain is V and whose value at each argument x is $\mathfrak{E}(x)$.

Lemma 8. Let $\alpha \in \text{FOFml}(\mathcal{C})$. Then, the following conditions are equivalent:

- (i) $\|\alpha\|_{\mathcal{K}'', w'}^{\rho''} = 1$.
- (ii) For any $v' \succeq' w'$ such that $v' \in \bigcap_{x \in \text{FV}(\alpha)} \text{dom}(\rho''(x))$, $\|\alpha\|_{\mathcal{K}', v'}^{\lambda x \in \text{FV}(\alpha). \rho''(x)(v')} = 1$.

Note that if $w' \in \bigcap_{x \in \text{FV}(\alpha)} \text{dom}(\rho''(x))$, then (ii) is equivalent to $\|\alpha\|_{\mathcal{X}', w'}^{\lambda x \in \text{FV}(\alpha). \rho''(x)(w')} = 1$.

Proof. The proof proceeds by induction on α . Since the other cases can be proved in the same way as in [1], we only prove the claim in the case where α is of the form $c(\beta_1, \dots, \beta_{\text{ar}(c)})$. (The full version [6] of the present paper contains the whole proof.) Suppose $\alpha \equiv c(\beta_1, \dots, \beta_{\text{ar}(c)})$. Put $\vec{\beta} = \beta_1, \dots, \beta_{\text{ar}(c)}$ and $n = \text{ar}(c)$.

(i) $\Rightarrow$ (ii): Suppose (i) holds. Let $v' \succeq' w'$ and $v' \in \bigcap_{x \in \text{FV}(\alpha)} \text{dom}(\rho''(x))$.

In order to show $\|c(\vec{\beta})\|_{\mathcal{X}', v'}^{\lambda x \in \text{FV}(\alpha). \rho''(x)(v')} = 1$, we take an arbitrary $u' \succeq' v'$, and show that $t_c(\|\vec{\beta}\|_{\mathcal{X}', u'}^{\lambda x \in \text{FV}(\alpha). \rho''(x)(u')}) = 1$. For each $1 \leq i \leq n$, by the induction hypothesis, we have $\|\beta_i\|_{\mathcal{X}'', u'}^{\rho''} = \|\beta_i\|_{\mathcal{X}', u'}^{\lambda x \in \text{FV}(\beta_i). \rho''(x)(u')} = \|\beta_i\|_{\mathcal{X}', u'}^{\lambda x \in \text{FV}(\beta). \rho''(x)(u')}$. Thus, we have $t_c(\|\vec{\beta}\|_{\mathcal{X}'', u'}^{\rho''}) = t_c(\|\vec{\beta}\|_{\mathcal{X}', u'}^{\lambda x \in \text{FV}(\alpha). \rho''(x)(u')})$, the left hand side of which equals to 1 by the supposition (i).

(ii) $\Rightarrow$ (i): Suppose (ii) holds. We show that $\|c(\vec{\beta})\|_{\mathcal{X}'', w'}^{\rho''} = 1$. In order to prove this, we suppose $v' \succeq' w'$ and show that $t_c(\|\vec{\beta}\|_{\mathcal{X}'', v'}^{\rho''}) = 1$. First, we consider the case $\|\vec{\beta}\|_{\mathcal{X}'', v'}^{\rho''} = 1$. Then, since $v' \dashv \bigcap_{x \in \text{FV}(\alpha)} \text{dom}(\rho''(x))$, there is some $u' \succeq' v'$ such that $u' \in \bigcap_{x \in \text{FV}(\alpha)} \text{dom}(\rho''(x))$. For any $1 \leq i \leq n$, by the hereditary property we have $\|\beta_i\|_{\mathcal{X}'', u'}^{\rho''} \geq \|\beta_i\|_{\mathcal{X}'', v'}^{\rho''} = 1$, and hence, by the induction hypothesis, we have $\|\beta_i\|_{\mathcal{X}', u'}^{\lambda x \in \text{FV}(\beta_i). \rho''(x)(u')} = 1$. Thus, we have $\|\vec{\beta}\|_{\mathcal{X}'', v'}^{\rho''} = 1 = \|\vec{\beta}\|_{\mathcal{X}', u'}^{\lambda x \in \text{FV}(\alpha). \rho''(x)(u')}$. On the other hand, since $\|\alpha\|_{\mathcal{X}', u'}^{\lambda x \in \text{FV}(\alpha). \rho''(x)(u')} = 1$ holds by assumption (ii), we have $t_c(\|\vec{\beta}\|_{\mathcal{X}', u'}^{\lambda x \in \text{FV}(\alpha). \rho''(x)(u')}) = 1$. Combining these results, we have $t_c(\|\vec{\beta}\|_{\mathcal{X}'', v'}^{\rho''}) = 1$.

Secondly, we consider the case $\|\vec{\beta}\|_{\mathcal{X}'', v'}^{\rho''} \neq 1$. Put $\mathcal{I} = \{1 \leq i \leq n \mid \|\beta_i\|_{\mathcal{X}'', v'}^{\rho''} = 0\}$ and $\mathcal{J} = \{1 \leq i \leq n \mid \|\beta_i\|_{\mathcal{X}'', v'}^{\rho''} = 1\}$. Note that $\mathcal{I} \neq \emptyset$. Now, it suffices to show that for each $i \in \mathcal{I}$, there exists a $t'_i \in W'$ such that $t'_i \in \bigcap_{x \in \text{FV}(\alpha)} \text{dom}(\rho''(x))$; $\|\beta_i\|_{\mathcal{X}', t'_i}^{\lambda x \in \text{FV}(\alpha). \rho''(x)(t'_i)} = 0$; $\|\beta_j\|_{\mathcal{X}', t'_i}^{\lambda x \in \text{FV}(\alpha). \rho''(x)(t'_i)} = 1$ for all $j \in \mathcal{J}$; and $t_c(\|\vec{\beta}\|_{\mathcal{X}', t'_i}^{\lambda x \in \text{FV}(\alpha). \rho''(x)(t'_i)}) = 1$. This is because, for such t'_i 's, if we take $\{\mathbf{a}_1, \dots, \mathbf{a}_n\}$ in Lemma 4 to be the set $\{\|\vec{\beta}\|_{\mathcal{X}', t'_i}^{\lambda x \in \text{FV}(\alpha). \rho''(x)} \mid i \in \mathcal{I}\}$, then it holds that

$$\mathbf{a}_1 \sqcap \dots \sqcap \mathbf{a}_n = \bigcap_{i \in \mathcal{I}} \|\vec{\beta}\|_{\mathcal{X}', t'_i}^{\lambda x \in \text{FV}(\alpha). \rho''(x)} = \|\vec{\beta}\|_{\mathcal{X}'', v'}^{\rho''},$$

and hence, $t_c(\|\vec{\beta}\|_{\mathcal{X}'', v'}^{\rho''}) = 1$ follows from Lemma 4. So, we fix an arbitrary $i \in \mathcal{I}$ and show that such $t'_i \in W'$ exists. By the induction hypothesis for β_i , there is some $u' \succeq' v'$ such that $u' \in \bigcap_{x \in \text{FV}(\beta_i)} \text{dom}(\rho''(x))$ and

$\|\beta_i\|_{\mathcal{K}', u'}^{\lambda x \in \text{FV}(\beta_i) \cdot \rho''(x)(u')} = 0$. By the property (II) of $\mathcal{K}'$, $u' \not\vdash \{r' \succeq' u' \mid \|\beta_i\|_{\mathcal{K}', r'}^{\lambda x \in \text{FV}(\beta_i) \cdot \rho''(x)(u')} = 1\}$ holds. Hence, there is some path $\mathcal{P}$ from u' such that $\mathcal{P} \cap \{r' \succeq' u' \mid \|\beta_i\|_{\mathcal{K}', r'}^{\lambda x \in \text{FV}(\beta_i) \cdot \rho''(x)(u')} = 1\} = \emptyset$. On the other hand, since $u' \vdash \bigcap_{x \in \text{FV}(\alpha)} \text{dom}(\rho''(x))$, $\mathcal{P}$ and $\bigcap_{x \in \text{FV}(\alpha)} \rho''(x)$ intersect at some point, say, $t'_i \in W'$. Then, we have $t'_i \succeq' u' \succeq' v'$, $t'_i \in \bigcap_{x \in \text{FV}(\alpha)} \text{dom}(\rho''(x))$ and

$$\|\beta_i\|_{\mathcal{K}', t'_i}^{\lambda x \in \text{FV}(\alpha) \cdot \rho''(x)(t'_i)} = \|\beta_i\|_{\mathcal{K}', t'_i}^{\lambda x \in \text{FV}(\alpha) \cdot \rho''(x)(u')} = 0.$$

By the hereditary property, we have $\|\beta_j\|_{\mathcal{K}'', t'_i}^{\rho''} \geq \|\beta_j\|_{\mathcal{K}'', v'}^{\rho''} = 1$ for every $j \in \mathcal{J}$. Hence, by the induction hypothesis we have $\|\beta_j\|_{\mathcal{K}', t'_i}^{\lambda x \in \text{FV}(\alpha) \cdot \rho''(x)(t'_i)} = 1$ for every $j \in \mathcal{J}$. Finally, since $\|c(\vec{\beta})\|_{\mathcal{K}', t'_i}^{\lambda x \in \text{FV}(\alpha) \cdot \rho''(x)(t'_i)} = 1$ holds by the assumption (ii), we have $\text{tc}(\|\vec{\beta}\|_{\mathcal{K}', t'_i}^{\lambda x \in \text{FV}(\alpha) \cdot \rho''(x)(t'_i)}) = 1$. Thus, we have proved that t'_i satisfies the desired conditions. $\square$

From Lemma 8, it follows that $\|\Gamma \Rightarrow \Delta\|_{\mathcal{K}'', w'_*}^{\rho''} = \|\Gamma \Rightarrow \Delta\|_{\mathcal{K}', w'_*}^{\rho'_*} = 0$, where ρ'' is the assignment in D'' such that for each free variables x in Γ and Δ , $\rho''(x)$ is the function on W' whose value is constantly $\rho_*(x) \in D'(w'_*)$. Thus, we have finished the proof of the “if” part of Theorem 3.

3.2 The “only if” Part

Here, we show the “only if” part of Theorem 3 by showing its contraposition:

Proposition 9. *Suppose there is a non-supermultiplicative $c \in \mathcal{C}$. Then, it holds that $\text{FOILS}(\mathcal{C}) \neq \text{FOCDS}(\mathcal{C})$.*

First, let us consider the case in which $\text{ar}(c) \leq 2$. Since, as mentioned in Sect. 2.2, non-supermultiplicative connectives whose arity is less than or equal to 2 are only $\vee$ (disjunction) and $\oplus$ (exclusive disjunction), we only have to consider the cases $c = \vee$ and $c = \oplus$. Regarding disjunction, it is known that the sequents of the form $\forall x(p(x) \vee q(x)) \Rightarrow \forall xp(x) \vee \exists xq(x)$ are CD-valid but not Kripke-valid (cf., e.g., [7]). As to exclusive disjunction, we can verify that the corresponding sequents of the form $\forall x(p(x) \oplus q(x)) \Rightarrow \forall xp(x) \oplus \exists xq(x)$ are also CD-valid but not Kripke-valid (cf. case (A) in the proof below)⁹. For c of general arity, we construct a sequent in $\text{FOCDS}(\mathcal{C}) \setminus \text{FOILS}(\mathcal{C})$ which plays the same role as $\forall x(p(x) \vee q(x)) \Rightarrow \forall xp(x) \vee \exists xq(x)$ and $\forall x(p(x) \oplus q(x)) \Rightarrow \forall xp(x) \oplus \exists xq(x)$. This construction requires an elaborate case analysis.

⁹ It is well-known that the formal system of the logic of constant domains is obtained by adding axiom schema $\forall x(\alpha \vee \beta) \Rightarrow \forall x\alpha \vee \beta$, where $x \notin \text{FV}(\beta)$, to intuitionistic predicate calculus. This schema is so-called **D**-axiom. In particular, **D**-axiom is CD-valid and not Kripke-valid. In contrast to both $\forall x(p(x) \vee q(x)) \Rightarrow \forall xp(x) \vee \exists xq(x)$ and $\forall x(p(x) \oplus q(x)) \Rightarrow \forall xp(x) \oplus \exists xq(x)$ being CD-valid and not Kripke-valid, we can verify that the sequents corresponding to **D**-axioms, $\forall x(p(x) \oplus r) \Rightarrow \forall xp(x) \oplus r$, where r is a 0-ary predicate symbol, are Kripke-valid.

Proof. Suppose there is a non-supermultiplicative connective $c \in \mathcal{C}$. Then, there are $\mathbf{a}, \mathbf{b} \in \{0, 1\}^{\text{ar}(c)}$ such that $t_c(\mathbf{a}) = t_c(\mathbf{b}) = 1$ and $t_c(\mathbf{a} \sqcap \mathbf{b}) = 0$. Let p and q be distinct unary predicate symbols. Fix two 0-ary predicate symbols T and R , which shall play particular roles later.

First, we define a Kripke model $\mathcal{K}^* = \langle W^*, \preceq^*, D^*, I^* \rangle$, which is used to show the constructed sequents are not Kripke-valid.

- $W^* = \{w_1, w_2\}$;
- $w_i \preceq^* w_j$ if and only if $i \leq j$;
- $D^*(w_1) = \{a_1\}$, $D^*(w_2) = \{a_1, a_2\}$;
- $\bullet I(w_1, p)(a_1) = 1, I(w_1, q)(a_1) = 0, I(w_1, T) = 1, I(w_1, R) = 0$;
- $\bullet I(w_2, p)(a_1) = 1, I(w_2, q)(a_1) = 0, I(w_2, T) = 1, I(w_2, R) = 0$,
- $I(w_2, p)(a_2) = 0, I(w_2, q)(a_2) = 1$.

We define $\mathbf{a}^*, \mathbf{b}^* \in \{0, 1\}^{\text{ar}(c)}$ as follows:

$$\mathbf{a}^*[i] = \begin{cases} \mathbf{a}[i] & \text{if } \mathbf{a}[i] = 1 \text{ or } \mathbf{b}[i] = 1 \\ 1 & \text{if } \mathbf{a}[i] = 0 \text{ and } \mathbf{b}[i] = 0, \end{cases}$$

$$\mathbf{b}^*[i] = \begin{cases} \mathbf{b}[i] & \text{if } \mathbf{a}[i] = 1 \text{ or } \mathbf{b}[i] = 1 \\ 1 & \text{if } \mathbf{a}[i] = 0 \text{ and } \mathbf{b}[i] = 0. \end{cases}$$

Then, we have $\mathbf{a} \sqcap \mathbf{b}^* = \mathbf{a}^* \sqcap \mathbf{b} = \mathbf{a} \sqcap \mathbf{b}$ and $\mathbf{a} \sqcup \mathbf{b}^* = \mathbf{a}^* \sqcup \mathbf{b} = \mathbf{a}^* \sqcup \mathbf{b}^* = \mathbf{1}$. We divide into five cases: (A) $t_c(\mathbf{1}) = 0$; (B) $t_c(\mathbf{a}^*) = 1$; (C) $t_c(\mathbf{b}^*) = 1$; (D) $t_c(\mathbf{a}^*) = t_c(\mathbf{b}^*) = t_c(\mathbf{a}^* \sqcap \mathbf{b}^*) = 0$ and $t_c(\mathbf{1}) = 1$; (E) $t_c(\mathbf{a}^*) = t_c(\mathbf{b}^*) = 0$ and $t_c(\mathbf{a}^* \sqcap \mathbf{b}^*) = t_c(\mathbf{1}) = 1$.

CASE (A): $t_c(\mathbf{1}) = 0$. Define $F \in \text{FOFml}(\mathcal{C})$ by $F \equiv c(T, \dots, T)$. Note that, for any Kripke model $\mathcal{K} = \langle W, \preceq, D, I \rangle$ and any $w \in W$, if $\|T\|_{\mathcal{K}, w}^{\emptyset} = 1$ then $\|F\|_{\mathcal{K}, v}^{\emptyset} = 0$ for all $v \succeq w$.

We define formulas $\varphi_1, \dots, \varphi_{\text{ar}(c)}, \varphi, \psi_1, \dots, \psi_{\text{ar}(c)}, \psi \in \text{FOFml}(\mathcal{C})$ as follows:

$$\varphi_i \equiv \begin{cases} F & \text{if } \mathbf{a}[i] = 0 \text{ and } \mathbf{b}[i] = 0 \\ p(x) & \text{if } \mathbf{a}[i] = 0 \text{ and } \mathbf{b}[i] = 1 \\ q(x) & \text{if } \mathbf{a}[i] = 1 \text{ and } \mathbf{b}[i] = 0 \\ T & \text{if } \mathbf{a}[i] = 1 \text{ and } \mathbf{b}[i] = 1, \end{cases}$$

$$\varphi \equiv \forall x c(\varphi_1, \dots, \varphi_{\text{ar}(c)}),$$

$$\psi_i \equiv \begin{cases} F & \text{if } \mathbf{a}[i] = 0 \text{ and } \mathbf{b}[i] = 0 \\ \forall x p(x) & \text{if } \mathbf{a}[i] = 0 \text{ and } \mathbf{b}[i] = 1 \\ \exists x q(x) & \text{if } \mathbf{a}[i] = 1 \text{ and } \mathbf{b}[i] = 0 \\ T & \text{if } \mathbf{a}[i] = 1 \text{ and } \mathbf{b}[i] = 1, \end{cases}$$

$$\psi \equiv c(\psi_1, \dots, \psi_{\text{ar}(c)}).$$

Put $\vec{\varphi} = \varphi_1, \dots, \varphi_{\text{ar}(c)}$ and $\vec{\psi} = \psi_1, \dots, \psi_{\text{ar}(c)}$. We show $T, \varphi \Rightarrow \psi \in \text{FOCDS}(\mathcal{C}) \setminus \text{FOILS}(\mathcal{C})$.

First, we show $T, \varphi \Rightarrow \psi \in \text{FOCDS}(\mathcal{C})$. Let $\mathcal{K} = \langle W, \preceq, D, I \rangle$ be a constant domain Kripke model and $w \in W$. We suppose $\|T\|_{\mathcal{K}, w}^{\emptyset} = \|\varphi\|_{\mathcal{K}, w}^{\emptyset} = 1$, in order to show $\|\psi\|_{\mathcal{K}, w}^{\emptyset} = 1$. To show $\|\psi\|_{\mathcal{K}, w}^{\emptyset} = 1$, it suffices to show $t_c(\|\vec{\psi}\|_{\mathcal{K}, v}^{\emptyset}) = 1$ for any $v \succeq w$. Let $v \succeq w$. Then, we can see $\|\forall xp(x)\|_{\mathcal{K}, v}^{\emptyset} = 1$ or $\|\exists xq(x)\|_{\mathcal{K}, v}^{\emptyset} = 1$ holds. For otherwise there exists some $a \in D$ such that $\|p(x)\|_{\mathcal{K}, v}^{\emptyset[x \mapsto a]} = \|q(x)\|_{\mathcal{K}, v}^{\emptyset[x \mapsto a]} = 0$, and hence, $\|\vec{\varphi}\|_{\mathcal{K}, v}^{\emptyset[x \mapsto a]} = \mathbf{a} \sqcap \mathbf{b}$, and thus, $t_c(\|\vec{\varphi}\|_{\mathcal{K}, v}^{\emptyset[x \mapsto a]}) = 0$, which contradicts $\|\varphi\|_{\mathcal{K}, w}^{\emptyset} = 1$. First, we consider the case $\|\forall xp(x)\|_{\mathcal{K}, v}^{\emptyset} = \|\exists xq(x)\|_{\mathcal{K}, v}^{\emptyset} = 1$. Then we have $\|\vec{\psi}\|_{\mathcal{K}, v}^{\emptyset} = \mathbf{a} \sqcup \mathbf{b}$ and there exists some $a \in D$ such that $\|p(x)\|_{\mathcal{K}, v}^{\emptyset[x \mapsto a]} = \|q(x)\|_{\mathcal{K}, v}^{\emptyset[x \mapsto a]} = 1$, so that $\|\vec{\varphi}\|_{\mathcal{K}, v}^{\emptyset[x \mapsto a]} = \mathbf{a} \sqcup \mathbf{b}$. Hence, we have $t_c(\|\vec{\psi}\|_{\mathcal{K}, v}^{\emptyset}) = t_c(\mathbf{a} \sqcup \mathbf{b}) = t_c(\|\vec{\varphi}\|_{\mathcal{K}, v}^{\emptyset[x \mapsto a]})$ the right hand side of which equals to 1 by $\|\varphi\|_{\mathcal{K}, w}^{\emptyset} = 1$. Thus, $t_c(\|\vec{\psi}\|_{\mathcal{K}, v}^{\emptyset}) = 1$. Next, we consider the case that one of $\|\forall xp(x)\|_{\mathcal{K}, v}^{\emptyset}$ and $\|\exists xq(x)\|_{\mathcal{K}, v}^{\emptyset}$ is 1 and the other is 0. Then, either $\|\vec{\psi}\|_{\mathcal{K}, v}^{\emptyset} = \mathbf{a}$ or $\|\vec{\psi}\|_{\mathcal{K}, v}^{\emptyset} = \mathbf{b}$, and hence, we have $t_c(\|\vec{\psi}\|_{\mathcal{K}, v}^{\emptyset}) = 1$.

Secondly, we show $T, \varphi \Rightarrow \psi \notin \text{FOILS}(\mathcal{C})$. In order to do so, we verify $\|T, \varphi \Rightarrow \psi\|_{\mathcal{K}^*, w_1}^{\emptyset} = 0$. First, we can easily see the following: $\|\vec{\varphi}\|_{\mathcal{K}^*, w_1}^{\emptyset[x \mapsto a_1]} = \|\vec{\varphi}\|_{\mathcal{K}^*, w_2}^{\emptyset[x \mapsto a_1]} = \mathbf{b}$; $\|\vec{\varphi}\|_{\mathcal{K}^*, w_2}^{\emptyset[x \mapsto a_2]} = \mathbf{a}$; $\|\forall xp(x)\|_{\mathcal{K}^*, w_1}^{\emptyset} = \|\exists xq(x)\|_{\mathcal{K}^*, w_1}^{\emptyset} = 0$; and $\|\vec{\psi}\|_{\mathcal{K}^*, w_1}^{\emptyset} = \mathbf{a} \sqcap \mathbf{b}$. From these it follows that $\|\varphi\|_{\mathcal{K}^*, w_1}^{\emptyset} = 1$ and $\|\psi\|_{\mathcal{K}^*, w_1}^{\emptyset} = 0$.

CASE (B): $t_c(\mathbf{a}^*) = 1$. Note that, in this case, there is no i such that $\mathbf{a}^*[i] = \mathbf{b}[i] = 0$. We define formulas $\varphi_1, \dots, \varphi_{\text{ar}(c)}, \varphi, \psi_1, \dots, \psi_{\text{ar}(c)}, \psi \in \text{FOFml}(\mathcal{C})$ as follows:

$$\begin{aligned} \varphi_i &\equiv \begin{cases} p(x) & \text{if } \mathbf{a}^*[i] = 0 \text{ and } \mathbf{b}[i] = 1 \\ q(x) & \text{if } \mathbf{a}^*[i] = 1 \text{ and } \mathbf{b}[i] = 0 \\ T & \text{if } \mathbf{a}^*[i] = 1 \text{ and } \mathbf{b}^*[i] = 1 \end{cases} \\ \varphi &\equiv \forall xc(\varphi_1, \dots, \varphi_{\text{ar}(c)}) \\ \psi_i &\equiv \begin{cases} \forall xp(x) & \text{if } \mathbf{a}^*[i] = 0 \text{ and } \mathbf{b}[i] = 1 \\ \exists xq(x) & \text{if } \mathbf{a}^*[i] = 1 \text{ and } \mathbf{b}[i] = 0 \\ T & \text{if } \mathbf{a}^*[i] = 1 \text{ and } \mathbf{b}^*[i] = 1 \end{cases} \\ \psi &\equiv c(\psi_1, \dots, \psi_{\text{ar}(c)}) \end{aligned}$$

Put $\vec{\varphi} = \varphi_1, \dots, \varphi_{\text{ar}(c)}$ and $\vec{\psi} = \psi_1, \dots, \psi_{\text{ar}(c)}$. We show $T, \varphi \Rightarrow \psi \in \text{FOCDS}(\mathcal{C}) \setminus \text{FOILS}(\mathcal{C})$.

First, $T, \varphi \Rightarrow \psi \in \text{FOCDS}(\mathcal{C})$ can be proved similarly to case (A); in fact, if we replace every $\mathbf{a}$ in the proof in case (A) by $\mathbf{a}^*$ (and thus, $\mathbf{a} \sqcap \mathbf{b}$ by $\mathbf{a}^* \sqcap \mathbf{b}$ and $\mathbf{a} \sqcup \mathbf{b}$ by $\mathbf{a}^* \sqcup \mathbf{b}$), then we obtain a proof for case (B).

$T, \varphi \Rightarrow \psi \notin \text{FOILS}(\mathcal{C})$ can also be proved similarly to case (A); in fact, if we replace every $\mathbf{a}$ in the proof in (A) by $\mathbf{a}^*$ (and thus, $\mathbf{a} \sqcap \mathbf{b}$ by $\mathbf{a}^* \sqcap \mathbf{b}$), then we obtain a proof for case (B).

CASE (C): $t_c(\mathbf{b}^*) = 1$. This case can be shown similarly to case (B).

CASE (D): $t_c(\mathbf{a}^*) = t_c(\mathbf{b}^*) = t_c(\mathbf{a}^* \sqcap \mathbf{b}^*) = 0$ and $t_c(\mathbf{1}) = 1$. We define formulas $\varphi_1, \dots, \varphi_{\text{ar}(c)}, \varphi, \psi_1, \dots, \psi_{\text{ar}(c)}, \psi \in \text{FOFml}(\mathcal{C})$ as follows:

$$\varphi_i \equiv \begin{cases} R & \text{if } \mathbf{a}[i] = 0 \text{ and } \mathbf{b}[i] = 0 \\ p(x) & \text{if } \mathbf{a}[i] = 0 \text{ and } \mathbf{b}[i] = 1 \\ q(x) & \text{if } \mathbf{a}[i] = 1 \text{ and } \mathbf{b}[i] = 0 \\ T & \text{if } \mathbf{a}[i] = 1 \text{ and } \mathbf{b}[i] = 1 \end{cases}$$

$$\varphi \equiv \forall xc(\varphi_1, \dots, \varphi_{\text{ar}(c)})$$

$$\psi_i \equiv \begin{cases} R & \text{if } \mathbf{a}[i] = 0 \text{ and } \mathbf{b}[i] = 0 \\ \forall xp(x) & \text{if } \mathbf{a}[i] = 0 \text{ and } \mathbf{b}[i] = 1 \\ \exists xq(x) & \text{if } \mathbf{a}[i] = 1 \text{ and } \mathbf{b}[i] = 0 \\ T & \text{if } \mathbf{a}[i] = 1 \text{ and } \mathbf{b}[i] = 1 \end{cases}$$

$$\psi \equiv c(\psi_1, \dots, \psi_{\text{ar}(c)})$$

Put $\vec{\varphi} = \varphi_1, \dots, \varphi_{\text{ar}(c)}$ and $\vec{\psi} = \psi_1, \dots, \psi_{\text{ar}(c)}$. We show $T, \varphi \Rightarrow \psi \in \text{FOCDS}(\mathcal{C}) \setminus \text{FOILS}(\mathcal{C})$.

First, we show $T, \varphi \Rightarrow \psi \in \text{FOCDS}(\mathcal{C})$. Let $\mathcal{K} = \langle W, \preceq, D, I \rangle$ be a constant domain Kripke model and $w \in W$. We suppose $\|T\|_{\mathcal{K}, w}^{\varnothing} = \|\varphi\|_{\mathcal{K}, w}^{\varnothing} = 1$, in order to show $\|\psi\|_{\mathcal{K}, w}^{\varnothing} = 1$. To show $\|\psi\|_{\mathcal{K}, w}^{\varnothing} = 1$, it suffices to show $t_c(\|\vec{\psi}\|_{\mathcal{K}, v}^{\varnothing}) = 1$ for any $v \succeq w$. Let $v \succeq w$. Then, we divide into two subcases according to the value of $\|R\|_{\mathcal{K}, v}^{\varnothing}$.

SUBCASE (i): $\|R\|_{\mathcal{K}, v}^{\varnothing} = 0$. In this case, $t_c(\|\vec{\psi}\|_{\mathcal{K}, v}^{\varnothing}) = 1$ can be shown similarly to case (A) because R plays the same role as F in case (A).

SUBCASE (ii): $\|R\|_{\mathcal{K}, v}^{\varnothing} = 1$. In this case, $\|\forall xp(x)\|_{\mathcal{K}, v}^{\varnothing} = \|\exists xq(x)\|_{\mathcal{K}, v}^{\varnothing} = 1$ holds. For, otherwise, there is some $a \in D$ such that $\|\vec{\varphi}\|_{\mathcal{K}, v}^{\varnothing[x \mapsto a]}$ equals to either $\mathbf{a}^*$, $\mathbf{b}^*$ or $\mathbf{a}^* \sqcap \mathbf{b}^*$, and hence, $t_c(\|\vec{\varphi}\|_{\mathcal{K}, v}^{\varnothing[x \mapsto a]}) = 0$, which contradicts $\|\varphi\|_{\mathcal{K}, w}^{\varnothing} = 1$. Thus, we have $\|\vec{\psi}\|_{\mathcal{K}, v}^{\varnothing} = \mathbf{1}$. Hence, $t_c(\|\vec{\psi}\|_{\mathcal{K}, v}^{\varnothing}) = 1$.

Secondly, we show $T, \varphi \Rightarrow \psi \notin \text{FOILS}(\mathcal{C})$. Since $\|R\|_{\mathcal{K}^*, w_1}^{\varnothing} = \|R\|_{\mathcal{K}^*, w_2}^{\varnothing} = 0$, R plays the same role as F in case (A), and hence $T, \varphi \Rightarrow \psi \notin \text{FOILS}(\mathcal{C})$ can be shown similarly to case (A).

CASE (E): $t_c(\mathbf{a}^*) = t_c(\mathbf{b}^*) = 0$ and $t_c(\mathbf{a}^* \sqcap \mathbf{b}^*) = t_c(\mathbf{1}) = 1$. Note that, in this case, there is no i such that $\mathbf{a}^*[i] = \mathbf{b}^*[i] = 0$. First, in order to construct the desired sequent, we define biconditional $\leftrightarrow$ ($t_{\leftrightarrow}(x, y) = 1$ if and only if $x = y$) using c . That is, for any $\alpha \in \text{FOFml}(\mathcal{C})$ and any $\beta \in \text{FOFml}(\mathcal{C})$, we define a formula $\alpha \leftrightarrow c \beta \in \text{FOFml}(\mathcal{C})$. First, we define $\theta_1^{\alpha, \beta}, \dots, \theta_{\text{ar}(c)}^{\alpha, \beta}$ for

$\alpha, \beta \in \text{FOFml}(\mathcal{C})$ by

$$\theta_i^{\alpha, \beta} \equiv \begin{cases} \alpha & \text{if } \mathbf{a}^*[i] = 0, \mathbf{b}^*[i] = 1 \\ \beta & \text{if } \mathbf{a}^*[i] = 1, \mathbf{b}^*[i] = 0 \\ T & \text{if } \mathbf{a}^*[i] = 1, \mathbf{b}^*[i] = 1. \end{cases}$$

Put $\overrightarrow{\theta^{\alpha, \beta}} \equiv \theta_1^{\alpha, \beta}, \dots, \theta_{\text{ar}(c)}^{\alpha, \beta}$. We define $\alpha \leftrightarrow_c \beta$ by $\alpha \leftrightarrow_c \beta \equiv c(\theta_1^{\alpha, \beta}, \dots, \theta_{\text{ar}(c)}^{\alpha, \beta})$. Then, $\leftrightarrow_c$ has the same meaning as biconditional $\leftrightarrow$ whenever the value of T is interpreted as 1, that is, for any Kripke model $\mathcal{K} = \langle W, \preceq, D, I \rangle$, any $w \in W$, and any assignment ρ in $D(w)$, if $\|T\|_{\mathcal{K}, w}^\rho = 1$, then $\|\alpha \leftrightarrow_c \beta\|_{\mathcal{K}, w}^\rho = 1$ holds if and only if $\|\alpha\|_{\mathcal{K}, v}^\rho = \|\beta\|_{\mathcal{K}, v}^\rho$ holds for every $v \succeq w$. In order to show the “if” part, let $\mathcal{K} = \langle W, \preceq, D, I \rangle$ be a Kripke model and $w \in W$, and suppose $\|T\|_{\mathcal{K}, w}^\rho = 1$ and $\|\alpha\|_{\mathcal{K}, v}^\rho = \|\beta\|_{\mathcal{K}, v}^\rho$ for all $v \succeq w$. Then, for any $v \succeq w$, $\|\overrightarrow{\theta^{\alpha, \beta}}\|_{\mathcal{K}, v}^\rho$ is either $\mathbf{a}^* \sqcap \mathbf{b}^*$ or $\mathbf{1}$, and hence, $t_c(\|\overrightarrow{\theta^{\alpha, \beta}}\|_{\mathcal{K}, v}^\rho) = 1$ for any $v \succeq w$. Thus, we have $\|\alpha \leftrightarrow_c \beta\|_{\mathcal{K}, w}^\rho = 1$. In order to show the (contraposition of) “only if” part, let $\mathcal{K} = \langle W, \preceq, D, I \rangle$ be a Kripke model and $w \in W$, and suppose $\|T\|_{\mathcal{K}, w}^\rho = 1$ and $\|\alpha\|_{\mathcal{K}, v}^\rho \neq \|\beta\|_{\mathcal{K}, v}^\rho$ for some $v \succeq w$. Then, $\|\overrightarrow{\theta^{\alpha, \beta}}\|_{\mathcal{K}, v}^\rho$ is either $\mathbf{a}^*$ or $\mathbf{b}^*$, and hence, we have $t_c(\|\overrightarrow{\theta^{\alpha, \beta}}\|_{\mathcal{K}, v}^\rho) = 0$. Thus, we have $\|\alpha \leftrightarrow_c \beta\|_{\mathcal{K}, w}^\rho = 0$.

Now, we define $\varphi_1, \dots, \varphi_{\text{ar}(c)}, \varphi, \psi_1, \dots, \psi_{\text{ar}(c)}, \psi \in \text{FOFml}(\mathcal{C})$ as in case (D). Put $\vec{\varphi} = \varphi_1, \dots, \varphi_{\text{ar}(c)}$ and $\vec{\psi} = \psi_1, \dots, \psi_{\text{ar}(c)}$. We show $T, R \leftrightarrow_c \forall xp(x), R \leftrightarrow_c \exists xq(x), \varphi \Rightarrow \psi \in \text{FOCDS}(\mathcal{C}) \setminus \text{FOILS}(\mathcal{C})$.

First, we show $T, R \leftrightarrow_c \forall xp(x), R \leftrightarrow_c \exists xq(x), \varphi \Rightarrow \psi \in \text{FOCDS}(\mathcal{C})$. Let $\mathcal{K} = \langle W, \preceq, D, I \rangle$ be a constant domain Kripke and $w \in W$. We suppose $\|T\|_{\mathcal{K}, w}^\varnothing = \|R \leftrightarrow_c \forall xp(x)\|_{\mathcal{K}, w}^\varnothing = \|R \leftrightarrow_c \exists xq(x)\|_{\mathcal{K}, w}^\varnothing = \|\varphi\|_{\mathcal{K}, w}^\varnothing = 1$, in order to show $\|\psi\|_{\mathcal{K}, w}^\varnothing = 1$. To show $\|\psi\|_{\mathcal{K}, w}^\varnothing = 1$, it suffices to show $t_c(\|\vec{\psi}\|_{\mathcal{K}, v}^\varnothing) = 1$ for any $v \succeq w$. Let $v \succeq w$. Then, we can see $\|R\|_{\mathcal{K}, v}^\varnothing = 1$ holds. For if $\|R\|_{\mathcal{K}, v}^\varnothing = 0$, then we would have either $\|\forall xp(x)\|_{\mathcal{K}, v}^\varnothing = 1$ or $\|\exists xq(x)\|_{\mathcal{K}, v}^\varnothing = 1$ similarly to case (A), whereas $\|R\|_{\mathcal{K}, v}^\varnothing = 0$ and $\|R \leftrightarrow_c \forall xp(x)\|_{\mathcal{K}, w}^\varnothing = \|R \leftrightarrow_c \exists xq(x)\|_{\mathcal{K}, v}^\varnothing = \|T\|_{\mathcal{K}, w}^\varnothing = 1$ would imply $\|\forall xp(x)\|_{\mathcal{K}, v}^\varnothing = \|\exists xq(x)\|_{\mathcal{K}, v}^\varnothing = 0$. Now, from $\|R\|_{\mathcal{K}, v}^\varnothing = 1$, $\|T\|_{\mathcal{K}, w}^\varnothing = 1$ and $\|R \leftrightarrow_c \forall xp(x)\|_{\mathcal{K}, w}^\varnothing = \|R \leftrightarrow_c \exists xq(x)\|_{\mathcal{K}, w}^\varnothing = 1$, it follows that $\|\forall xp(x)\|_{\mathcal{K}, v}^\varnothing = \|\exists xq(x)\|_{\mathcal{K}, v}^\varnothing = 1$. Thus, we have $\|\vec{\psi}\|_{\mathcal{K}, v}^\varnothing = \mathbf{1}$. Hence, we have $t_c(\|\vec{\psi}\|_{\mathcal{K}, v}^\varnothing) = 1$.

Secondly, $T, R \leftrightarrow_c \forall xp(x), R \leftrightarrow_c \exists xq(x), \varphi \Rightarrow \psi \notin \text{FOILS}(\mathcal{C})$ can be shown similarly to case (D).

4 Conclusion

We have extended generalized Kripke semantics to first-order logic. By considering general propositional connectives, we have clarified what property of connectives causes the difference between intuitionistic first-order logic and the logic of

constant domains in terms of validity of sequents. Furthermore, as mentioned in Sect. 1.2, we have also found out what property of connectives causes the difference between the logic of constant domains and classical logic and the difference between intuitionistic logic and classical logic. The results are summarized as follows:

Theorem. *Let $\mathcal{C}$ be a set of propositional connectives. Let $\text{FOCLS}(\mathcal{C})$ be the set of classically valid predicate sequents built out of connectives in $\mathcal{C}$, $\text{FOILS}(\mathcal{C})$, the set of Kripke-valid predicate sequents built out of connectives in $\mathcal{C}$, and $\text{FOCDS}(\mathcal{C})$, the set of CD-valid predicate sequents built out of connectives in $\mathcal{C}$. Then, the following statements hold.*

- $\text{FOILS}(\mathcal{C}) = \text{FOCDS}(\mathcal{C})$ if and only if every connective in $\mathcal{C}$ is supermultiplicative.
- $\text{FOCDS}(\mathcal{C}) = \text{FOCLS}(\mathcal{C})$ if and only if every connective in $\mathcal{C}$ is monotonic.
- $\text{FOILS}(\mathcal{C}) = \text{FOCLS}(\mathcal{C})$ if and only if every connective in $\mathcal{C}$ is both monotonic and supermultiplicative.

References

1. Gabbay, D.M.: Semantical Investigations in Heyting's Intuitionistic Logic. Synthese Library, Springer, Netherlands (1981)
2. Geuvers, H., Hurkens, T.: Deriving natural deduction rules from truth tables. In: Ghosh, S., Prasad, S. (eds.) ICLA 2017. LNCS, vol. 10119, pp. 123–138. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54069-5_10
3. Kawano, T., Matsuda, N., Takagi, K.: Effect of the choice of connectives on the relation between classical logic and intuitionistic logic. Notre Dame J. Formal Logic **63**(2), 243–259 (2022)
4. Kripke, S.A.: Semantical analysis of intuitionistic logic I. Formal Syst. Recursive Functions **40**, 92–130 (1965)
5. Matsuda, N., Takagi, K.: Effect of the choice of connectives on the relation between the logic of constant domains and classical predicate logic (2021). [arXiv:2107.03972](https://arxiv.org/abs/2107.03972)
6. Matsuda, N., Takagi, K.: What kinds of connectives cause the difference between intuitionistic predicate logic and the logic of constant domains? (2022). [arXiv:2205.05995](https://arxiv.org/abs/2205.05995)
7. Nagashima, T.: An intermediate predicate logic. Hitotsubashi J. Arts Sci. **14**(1), 53–58 (1973)
8. Rouseau, G.: Sequents in many valued logic II. Fundam. Math. **67**(1), 125–131 (1970)



Logic of Visibility in Social Networks

Rustam Galimullin, Mina Young Pedersen^(✉), and Marija Slavkovik

Department of Information Science and Media Studies,
University of Bergen, Bergen, Norway
{rustam.galimullin,mina.pedersen,marija.slavkovik}@uib.no

Abstract. Social media is not a neutral channel for news consumption. How visible information posted online is, depends on many factors such as the network structure, the emotional volatility of the content and the design of the social media platform. In this paper, we use formal methods to study the visibility of agents and information in a social network. We introduce a modal logic to reason about a social network of agents that can follow each other, post and share information. We show that by imposing some simple rules on the system, a potentially malicious agent can take advantage of the network construction to post an unpopular opinion that may reach many agents. The network is presented both in a static and dynamic form. We prove completeness, expressivity and model checking problem complexity results for the corresponding logical systems.

Keywords: Social networks · Visibility · Modal logic · Reachability

1 Introduction

An overwhelming majority of people across the globe consume news on social media [1]. Social media, however, is not a neutral channel. How visible information posted online is, and how many users in a social network it can reach, depends on many factors. These include the network structure [16], the emotional volatility of the content [8], past exposure to similar information [21] and the design and recommendation algorithms of the particular social media platform [20].

This paper contributes to the study of social networks and the measurable impact social media platforms have on their users. Social networks have been studied using numerous methods in numerous disciplines. Formal logic methods for representing and reasoning about social networks have been used to analyze opinion diffusion and social influence [9–11, 19], social bots [22], group polarization [24, 25], gatekeepers [5], echo chambers [23] and informational cascades [3], among other phenomena. Our work is positioned within this literature.

We are here concerned with the problem of applying **formal methods to study the visibility of agents and information in a social network**. In addition to having structural properties, number of agents and how they are connected, a social network also has other properties connected to the visibility

of an agent, such as: which interests and opinions the agents have, what they are communicating and how the network changes through time. It is our position that logic based methods are needed to complement empirical methods to reach a full understanding of these properties of social networks.

The notion of visibility is rooted in the idea of being seen. One of our main motivations is to present an analysis of visibility that captures a complex view of what it means to be visible in a social network, one that extends merely counting the followers of an agent. To do this, we introduce a modal logic for representing agents, their opinions and interactions in a social network.

Our social network consists of a set of agents and two sets of relations between them: one represents followers, the other represents posts that pass through the network. We turn to Fig. 1 for an intuitive explanation of the network.

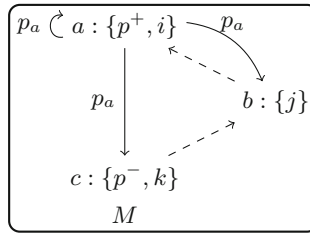


Fig. 1. Model M with the followership relation depicted by dashed arrows.

The network M consists of three agents a, b and c . Dashed arrows represent a followership relation: c follows b and b follows a . The situation concerns a post on a particular topic, called p . Agent a is in favour of, *pro*, p , denoted p^+ , whereas c is *contra* p , denoted p^- . Agent b has no opinion about p . Furthermore, agent a has posted on p , represented by a reflexive loop denoted p_a and agents b and c have seen the post, denoted by p_a -arrows from a .

The intuition behind our models is to observe a situation of posting and sharing a post, after it has happened. Posting, sharing, following and unfollowing adhere to some simple rules of the system:

1. When an agent posts, all her followers can see the post.
2. If an agent sees a post on a topic she likes, she will reshare the post and follow the original poster.
3. If an agent sees a post on a topic she dislikes, she does not reshare it and unfollows the agent from whom she has seen the post.
4. If an agent sees a post on a topic she is indifferent to, she does not do anything.

These rules are an oversimplification of a real-life network, but we believe they capture some key notions of a social network which we can use to analyze situations that may occur in an actual network setting. Knowing the rules of the system, we can return to M in Fig. 1 and observe that c likely has unfollowed a after a posted on p .

We first present a logic that specifies a static network, as seen in Fig. 1. The purpose of this logic is not to define what visibility is, but to allow us to discuss different qualitative and quantitative measures of visibility and formalise some of them in the logic. Next, we extend the framework into a dynamic setting where we step-wise observe what happens when information is posted in the network. We show that according to the rules of the system, the interests of the agents' followers matter a lot to what information is shared and seen. We also show that a malicious agent could take advantage of the network construction to post an unpopular opinion that will reach many agents. We believe these observations can be useful to understand how agents in a network contribute to spreading controversial information such as misinformation.

We are also interested in the mathematical properties of the system itself. We give formulas corresponding to the rules of the system, and claim that the logic is complete with respect to the models with these rules. The model checking problem for the static logic is proved to be in P . In the dynamic extension, we show that the language with the dynamic modality is strictly more expressive than the static language without. We also prove that the model checking problem for the dynamic logic is $PSPACE$ -complete.

The contribution of the paper is the following:

- We introduce a novel logic to analyze posting and sharing information in a social network, and prove mathematical results about this formal system.
- We propose quantitative and qualitative measures of visibility and reachability, and formalize some of the properties as logical formulas.
- We use our formal system to reason about mechanisms that might occur in real-life online social networks, specifically we formalize how a potentially malicious agent could take advantage of the network construction to post a controversial opinion that will reach many agents.

The paper is structured as follows. In Sect. 2, we give an overview of the related work in social network analysis on reachability and visibility. In Sect. 3, we present static visibility logic (SVL). We specify some mathematical properties of the logic, propose measures of visibility and give some corresponding logical formulas. In Sect. 4, we extend SVL with a dynamic operator and name it visibility logic (VL). We give a motivating example where we show that one can exploit the network structure to expose more agents to a controversial opinion. In Sect. 5, we summarize our paper and outline directions for future work.

2 Related Work: Visibility and Reachability

Visibility in social networks is yet to be explicitly explored from a formal logical perspective. The concept has however been researched in the social network analysis literature. We present a selected collection of this work to learn how this related field has attempted to measure visibility. There seem to be no consensus in the literature of what it means to be visible in a social network, which motivates the usefulness of a further study on this topic.

Closely related to visibility in social networks is the notion of reachability. What exactly reachability is, or how closely related it is to visibility, is not agreed upon. This is illustrated by the different measures seen in this section. Visibility and reachability are presented as properties of both networks, agents and posts, when relevant we specify which in the following.

[13], known to be part of the canonical literature in social network analysis, describe the reachability properties of a network in terms of identifying which agents are reachable from which others through connected paths of edges.

[28] distinguishes reachability and visibility in an online social network, where the first measure is dependent on the second. The network is represented as an undirected graph where nodes represent agents and the relation between them represent one of three non-overlapping relations: trusted friends, acquaintances or distrusted agents. Agents can post information with four different visibility settings: trusted friends, trusted friends and acquaintances, all friends and public. The visibility of an agent is therefore measured with respect to what relation the viewers of the post has to the agent that posts.

The reachability factor of a post is defined in terms of a function: $d(v_1, v_2) = \frac{|e(v_1, v_2)|}{\sqrt{|v_1|} \times \sqrt{|v_2|}}$. In this function, v_1 is the set of agents in the network that have seen the post and v_2 is the set of agents that have not seen the post. $e(v_1, v_2)$ is the set of relations between agents across v_1 and v_2 specified with respect to the relations in the network graph. The reachability factor is dependent on the visibility settings of the agent who posts; the set v_1 increases and v_2 decreases when the visibility settings include a higher number of agents.

[29] presents a temporal characterisation of reachability. In this work, the reachability is measured between two given nodes in a time interval in the network. The network is presented as a series of undirected graphs which represent how a network changes through time. The nodes in the network can be regarded as agents and the relation between them as information channels. Node j is reachable in the time interval $[t_{min}, t_{max}]$ from node i if a message can be delivered through the information channel in that time interval.

[27] defines two types of visibility of an agent in an online social network: topological and behavioral visibility. Although it is mentioned that this could be a generic social network, the examples refer to the microblogging network Twitter, which is represented as a directed graph of agents who can post and follow each other. In the model presented in [27], a tweet embodies at least one topic from a set of interests S . Each agent in the network also has some specified interests from S . Topological visibility of an agent is calculated based on the number of followers of the agent and the clustering coefficient of the network. The clustering coefficient is usually defined in the literature in terms of directed graphs, and is meant to give a view of the network structure. The higher the number, the more highly connected the network is. It is not specified which definition of clustering coefficient is used in [27]. The behavioral visibility of an agent is defined as the average of the visibility of all the tweets that are shared by the user in a time interval Δt . The visibility of a tweet represents the number

of users influenced by the tweet, and is proportional to the number of followers whose interests matches the topics of the tweet.

3 Reasoning About Visibility in a Static Setting

In this section we present some of the main concepts underlying our intuitions about visibility in social networks.

3.1 Language and Semantics of SVL

Let $\text{Nom} = \{i, j, k, \dots\}$ be a countable set of nominals, and $\text{Top} = \{p, q, r, \dots\}$ be a countable set of topics, such that $\text{Nom} \cap \text{Top} = \emptyset$.

Definition 1 (Syntax). *We define the well-formed formulas of the language of the static fragment of visibility logic SVL to be generated by the following grammar:*

$$\varphi ::= p^+ \mid p^- \mid i \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \Diamond_{i:p}\varphi \mid \Diamond_{i:p}^{-1}\varphi \mid \blacklozenge\varphi \mid \blacklozenge^{-1}\varphi \mid @_i\varphi$$

where $p \in \text{Top}$ and $i \in \text{Nom}$. We define propositional connectives like $\vee, \rightarrow$ and the formulas $\top, \perp$ as usual and the duals as standard $\Box := \neg\Diamond\neg$, $\Box^{-1} := \neg\Diamond^{-1}\neg$, $\blacksquare := \neg\blacklozenge\neg$, and $\blacksquare^{-1} := \neg\blacklozenge^{-1}\neg$.

In our language, similar to other approaches to logic-based analysis of social networks (see, e.g., [10]), we distinguish three possible dispositions of an agent to a topic $p \in \text{Top}$. The agent may be *pro* p , which we express with p^+ , *contra* p , expressed by p^- , or *indifferent* to p , if the agent is neither *pro* nor *contra* p .

Constructs $\blacklozenge\varphi$ and $\blacklozenge^{-1}\varphi$ express that ‘the current agent follows an agent satisfying φ ’ and ‘the current agent is followed by an agent that satisfies φ ’ respectively. Formulas $\Diamond_{i:p}\varphi$ and $\Diamond_{i:p}^{-1}\varphi$ mean that ‘there is an agent satisfying φ who sees the (re)post of agent i on topic p ’ and ‘there is an agent satisfying φ whose (re)post on topic p (originally posted by agent i) is seen by the current one’.

Formulas of SVL are defined on relational visibility models.

Definition 2. *A visibility model (or a model) M is a tuple $(A, F, +, -, V, R)$, where*

- A is a non-empty set of agents;
- $F : A \rightarrow 2^A$ is an irreflexive followership relation,
- $+$: $A \rightarrow 2^{\text{Top}}$ assigns to each agent a set of topics she is *pro*,
- $-$: $A \rightarrow 2^{\text{Top}}$ assigns to each agent a set of topics she is *contra* such that for all agents a it holds that $+(a) \cap -(a) = \emptyset$,
- $V : \text{Nom} \rightarrow 2^A$ is a valuation such that for all $i \in \text{Nom}$: $|V(i)| = 1$,
- $R : \text{Top} \times A \rightarrow 2^{A \times A}$ is a visibility relation for each topic and each agent satisfying the following conditions, where $p \in \text{Top}$ and $c \in A$:
 1. If $(a, b) \in R(p, c)$, then $(a, a) \in R(p, c)$.

2. If $(a, a) \in R(p, c)$, then $(a, b) \in R(p, c)$ for all b such that $b \in F(a)$.
3. If $(a, b) \in R(p, c)$, $p \in +(b)$, and $b \neq c$, then $(b, b) \in R(p, c)$ and $b \in F(c)$.
4. If $(a, b) \in R(p, c)$, $p \in -(b)$, and $a \neq b$, then $(b, b) \notin R(p, c)$ and $b \notin F(a)$.
5. If $(a, b) \in R(p, c)$, $p \notin +(b)$, $p \notin -(b)$, and $a \neq b$, then $(b, b) \notin R(p, c)$.

A pointed visibility model M_a is a model M with a distinguished point $a \in A$ where evaluation takes place. If necessary, we refer to the elements of the tuple as $A_M, F_M, +_M, -_M, V_M$, and R_M . A visibility model such that for all $a \in A$ there is some $i \in \text{Nom}$ such that $V(i) = \{a\}$ is called named. All models we will be dealing with in the paper are named. A visibility model with no restrictions on F or R is called minimal.

In the definition above, the first condition on R states that if agent b sees a post, which was originally posted by agent c on topic p , from agent a , then a herself can see the post. The second condition ensures that if an agent posts a post, all her followers can see the post. Condition number three specifies that if an agent sees a post on the topic she likes, she will reshare the post and follow the original poster. The fourth condition says that if an agent sees a post on the topic she dislikes, she does not reshare it and unfollows the agent from whom she has seen the post. Finally, the last condition stipulates that if an agent sees a post on a topic she is indifferent to, she does not reshare the post.

Note that our definition of R does not preclude situations where agents may have seen a post on the topic they dislike from an agent they do not follow. How such situation may come about will be the focus of the next section.

Definition 3 (Semantics). Let $M = (A, F, +, -, V, R)$ be a model, $a, b, c \in A$, $p \in \text{Top}$, $i \in \text{Nom}$, and $\varphi, \psi \in \text{SVL}$. The semantics of SVL is recursively defined as follows:

$$\begin{aligned}
 M_a \models p^+ & \quad \text{iff } p \in +(a) \\
 M_a \models p^- & \quad \text{iff } p \in -(a) \\
 M_a \models i & \quad \text{iff } a \in V(i) \\
 M_a \models \neg\varphi & \quad \text{iff } M_a \not\models \varphi \\
 M_a \models \varphi \wedge \psi & \quad \text{iff } M_a \models \varphi \text{ and } M_a \models \psi \\
 M_a \models \Diamond_{i:p}\varphi & \quad \text{iff } \exists b, c \in A : (a, b) \in R(p, c) \text{ and } V(i) = \{c\} \text{ and } M_b \models \varphi \\
 M_a \models \Diamond_{i:p}^{-1}\varphi & \quad \text{iff } \exists b, c \in A : (b, a) \in R(p, c) \text{ and } V(i) = \{c\} \text{ and } M_b \models \varphi \\
 M_a \models \blacklozenge\varphi & \quad \text{iff } \exists b \in A : a \in F(b) \text{ and } M_b \models \varphi \\
 M_a \models \blacklozenge^{-1}\varphi & \quad \text{iff } \exists b \in A : b \in F(a) \text{ and } M_b \models \varphi \\
 M_a \models @_i\varphi & \quad \text{iff } M_b \models \varphi \text{ and } \{b\} = V(i)
 \end{aligned}$$

Observe that if $M_a \not\models p^+$ then we have that either $p \in -(a)$ or not. This corresponds to the intuition that agent a is not *pro* p if she actively dislikes the topic (she is *contra* p), or she is indifferent to it. Similarly, for $M_a \not\models p^-$.

Recall the example from Fig. 1. In the figure, we have that $M_a \models p^+$, $M_c \models p^-$, and $M_b \models \neg p^+ \wedge \neg p^-$, meaning that agent a is *pro* topic p , agent b is indifferent towards the topic, and c is *contra* p . Moreover, we have, for example, that $M_c \models \Diamond_{i:p}^{-1}\top \wedge \blacksquare\neg p^+$, meaning that agent c has seen a post by the agent with name i on topic p , and that all agents that c follows are not *pro* p .

Definition 4. A formula φ is called *valid* if for all models M_a , we have that $M_a \models \varphi$. Formulas φ and ψ are *equivalent*, if for all models M_a , it holds that $M_a \models \varphi$ if and only if $M_a \models \psi$.

Definition 5. Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two languages. We say that $\mathcal{L}_2$ is *more expressive* than $\mathcal{L}_1$ if for each $\varphi \in \mathcal{L}_1$ there is an equivalent $\psi \in \mathcal{L}_2$, and there is a $\chi \in \mathcal{L}_2$ for which there is no equivalent $\tau \in \mathcal{L}_1$.

The following notion of bisimulation is based on hybrid bisimulation [2] and on bisimulation for logics with ‘backwards-looking’ modalities (see, e.g., [18]).

Definition 6. Let $M = (A_M, F_M, +_M, -_M, V_M, R_M)$ and $N = (A_N, F_N, +_N, -_N, V_N, R_N)$ be *visibility models*, and $Q \subseteq \text{Nom}$. We say that M and N are Q -bisimilar (denoted $M \sqsubseteq_Q N$) if there is a non-empty relation $B \subseteq A_M \times A_N$, called Q -bisimulation such that the following conditions are satisfied:

- Atoms⁺** If $B(a, b)$, then for all $p \in \text{Top}$: $p \in +_M(a)$ iff $p \in +_N(b)$,
- Atoms⁻** If $B(a, b)$, then for all $p \in \text{Top}$: $p \in -_M(a)$ iff $p \in -_N(b)$,
- Nominals 1** If $B(a, b)$, then for all $i \in Q$: $a \in V_M(i)$ iff $b \in V_N(i)$,
- Nominals 2** For all $i \in Q$, if $V_M(i) = \{a\}$ and $V_N(i) = \{b\}$, then $B(a, b)$,
- Forth $\Diamond$** If $B(a, b)$ and $(a, a') \in R_M(p, c)$, then there is a $b' \in A_N$ such that $(b, b') \in R_N(p, c)$ and $B(a', b')$,
- Back $\Diamond$** If $B(a, b)$ and $(b, b') \in R_N(p, c)$, then there is an $a' \in A_M$ such that $(a, a') \in R_M(p, c)$ and $B(a', b')$,
- Forth $\Diamond^{-1}$** If $B(a, b)$ and $(a', a) \in R_M(p, c)$, then there is a $b' \in A_N$ such that $(b, b') \in R_N(p, c)$ and $B(a', b')$,
- Back $\Diamond^{-1}$** If $B(a, b)$ and $(b', b) \in R_N(p, c)$, then there is an $a' \in A_M$ such that $(a', a) \in R_M(p, c)$ and $B(a', b')$,
- Forth $\blacklozenge$, Back $\blacklozenge$, Forth $\blacklozenge^{-1}$, Back $\blacklozenge^{-1}$** Similar to the cases of $\Diamond$ and $\Diamond^{-1}$ substituting R for F taking care of arguments.

We say that M_a and N_b are Q -bisimilar and denote this by $M_a \sqsubseteq_Q N_b$ if there is a bisimulation linking agents a and b .

The following theorem is a standard result in modal logic [2].

Theorem 1. Let M_a and N_b be two models. If $M_a \sqsubseteq_Q N_b$, then for all $\varphi \in \text{SVL}$ such that φ includes only nominals from Q , $M_a \models \varphi$ if and only if $N_b \models \varphi$.

3.2 Soundness, Completeness and Model Checking of SVL

To argue that SVL is complete, we first present an axiomatisation for *the minimal version* of SVL. Recall that a visibility model is called *minimal*, if there are no restrictions on relations F or R .

Definition 7. The proof system of the minimal version of SVL, **MSVL**, comprises axioms and rules of inference of basic hybrid logic $\mathcal{H}(@)$ and additional axioms and rules of inference dealing with the converse relation.

(A0) <i>Propositional tautologies</i> (A1) $\Box_{i:p}(\varphi \rightarrow \psi) \rightarrow (\Box_{i:p}\varphi \rightarrow \Box_{i:p}\psi)$ (A2) $\Box_{i:p}^{-1}(\varphi \rightarrow \psi) \rightarrow (\Box_{i:p}^{-1}\varphi \rightarrow \Box_{i:p}^{-1}\psi)$ (A3) $\blacksquare(\varphi \rightarrow \psi) \rightarrow (\blacksquare\varphi \rightarrow \blacksquare\psi)$ (A4) $\blacksquare^{-1}(\varphi \rightarrow \psi) \rightarrow (\blacksquare^{-1}\varphi \rightarrow \blacksquare^{-1}\psi)$ (A5) $@_i(\varphi \rightarrow \psi) \rightarrow (@_i\varphi \rightarrow @_i\psi)$ (A6) $@_i\varphi \leftrightarrow \neg @_i\neg\varphi$ (A7) $@_ii$ (A8) $@_i@_j\varphi \leftrightarrow @_j\varphi$ (A9) $\Diamond_{i:p}@_i\varphi \rightarrow @_i\varphi$ (A10) $\Diamond_{i:p}^{-1}@_i\varphi \rightarrow @_i\varphi$ (A11) $\blacklozenge @_i\varphi \rightarrow @_i\varphi$ (A12) $\blacklozenge^{-1}@_i\varphi \rightarrow @_i\varphi$	(A13) $@_i\Box_{i:p}\Diamond_{i:p}^{-1}i$ (A14) $@_i\Box_{i:p}^{-1}\Diamond_{i:p}i$ (A15) $@_i\blacksquare\blacklozenge^{-1}i$ (A16) $@_i\blacksquare^{-1}\blacklozenge i$ (A17) $p^+ \wedge p^- \rightarrow \perp$ (R0) <i>From $\varphi \rightarrow \psi$ and φ, infer ψ</i> (R1) <i>From φ, infer $\Box_{i:p}\varphi$</i> (R2) <i>From φ, infer $\Box_{i:p}^{-1}\varphi$</i> (R3) <i>From φ, infer $\blacksquare\varphi$</i> (R4) <i>From φ, infer $\blacksquare^{-1}\varphi$</i> (R5) <i>From φ, infer $@_i\varphi$</i> (R6) <i>From φ, infer φ^σ</i> (R7) <i>From $@_i\varphi$ with $i \notin \varphi$, infer φ</i>

In the proof system, σ is a substitution that uniformly replaces nominals by nominals, and topics with formulas.

The axiomatisation of the minimal version of SVL is essentially an axiomatisation of hybrid tense logic [6] with additional axioms and rules of inference for followership. In particular, axioms (A13)–(A16) capture the interaction between modalities and their converses. Axiom (A17) specifies that agents cannot be both *pro* and *contra* the same topic. The completeness of the axiomatisation can be shown via a standard canonical model construction for hybrid logics (see, e.g., [7, Sect. 7.3])

Theorem 2. *MSVL is sound and complete with respect to the class of minimal visibility models.*

To get the axiomatisation of full SVL, we extend the axiomatisation from Definition 7 with an axiom for irreflexivity of followership $@_i\neg\blacklozenge i$ and five axioms for conditions on the visibility relation. We call the resulting proof system **SVL**.

Proposition 1. *The following formulas capture the conditions on the visibility relation (from 1 to 5 in Definition 2):*

1. $@_i(\Diamond_{j:p}\top \rightarrow \Diamond_{j:p}i)$
2. $@_i(\Diamond_{j:p}i \rightarrow \blacksquare^{-1}\Diamond_{j:p}^{-1}i)$
3. $@_i((p^+ \wedge \Diamond_{j:p}^{-1}\top \wedge \neg j) \rightarrow (\Diamond_{j:p}i \wedge \blacklozenge j))$
4. $@_i((p^- \wedge \Diamond_{j:p}^{-1}(\neg i \wedge k)) \rightarrow (\neg\Diamond_{j:p}\top \wedge \neg\blacklozenge k))$
5. $@_i((\neg p^- \wedge \neg p^+ \wedge \Diamond_{j:p}^{-1}\neg i) \rightarrow \neg\Diamond_{j:p}\top)$

We believe that completeness of **SVL** can be shown by the canonical model construction [7, Sect. 7.3], where our model will additionally have two different valuation functions, one for $+$ and one for $-$, and two types of relations for F and R . That the model satisfies conditions 1–5 from Definition 2 is guaranteed by axioms from Proposition 1 and the irreflexivity of followership axiom. We omit full details due to the lack of space.

Theorem 3. *SVL is complete with respect to visibility models.*

We also mention here that the complexity of the model checking problem of SVL is in P . This result follows trivially from the fact that model checking hybrid tense logic with universal modality is in P [15].

Theorem 4. *Model checking SVL is in P .*

3.3 Expressing Visibility

We look at measures of visibility and reachability, and formalize some of them as formulas in SVL. Let $M = (A, F, +, -, V, R)$ be a model. Some quantitative amounts related to visibility that we can count in finite models are:

- How many followers the agent called i has: $|\{a \in A \mid M_a \models \blacklozenge i\}|$
- How many agents have seen the agent called i 's post on p :
 $|\{a \in A \mid M_a \models \Diamond_{i:p}^{-1} \top\}|$
- How many agents that are pro p have seen the agent called i 's post on p :
 $|\{a \in A \mid M_a \models p^+ \wedge \Diamond_{i:p}^{-1} \top\}|$

We also present some formulas corresponding to qualitative properties of agents in the network. The following formulas are forced at an agent iff the property holds of that agent:

- The current agent i is the original poster of a post on p : $i \wedge \Diamond_{i:p} \top$
- The current agent has seen i 's post on p : $\Diamond_{i:p}^{-1} \top$
- All the followers of the current agent i have shared i 's post on p :
 $i \wedge \blacksquare^{-1} \Diamond_{i:p} \top$
- The current agent i shared a post to a follower j , but j also saw the post from another source: $i \wedge \blacklozenge^{-1} (j \wedge \Diamond_{i:p}^{-1} i \wedge \Diamond_{i:p}^{-1} (\neg i \wedge \neg j))$
- The current agent i has gained a follower who is pro p , after i posted on p :
 $i \wedge \Diamond_{i:p} \top \wedge \blacklozenge^{-1} (p^+ \wedge \Diamond_{i:p}^{-1} i)$
- The current agent i has reached the agent j with i 's post on p in no more than 3 steps: $i \wedge \Diamond_{i:p} \Diamond_{i:p} \Diamond_{i:p} j$

4 Visibility Logic

To reason about the effects of agents posting on various topics, we introduce a dynamic extension of SVL that we call *visibility logic* (VL). Compared to SVL, VL is enriched with dynamic operators $[\pi]\varphi$, where π is an action of the current agent making a post. While defining VL, we follow *dynamic epistemic logics* (DELs) [12], and in particular *action model logic* [4, 12]. We begin with a motivating example.

4.1 Example: Taking the Advantage to Be Seen by Many

In some networks, the best tactic for exposing more agents to a controversial opinion is to first post on a popular topic. Consider the follower-network M in Fig. 2. For simplicity, we do not include nominals in the Figure. This network consists of 6 agents named in alphabetical order from a to f . Agent a has two followers b and c . Agent b has three followers d, e and f . We assume that agents d, e and f might have some followers that we do not have information about, noted in the figure with dots. Furthermore, agent a is positive in favor of vaccination (abbreviated v in the figure) which is a controversial topic amongst the agents: all agents have an opinion about vaccines and three of the agents a, c and f are *pro* vaccination, whereas b, d and e are *contra* vaccination. The topic of dogs (abbreviated d) on the other hand, is widely liked. All agents like dogs, except a who is indifferent: $d \notin +(a)$ and $d \notin -(a)$.

Imagine that agent a wants to post on vaccines, and want as many as possible of the other agents in the network to see the post. We show that the best tactic for agent a is to first post on dogs, even though a is indifferent about dogs, and then later post on vaccines. Consider first the scenario in Fig. 2 where agent a posts v from the initial outset. An update happens in two steps. First, we add visibility arrows corresponding to posting and resharing. In the second step, we update the followership relation based on whether the agents who has seen a post are *pro* or *contra* the post. The resulting update is $M^{a:v}$ in Fig. 2, where agents b and c has seen a 's vaccine-post and only c remains as a 's follower.

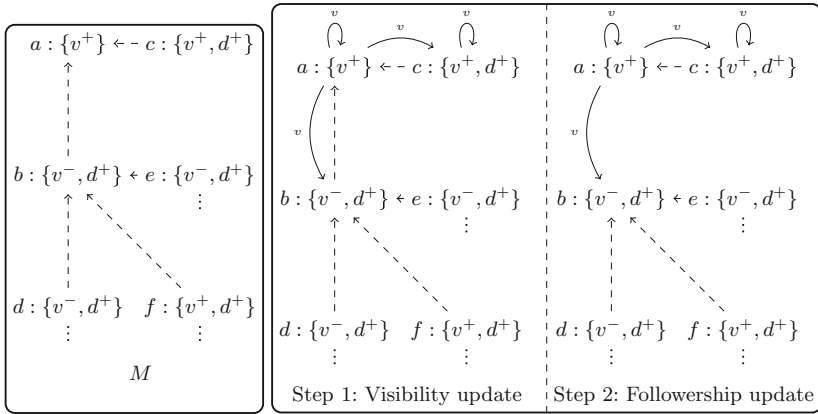


Fig. 2. A follower-network M (left) where vaccination is a controversial topic, and update $M^{a:v}$ (right) after agent a posts in favor of vaccines.

Then, consider instead the situation where agent a posts on dogs in the update $M^{a:d}$ before posting on vaccines in the update $M^{a:d,a:v}$, both in Fig. 3. Note that to make the situation easier to read, we omit the followership arrows in the visibility update and the visibility arrows in the followership update in

both figures. After $M^{a:d}$, all agents have seen agent a 's post on dogs. Since they like dogs, all agents also follow a after the update.

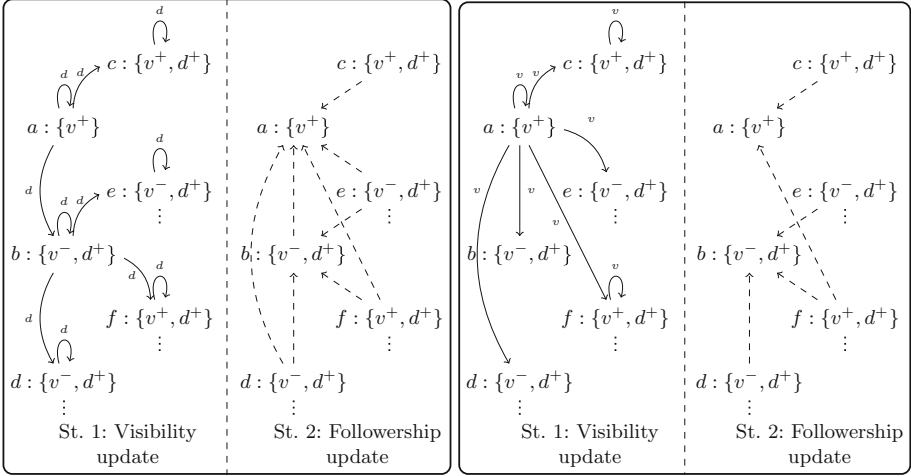


Fig. 3. Updates $M^{a:d}$ (left) and $M^{a:d,a:v}$ (right) after agent a posts on dogs before posting on vaccination.

In $M^{a:d,a:v}$, we see the results after agent a first posted on dogs, and then vaccines. All the agents have now seen a 's vaccine-post. Most of them did not like it and unfollowed a , but only after they were exposed to the post. Interestingly, we also notice that agent f , who was not originally a follower of a in the initial network outset, now follows a and has shared the vaccine post to their followers.

There are two tactical reasons for agent a to post on dogs before their more controversial post on vaccines. Firstly, a larger portion of the agents now saw a 's post on vaccines since they followed agent a after the dog-post. Secondly, a has been able to reach out and expand their network: agent f who is also pro vaccines, has shared the vaccination post to their, for us unknown, followers.

The reason behind a phenomenon such as this is directly connected to an underlying notion of trust between agents in the network. In our setting, agents follow other agents when the former is exposed to content that they like by the latter. In the example, we imagine agents likely followed a because they wanted to see more dog-friendly content. Agent a misuses the trust of their followers by pretending to be interested in dogs before posting on vaccination.

What becomes clear in this example, is that in our simplified setting of posting and sharing in a social network, the interests of an agents' followers matter a lot to what information is shared and seen. Secondly, the system is vulnerable to exploitation by a potentially malicious agent: there are opportunities to tactically post on popular topics to later expose more agents to a controversial opinion. To reason about dynamic situations such as these, we introduce VL.

4.2 Language, Semantics, and Logical Properties of VL

The language of VL is an extension of the language of SVL.

Definition 8 (Syntax). The language of visibility logic $\mathcal{VL}$ is defined recursively by the following BNF:

$$\begin{aligned}\varphi &::= p^+ \mid p^- \mid i \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \Diamond_{i:p}\varphi \mid \Diamond_{i:p}^{-1}\varphi \mid \blacklozenge\varphi \mid \blacklozenge^{-1}\varphi \mid @_i\varphi \mid [\pi]\varphi \\ \pi &::= p \mid (\pi \cup \pi)\end{aligned}$$

where $[\pi]\varphi$ is read ‘after the current agent executes action π , φ holds’.

Union of actions $(\pi \cup \tau)$ was inherited by DELs from propositional dynamic logic [14], and in the context of visibility formulas, $[p \cup q]\varphi$ mean ‘whichever topic the current agent posts on, p or q , φ will be true (in both cases)’.

Definition 9 (Semantics). Let $M = (A, F, +, -, V, R)$ be a visibility model, $a \in A$, and $p, q \in \text{Top}$. The semantics of VL is the same as in Definition 3 with the following additions:

$$\begin{aligned}M_a &\models [p]\varphi \quad \text{iff } M_a^{a:p} \models \varphi \\ M_a &\models [\pi \cup \tau]\varphi \quad \text{iff } M_a \models [\pi]\varphi \text{ and } M_a \models [\tau]\varphi\end{aligned}$$

where $M_a^{a:p}$ is defined in the following two steps. First, let $M^* = (A, F, +, -, V, R^*)$, where $R^*(p, a)$ is the least fixed point of function $f : 2^A \rightarrow 2^A$ defined as

$$\begin{aligned}f(X) &= X \cup \{(a, a)\} \cup \{(b, c) \mid (b, b) \in X \text{ and } c \in F(b)\} \cup \\ &\quad \cup \{(c, c) \mid p \in +(c) \text{ and } \exists b : (b, c) \in X\}.\end{aligned}$$

Informally, intermediate model M^* differs from M only in R in such a way that R^* now contains the fact that a has posted on p , that her post has reached all her followers, and that all followers who are *pro* p reshare the post further to their followers. Secondly, we construct $M^{a:p}$ out of M^* by updating F :

1. $F^{a:p}(a) = F(a) \cup \{b\}$, if $a \neq b$, $p \in +(b)$, and $\exists c : (c, b) \in R^*(p, a)$,
2. $F^{a:p}(b) = F(b) \setminus \{c\}$, if $p \in -(b)$ and $(c, b) \in R^*(p, a)$.

Intuitively, agent b will follow the original poster a if she has seen the post, maybe not even from a , and if she is *pro* the topic. Agent c will stop following anyone from whom she has seen a post on a topic she dislikes.

To give a taste of VL, let us provide some properties that are valid or not valid on visibility models. All the validities can be shown by an application of the definition of the semantics.

Proposition 2 Let $p, q \in \text{Top}$ and $\varphi \in \mathcal{VL}$.

1. $\neg[p]\varphi \leftrightarrow [p]\neg\varphi$ is valid.
2. $[\pi \cup \tau]\varphi \leftrightarrow [\pi]\varphi \wedge [\tau]\varphi$ is valid.
3. $\Diamond_{i:p}^{-1}\varphi \leftrightarrow [q]\Diamond_{i:p}^{-1}\varphi$ is valid.
4. $[p]\varphi \rightarrow [p][p]\varphi$ is not valid.
5. $[p][p]\varphi \rightarrow [p]\varphi$ is not valid.

The first formula states that the operator of posting on a topic is its own dual. The second property shows how to eliminate non-deterministic choice. The third item claims that once an agent has seen a post of agent with name i on topic p , no further post can revoke this. The fact that formulas four and five are not valid indicates that consecutive posting on the same topic yields different results. A counterexample showing this would include the current agent with name i posting on p and gaining new followers. Additional post on the same topic by the same agent will add new p -arrows to those new followers thus resulting in a different updated model that is not guaranteed to satisfy φ .

4.3 Expressivity and Model Checking

Now, we state that VL is more expressive than its static fragment SVL. This result is quite interesting, since many of DELs, for example public announcement logic [26], arrow update logic [17], and action model logic [12, Chapter 6], are equally expressive as the static logic they are built upon. Those expressivity results are usually obtained with the use of so-called reduction axioms that allow one to equivalently rewrite formulas of dynamic extensions to formulas of the static fragment. Thus, the fact that VL is more expressive than SVL also entails that no reduction axioms for VL are possible.

Theorem 5 $\mathcal{SVL} < \mathcal{VL}$.

Proof Consider a $\mathcal{VL}$ formula $[p]\blacklozenge^{-1}\blacksquare^{-1}\neg$, and assume towards a contradiction that there is an equivalent formula ψ of $\mathcal{SVL}$. Since ψ has a finite size $n = |\psi|$, there is a set of nominals $Q = \{j_1, \dots, j_{n+1}\}$ that are not present in ψ .

Consider models M and N in Fig. 4. The models are chains of length $n + 2$ that start with agent a and with each next agent following the previous one. The only difference between the models is that the last agent in the chain in model M is *pro* topic p , and the last agent in the chain in model N is neither *pro* nor *contra* topic p .

Now we will argue that $[p]\blacklozenge^{-1}\blacksquare^{-1}\perp$ distinguishes M_a and N_a . In particular, $M_a \models [p]\blacklozenge^{-1}\blacksquare^{-1}\perp$ and $N_a \not\models [p]\blacklozenge^{-1}\blacksquare^{-1}\perp$. Indeed, agent a posting on topic p results in the updated visibility model $M_a^{a:p}$ presented in the figure. In the updated model, it holds that $M_{b_{n+1}}^{a:p} \models \blacklozenge i$, i.e. that agent b_{n+1} follows agent a . Moreover, agent b_{n+1} does not have any followers, so $M_{b_{n+1}}^{a:p} \models \blacksquare^{-1}\perp$ is vacuously true. Hence, $M_a \models [p]\blacklozenge^{-1}\blacksquare^{-1}\perp$. To see that $N_a \not\models [p]\blacklozenge^{-1}\blacksquare^{-1}\perp$, it is enough to notice that agent b_{n+1} is not *pro* topic p , and thus they do not follow agent a in the updated model.

In order to show that $M_a \models \psi$ if and only if $N_a \models \psi$, we informally argue that M_a and N_a are $n - Q$ -bisimilar. Let us first recall that ψ does not contain any of $j_1, \dots, j_{n+1}$, and thus these nominals cannot be used to access the corresponding agents in the models. Without such an ability, the only way for ψ to spot a difference between M_a and N_a is via a sequence of $\blacklozenge^{-1}$ steps reaching agent b_{n+1} . Observe, however, that since the size of ψ is n , and that both M_a and N_a are of lengths $n + 2$, there is not enough modal depth in ψ to reach the distinguishing state b_{n+1} . Hence, a contradiction. $\square$

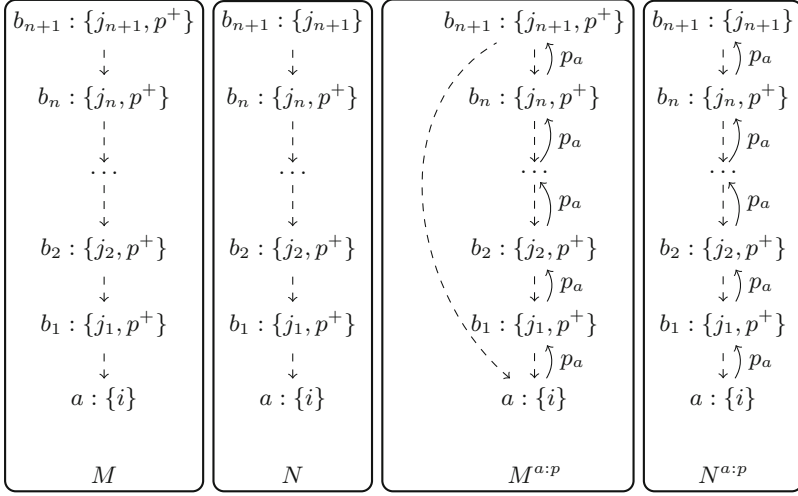


Fig. 4. Models M , N , $M^{a:p}$, and $N^{a:p}$. For models $M^{a:p}$ and $N^{a:p}$ reflexive p_a -arrows and followership arrows from b_k to a for $k \in \{1, \dots, n\}$ are omitted for readability.

Not only is VL more expressive than SVL, its model checking problem is also more computationally demanding. We show this by providing a model checking algorithm for VL that runs in polynomial space. For hardness, we use the classic reduction from quantified Boolean formulas.

Theorem 6. *The model checking problem for VL is PSPACE-complete.*

Proof. To show that the model checking problem for VL is in PSPACE, we present Algorithm 1. For the sake of brevity, we focus on the dynamic modality, and the interested reader can find more on model checking hybrid logics in [15].

Algorithm 1. An algorithm for model checking VL

```

1: procedure MC( $M, a, \varphi$ )
2:   case  $\varphi = [p]\psi$ 
3:     return MC( $M^{a:p}, a, \psi$ )
4:   case  $\varphi = [\pi \cup \tau]\psi$ 
5:     return MC( $M, a, [\pi]\psi$ ) and MC( $M, a, [\tau]\psi$ )

```

The algorithm follows the semantics and its correctness can be shown via induction on φ . Now we argue that the algorithm requires at most polynomial space. The interesting case here is $\varphi = [p]\psi$. Without giving an explicit algorithm for constructing $M^{a:p}$, we note that the size of $M^{a:p}$ is bounded by $\mathcal{O}(|M|^2)$ (the worst-case scenario of $R(p, a)$ and F being universal). Since there are at most $|\varphi|$ symbols in φ , the total space required by the algorithm is bounded by $\mathcal{O}(|\varphi| \cdot |M|^2)$.

To show hardness of the model checking problem we use the classic reduction from the satisfiability of quantified Boolean formula: given a QBF $\Psi :=$

$Q_1 p_1 \dots Q_n p_n \psi(p_1, \dots, p_n)$, where $Q_i \in \{\forall, \exists\}$, determine whether Ψ is true. To reduce the satisfiability of QBF Ψ to the model checking of VL, we construct a model M_a and a formula Ψ' of $\mathcal{VL}$ such that Ψ is true if and only if $M_a \models \Psi'$.

More specifically, given a QBF $Q_1 p_1 \dots Q_n p_n \psi(p_1, \dots, p_n)$, we construct a visibility model $M = (A, F, +, -, V, R)$, where $A = \{a_0, \dots, a_n\}$, $F(a_i) = \{a_0\}$ for all $i \neq 0$, $+(a_i) = p_i$ for all $i \neq 0$, $-(a_i) = \emptyset$ for all i , $V(i_j) = \{a_j\}$, and $R(p, a) = \emptyset$ for all $p \in \text{Top}$ and $a \in A$. Additionally, we assume that there is a topic q that no agent is either *pro* or *contra*. Intuitively, M is a model consisting of $n+1$ agents, where everyone follows agent a_0 , who follows no one. Each agent, apart from a_0 , is *pro* exactly one topic, and no one is *contra* anything. Finally, the translation of the QBF is done recursively as follows:

$$\begin{aligned}\psi'_0 &:= \psi(\Diamond_{i_0:p_1}(i_1 \wedge \Diamond_{i_0:p_1} \top), \dots, \Diamond_{i_0:p_n}(i_n \wedge \Diamond_{i_0:p_n} \top)) \\ \psi'_k &:= \begin{cases} [p_k \cup q] \psi'_{k-1} & \text{if } Q_k = \forall \\ \neg[p_k \cup q] \neg \psi'_{k-1} & \text{if } Q_k = \exists \end{cases} \\ \psi' &:= \psi'_n.\end{aligned}$$

We need to show that

$$Q_1 x_1 \dots Q_n x_n \psi(p_1, \dots, p_n) \text{ is satisfiable iff } M_a \models \psi'$$

Agent a_0 posting on topic p_i means that the truth-value of p_i has been set to 1. If agent a_0 posts on topic q , this means that the truth-value of the corresponding p_i has been set to 0. Since there are no two agents that are *pro* the same topic, the choice of truth values is unambiguous.

We use non-deterministic choice to model quantifiers. The universal quantifier $\forall p_k$ is emulated with $[p_k \cup q] \psi'_{k-1}$ meaning that no matter what agent a_0 chooses to post on, p_k or q , formula ψ'_{k-1} will be true. Similarly, the existential quantifier $\exists p_k$ is emulated with $\neg[p_k \cup q] \neg \psi'_{k-1}$ meaning that agent a_0 can post on a topic, either p_k or q , to make ψ'_{k-1} true. Finally, propositional variable p_j is translated into the formula $\Diamond_{i_0:p_j}(i_j \wedge \Diamond_{i_0:p_j} \top)$ that is true if and only if there has been a post on p_j , and the corresponding agent a_j , who is *pro* p_j , has reposted it. For all other agents a_k , the formula will not hold. Posting on q instead of p_j results in the fact that $\Diamond_{i_0:p_j}(i_j \wedge \Diamond_{i_0:p_j} \top)$ is not satisfied anywhere in the model, thus corresponding to setting p_j to 0. $\square$

5 Conclusion and Future Work

This work was devoted to the analysis of the concept of visibility in social networks using modal logic. After discussing related work from social network analysis, we introduced a logic we named static visibility logic (SVL) and its dynamic extension, visibility logic (VL). We did not give a definite answer as to how one should measure visibility, but proposed several quantitative and qualitative measures relevant to our social network models. To motivate VL, we presented an

example where we show how, given some simple rules of the system, a potential malicious agent can take advantage of the network to expose more agents to a controversial opinion. On the mathematical side, we showed soundness and completeness of SVL with respect to social networks that follow our given rules. We also proved that the language of VL is strictly more expressive than the language of SVL and that the complexity of the model checking problem for VL is *PSPACE*-complete.

As we mention in the paper, an implication of the result $\mathcal{SVL} < \mathcal{VL}$ is that a proof of the completeness of VL using reduction axioms is not possible. Thus one of the open problems is to find a sound and complete axiomatisation of VL.

Another direction for future work is to explore triggering in social network communication. The idea is that seeing a post on a controversial topic might trigger an agent to post a reaction. To do this, we could expand our framework such that agents can not only post on a topic, but also pro or contra a topic. This entails letting $\pi ::= p \mid p^+ \mid p^- \mid \pi \cup \pi$ in the dynamic formula $[\pi]\phi$. Then, we could specify particular controversial topics and add a rule stating that if an agent sees a post that is pro the controversial topic and they are themselves contra, then the agent will post contra the topic, or vice versa. We leave a proper implementation to a later possible extended version of the paper.

Acknowledgements. We would like to thank anonymous reviewers for their insightful comments and constructive criticism.

References

1. Andi, S.: How and why do consumers access news on social media? (2021). <https://reutersinstitute.politics.ox.ac.uk/digital-news-report/2021/how-and-why-do-consumers-access-news-social-media>
2. Areces, C., ten Cate, B.: Hybrid logics. In: Blackburn, P., van Benthem, J., Wolter, F. (eds.) Handbook of Modal Logic, Studies in Logic and Practical Reasoning, vol. 3, pp. 821–868. Elsevier (2007)
3. Baltag, A., Christoff, Z., Rendsvig, R.K., Smets, S.: Dynamic epistemic logics of diffusion and prediction in social networks. *Stud. Log.* **107**(3), 489–531 (2019). <https://doi.org/10.1007/s11225-018-9804-x>
4. Baltag, A., Moss, L.S.: Logics for epistemic programs. *Synthese* **139**(2), 165–224 (2004). <https://doi.org/10.1023/B:SYNT.0000024912.56773.5e>
5. Belardinelli, G.: Gatekeepers in Social Networks: Logics for Communicative Actions. Master’s thesis, ILLC, University of Amsterdam (2019)
6. Blackburn, P., ten Cate, B.: Pure extensions, proof rules, and hybrid axiomatics. *Stud. Log.* **84**(2), 277–322 (2006). <https://doi.org/10.1007/s11225-006-9009-6>
7. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic, Cambridge Tracts in Theoretical Computer Science, vol. 53. CUP (2001)
8. Brady, W.J., Wills, J.A., Jost, J.T., Tucker, J.A., Bavel, J.J.V.: Emotion shapes the diffusion of moralized content in social networks. *Proc. Natl. Acad. Sci.* **114**(28), 7313–7318 (2017)
9. Christoff, Z., Hansen, J.U.: A logic for diffusion in social networks. *J. Appl. Log.* **13**(1), 48–77 (2015)

10. Christoff, Z., Hansen, J.U., Proietti, C.: Reflecting on social influence in networks. *J. Logic Lang. Inf.* **25**(3–4), 299–333 (2016). <https://doi.org/10.1007/s10849-016-9242-y>
11. Dennis, L.A., Fu, Y., Slavkovik, M.: Markov chain model representation of information diffusion in social networks. *J. Log. Comput.* (2022)
12. van Ditmarsch, H., van der Hoek, W., Kooi, B.: *Dynamic Epistemic Logic*, Synthese Library, vol. 337. Springer (2008). <https://doi.org/10.1007/978-1-4020-5839-4>
13. Easley, D., Kleinberg, J.: *Networks. Crowds and Markets*, CUP (2010)
14. Fischer, M.J., Ladner, R.E.: Propositional dynamic logic of regular programs. *J. Comput. Syst. Sci.* **18**(2), 194–211 (1979)
15. Franceschet, M., de Rijke, M.: Model checking hybrid logics (with an application to semistructured data). *J. Appl. Log.* **4**(3), 279–304 (2006)
16. Jackson, M.O., Rogers, B.W., Zenou, Y.: The economic consequences of social-network structure. *J. Econ. Lit.* **55**(1), 49–95 (2017)
17. Kooi, B., Renne, B.: Arrow update logic. *Rev. Symb. Log.* **4**(4), 536–559 (2011)
18. Kurtonina, N., de Rijke, M.: Bisimulations for temporal logic. *J. Logic Lang. Inf.* **6**(4), 403–425 (1997). <https://doi.org/10.1023/A:1008223921944>
19. Lorini, E., Sartor, G.: A STIT logic for reasoning about social influence. *Stud. Log.* **104**(4), 773–812 (2016). <https://doi.org/10.1007/s11225-015-9636-x>
20. Napoli, P.M.: Social media and the public interest: governance of news platforms in the realm of individual and algorithmic gatekeepers. *Telecomm. Policy* **39**(9), 751–760 (2015)
21. Nguyen, C.T.: Echo chambers and epistemic bubbles. *Episteme* **17**(2), 141–161 (2020)
22. Pedersen, M.Y., Slavkovik, M., Smets, S.: Social bot detection as a temporal logic model checking problem. In: Ghosh, S., Icard, T. (eds.) *LORI 2021*. LNCS, vol. 13039, pp. 158–173. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-88708-7_13
23. Pedersen, M.Y., Smets, S., Ågotnes, T.: Analyzing echo chambers: a logic of strong and weak ties. In: Blackburn, P., Lorini, E., Guo, M. (eds.) *LORI 2019*. LNCS, vol. 11813, pp. 183–198. Springer, Heidelberg (2019). https://doi.org/10.1007/978-3-662-60292-8_14
24. Pedersen, M.Y., Smets, S., Ågotnes, T.: Further steps towards a logic of polarization in social networks. In: Dastani, M., Dong, H., van der Torre, L. (eds.) *CLAR 2020*. LNCS (LNAI), vol. 12061, pp. 324–345. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-44638-3_20
25. Pedersen, M.Y., Smets, S., Ågotnes, T.: Modal logics and group polarization. *J. Log. Comput.* **31**(8), 2240–2269 (2021)
26. Plaza, J.: Logics of public communications. *Synthese* **158**(2), 165–179 (2007). <https://doi.org/10.1007/s11229-007-9168-7>
27. Rathore, N.C., Tripathy, S.: Effective visibility prediction on online social network. *IEEE Trans. Comput. Soc. Syst.* **8**(2), 355–364 (2021)
28. Samanta, M., Pal, P., Mukherjee, A.: A novel scheme for abatement of privacy concern by controlling the reachability in online social network. In: Abraham, A., Cherukuri, A.K., Madureira, A.M., Muda, A.K. (eds.) *SoCPaR 2016*. AISC, vol. 614, pp. 686–697. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-60618-7_67
29. Tang, J., Musolesi, M., Mascolo, C., Latora, V.: Characterising temporal distance and reachability in mobile and online social networks. *Comput. Commun. Rev.* **40**(1), 118–124 (2010)



The Alternation Hierarchy of the μ -calculus over Weakly Transitive Frames

Leonardo Pacheco^(✉)  and Kazuyuki Tanaka

Mathematical Institute, Tohoku University, Sendai, Japan
leonardovpacheco@gmail.com, tanaka.math@tohoku.ac.jp

Abstract. It is known that the μ -calculus collapses to its alternation-free fragment over transitive frames and to modal logic over equivalence relations. We adapt a proof by D'Agostino and Lenzi to show that the μ -calculus collapses to its alternation-free fragment over weakly transitive frames. As a consequence, we show that the μ -calculus with derivative topological semantics collapses to its alternation-free fragment. We also study the collapse over frames of S4.2, S4.3, S4.3.2, S4.4 and KD45, logics important for Epistemic Logic. At last, we use the μ -calculus to define degrees of ignorance on Epistemic Logic and study the implications of μ -calculus's collapse over the logics above.

Keywords: μ -calculus · Fragments of the alternation hierarchy · Topological modal logic · Epistemic logic

1 Introduction

The modal μ -calculus is obtained by adding least and greatest fixed-points operators to modal logic. By modal logic, we mean propositional logic with dual modalities $\Box$ and $\Diamond$. In general, the μ -calculus is much more expressive than modal logic.

The alternation depth is a measure of the complexity of μ -formulas, which calibrates the entanglement of the fixed-point operators in a formula. The alternation hierarchy is generated by the alternation depth of the formulas. Bradfield [7, 8] showed that the alternation hierarchy is strict over all Kripke models. That is, for each $n \in \mathbb{N}$ there is a μ -formula φ with alternation depth n which is not equivalent to any μ -formula with alternation depth less than n .

If we consider restricted classes of frames, the situation may change. Alberucci and Facchini [1] showed that if we only consider frames that are equivalence relations, then every μ -formula is equivalent to a formula of modal logic. Therefore, in this case, adding fixed-points does not increase the expressiveness. They also showed that, over transitive frames, the alternation hierarchy collapses to its alternation-free fragment, that is, each μ -formula is equivalent

Supported by MEXT, Japan.

to an alternation-free formula. An alternation-free formula is a formula with no entangled fixed-point operators, that is, the evaluation of any fixed-point operator η does not depend on operators outside the scope of η . On the other hand, the alternation hierarchy is still strict on reflexive frames [1], symmetric frames [12] and tree-like frames [3, 21]. The alternation hierarchy also collapses to modal logic on frames of GL, the reverse well-founded transitive frames [2]. In this paper, we study the alternation hierarchy over the class of weakly transitive frames and over a few subclasses of the transitive frames.

We first show that the alternation hierarchy over weakly transitive frames collapses to its alternation-free segment, modifying a proof of the collapse for transitive frames due to D’Agostino and Lenzi [11]. We also study derivative topological semantics. The modal logic $\mathbf{wK4}$ is complete for derivative topological semantics [13], and also for relational semantics on weakly transitive frames. We use a theorem of Baltag *et al.* [5] to show that the alternation hierarchy collapses to its alternation-free fragment over derivative topological semantics. The derivative topological semantics for modal logic was first defined by McKinsey and Tarski [23], and is more expressive than the interior topological semantics.

We study the alternation hierarchy over frames which satisfy the modal logics **S4.2**, **S4.3**, **S4.3.2**, **S4.4** and **KD45**. We show that the alternation hierarchy collapses to its alternation-free fragment over the first two classes of frames, and to modal logic over the last three. These logics are used for epistemic logic. Stalnaker [28] discusses these logics (along with **S5**) and argues that **S4.2** is the right formalization for the logic of knowledge, as well as **KD45** for the logic of belief. Van der Hoek and Lomuscio [30] introduced a modality I for ignorance, with $I\varphi := \neg K\varphi \wedge \neg K\neg\varphi$. We analyze some μ -formulas which are not equivalent to any modal formula over **S4.2** and **S4.3** and interpret it and its approximants as degrees of ignorance. We argue that logics “closer” to **S4.2** allow more degrees of ignorance, and logics “closer” to **S5** allow fewer degrees of ignorance.

2 Preliminaries

The μ -calculus. Fix a set Prop of propositional symbols and a set Var of variable symbols. The *formulas* of the μ -calculus are generated by the following grammar:

$$\varphi := P \mid \neg P \mid X \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \Box\varphi \mid \Diamond\varphi \mid \mu X.\varphi \mid \nu X.\varphi,$$

where $P \in \text{Prop}$ and $X \in \text{Var}$. We write $\Delta\varphi$ for $\Box\varphi$ or $\Diamond\varphi$, and $\eta X.\varphi$ for $\mu X.\varphi$ or $\nu X.\varphi$. A formula φ is called a *sentence* iff every variable X of φ is *bound*, namely in a scope of ηX . We define the *negation* of formulas by recursion: $\neg(\neg\varphi) := \varphi$, $\neg(\varphi \wedge \psi) := \neg\varphi \vee \neg\psi$, $\neg(\Box\varphi) := \Diamond\neg\varphi$, and $\neg\mu X.\varphi := \nu X.\neg\varphi[X/\neg X]$. The definition for $\vee$, $\Diamond$ and ν are similar. Note that each occurrence of a variable X in a formula φ is *positive*. We define $\top := P \vee \neg P$ and $\perp := P \wedge \neg P$, where P is a fixed propositional symbol. The *set of subformulas* of a formula φ is denoted by $\text{Sub}(\varphi)$. A variable X is *guarded* in a formula φ iff every occurrence of X in φ is in the scope of some Δ . A formula φ is *guarded* iff, for any subformula $\eta X.\psi$

of φ , X is guarded in ψ . A formula φ is *well-bound* iff, for any X which occurs bound in φ , X and ηX occur only once each in φ . A formula is *well-named* iff it is guarded and well-bound. For more on the modal μ -calculus, see [9, 22].

Kripke Models. A *Kripke model* is a triple $\langle W, R, V \rangle$ where: W is a set, the set of *possible worlds* (or *states*); R is a binary relation on W , the *accessibility relation*; and V is a function from propositional symbols to subsets of W , the *valuation function*. Kripke models are also known as *transition systems*. A *Kripke frame* is a pair $\langle W, R \rangle$ consisting only of the set of possible worlds W and a binary relation R .

Relational Semantics. Let $M = \langle W, R, V \rangle$ be a Kripke model and φ be a μ -formula, we define the valuation $\|\varphi\|^M$ of φ on M inductively:

- $\|P\|^M := V(P)$;
- $\|X\|^{[M, X := A]} := A$;
- $\|\neg\varphi\|^M := W \setminus \|\varphi\|^M$;
- $\|\varphi \wedge \psi\|^M := \|\varphi\|^M \cap \|\psi\|^M$;
- $\|\Box\varphi\|^M := \{w \in W \mid \forall s. wRs \rightarrow s \in \|\varphi\|^M\}$;
- $\|\mu X.\varphi\|^M$ is the least fixed-point of Γ_φ ;
- and $\|\nu X.\varphi\|^M$ is the greatest fixed-point of Γ_φ ,

where Γ_φ maps $X \subseteq W$ to $\|\varphi(X)\|^M$. As X is positive in $\varphi(X)$, the operator Γ_φ is monotone and its least and greatest fixed-points are well-defined. The definitions of $\vee$, $\Diamond$, $\neg$ are dual. We write $\|\varphi\|$ for $\|\varphi\|^M$ when it is clear from context. Note that any μ -formula is equivalent to a well-named μ -formula.

Game Semantics. We also consider the game semantics for the μ -calculus. We present simplified semantics valid only for alternation-free sentences, as we will only use it on models of **S4**. Fix a well-named μ -formula φ , a Kripke model $M = \langle W, R, V \rangle$, and a state w . The evaluation game for $M, w \models \varphi$ is a game for two players: Verifier and Refuter, denoted by **V** and **R** respectively. The positions of the game are of the form $\langle \psi, v \rangle$ with $\psi \in \text{Sub}(\varphi)$ and $v \in W$. The initial position is $\langle \varphi, w \rangle$. Each position $\langle \psi, v \rangle$ is owned by a player, who makes the next move. Table 1 summarizes the ownership of $\langle \psi, v \rangle$ and admissible moves on it; both are determined by the construction of ψ . On Table 1, ψ_X denotes the unique subformula of φ such that X occurs freely in ψ_X and $\eta X.\psi_X \in \text{Sub}(\varphi)$. **V** wins a run when **R** can't make a move, or if some formula $\nu X.\psi$ appears infinitely many times¹. Relational semantics and game semantics are equivalent: **V** has a winning strategy in the evaluation game for $M, w \models \varphi$ iff $w \in \|\varphi\|^M$. See [17] for details on game semantics and a proof that relational semantics and game semantics coincide.

¹ The winning condition for general formulas is: **V** wins a run when **R** can't make a move or the outermost $\eta X.\psi$ which appears infinitely often in the run is some $\nu X.\varphi$.

Table 1. The rules of evaluation game for modal μ -calculus.

Verifier		Refuter	
Position	Admissible moves	Position	Admissible moves
$\langle \psi_1 \vee \psi_2, w \rangle$	$\{\langle \psi_1, w \rangle, \langle \psi_2, w \rangle\}$	$\langle \psi_1 \wedge \psi_2, w \rangle$	$\{\langle \psi_1, w \rangle, \langle \psi_2, w \rangle\}$
$\langle \Diamond \psi, w \rangle$	$\{\langle \psi, v \rangle \mid \langle w, v \rangle \in R\}$	$\langle \Box \psi, w \rangle$	$\{\langle \psi, v \rangle \mid \langle w, v \rangle \in R\}$
$\langle P, w \rangle$ and $w \notin V(P)$	$\emptyset$	$\langle P, w \rangle$ and $w \in V(P)$	$\emptyset$
$\langle \neg P, w \rangle$ and $w \in V(P)$	$\emptyset$	$\langle \neg P, w \rangle$ and $w \notin V(P)$	$\emptyset$
$\langle \mu X. \psi_X, w \rangle$	$\{\langle \psi_X, w \rangle\}$	$\langle \nu X. \psi_X, w \rangle$	$\{\langle \psi_X, w \rangle\}$
$\langle X, w \rangle$	$\{\langle \psi_X, w \rangle\}$	$\langle X, w \rangle$	$\{\langle \psi_X, w \rangle\}$

Alternation Hierarchy. We define the syntactical alternation hierarchy as follows:

- $\Sigma_0^\mu = \Pi_0^\mu$ is the set of all μ -formulas with no fixed-point operators.
- Σ_{n+1}^μ is the closure of $\Sigma_n^\mu \cup \Pi_n^\mu$ under propositional symbols, modal operators, μX , and the substitution: for a $\varphi(X) \in \Sigma_{n+1}^\mu$ and $\psi \in \Sigma_{n+1}^\mu$ such that no free variable of ψ is captured by φ , $\varphi(\psi) \in \Sigma_{n+1}^\mu$.
- Π_{n+1}^μ is the closure of $\Sigma_n^\mu \cup \Pi_n^\mu$ under propositional symbols, modal operators, νX , and the analogous substitution.
- Δ_n^μ is $\Sigma_n^\mu \cap \Pi_n^\mu$.

The alternation-free fragment of the μ -calculus is Δ_2^μ .

We say that the alternation hierarchy collapses to modal logic over a class of frames $\mathbf{F}$ iff every μ -formula is equivalent to a modal formula over the frames in $\mathbf{F}$. Similarly, the alternation hierarchy collapses to the alternation-free fragment over a class of frames $\mathbf{F}$ iff every μ -formula is equivalent to an alternation-free formula over the frames in $\mathbf{F}$; in this case, we also suppose that the alternation hierarchy does not collapse to modal logic.

Frame Conditions. Modal formulas are commonly used to define frame conditions. For example, a frame $F = (W, R)$ satisfies $\Box P \rightarrow P$ iff R is a reflexive relation. The axioms and frame conditions considered in this paper are listed in Table 2. By **wK4**, **S4**, **S4.x** and **S5** we mean the respective closures of $\{K, w4\}$, $\{K, T, 4\}$, $\{K, T, 4, .x\}$ and $\{K, T, 5\}$ under necessitation and the rules of propositional logic. All these logics are strongly complete with respect to their respective frame conditions. By a *frame of a logic* $\mathcal{L}$, we mean a frame which satisfies the frame conditions specified by the axioms of $\mathcal{L}$. For more information on frame conditions, see [6] or any other standard textbook on modal logic.

Table 2. Axioms and frame conditions.

Axiom Name	Axiom	Frame conditions
K	$\Box(P \rightarrow Q) \rightarrow (\Box P \rightarrow \Box Q)$	
Serial: D	$\Box P \rightarrow \Diamond P$	$\exists v.wRv$
Reflexive: T	$\Box P \rightarrow P$	wRw
Transitive: 4	$\Box P \rightarrow \Box \Box P$	$wRu \wedge uRv \rightarrow wRv$
Euclidean: 5	$\Diamond P \rightarrow \Box \Diamond P$	$wRu \wedge wRv \rightarrow uRv$
Convergent: .2	$\Diamond \Box P \rightarrow \Box \Diamond P$	$wRu \wedge wRv \rightarrow \exists tuRt \wedge vRt$
Weakly Connected: .3	$\Box(\Box P \rightarrow Q) \vee \Box(\Box Q \rightarrow P)$	$wRu \wedge wRv \rightarrow u = v \vee uRv \vee vRu$
Semi-Euclidean: .3.2	$(\Diamond P \wedge \Box \Diamond Q) \rightarrow \Box(\Diamond P \vee Q)$	$wRv \wedge \neg vRw \wedge wRu \rightarrow uRv$
.4	$(P \wedge \Diamond \Box P) \rightarrow \Box P$	$wRv \wedge w \neq v \wedge wRu \rightarrow uRv$
Weakly Transitive: $w4$	$\Diamond \Diamond P \rightarrow P \vee \Diamond P$	$wRu \wedge uRv \rightarrow w = v \vee wRv$

3 The Collapse to the Alternation-Free Fragment over Weakly Transitive Frames

In this section we prove that the μ -calculus collapses to its alternation-free fragment over weakly transitive frames. This proof is a generalization of D’Agostino and Lenzi’s proof of the collapse over transitive frames [11].

Let $\varphi(X)$ be a μ -formula. Define by recursion $\varphi^0(X) = X$ and $\varphi^{n+1}(X) = \varphi(\varphi^n(X))$, for all $n \in \mathbb{N}$. The key lemma for the collapse over weakly transitive frames is:

Lemma 1. *Let $\varphi(X)$ be a μ -formula. Over weakly transitive frames,*

$$\Diamond \mu X. \varphi(X) \equiv \Diamond \varphi^2(\perp) \text{ and } \Box \nu X. \varphi(X) \equiv \Box \varphi^2(\top).$$

Proof. We only prove $\Diamond \mu X. \varphi \equiv \Diamond \varphi^2(\perp)$ as the other equivalence is dual.

Fix a Kripke model $M = \langle W, R, V \rangle$ with R weakly transitive and a world $w \in W$. By weak transitivity, we can suppose $W = \{w\} \cup \{v \mid wRv\}$, as only w worlds accessible from w affect the valuation of μ -formulas at w . As X is positive in $\varphi(X)$, $w \in \|\Diamond \varphi^2(\perp)\|$ implies $w \in \|\Diamond \mu X. \varphi(X)\|$.

Suppose now that $w \in \|\Diamond \mu X. \varphi\|$. Then $v \in \|\mu X. \varphi\|$ for some v accessible from w . As $\|\mu X. \varphi\|$ is not empty, $\|\varphi(\perp)\|$ is also not empty. We now have two cases:

- If $v \in \|\varphi(\perp)\|$ for some v accessible from w , then $v \in \|\varphi^2(\perp)\|$ and we have finished the proof.
- Otherwise, $\|\varphi(\perp)\|$ contains only w and wRw does not hold. If $\|\varphi(\varphi(\perp))\|$ had no element of $W \setminus \{w\}$, we would have $\|\mu X. \varphi\| = \{w\}$, a contradiction.

Therefore $w \in \|\Diamond \varphi^2(\perp)\|$. □

Let φ be a well-named μ -formula and X be a bound variable in φ . Let ψ be such that X occurs in ψ and $\eta X. \psi \in \text{Sub}(\varphi)$. X is *existential* iff X is not in the scope of any $\Box$ in ψ , that is, X is only in the scope of diamonds. X is *weakly*

universal iff X is in the scope of some $\square$ in ψ . Given a well-named φ , we assign to each occurrence νX of greatest fixed-point operator a sign $+$ or $-$ iff X is existential or weakly universal, respectively. For example,

$$\nu^- X \nu^+ Y \nu^- Z. (\square X \vee \diamond Y \vee \square \diamond Z).$$

Lemma 2. *Suppose the formula $\nu X.\varphi$ is well-named and X is weakly universal in it. Then, over weakly transitive frames,*

$$\nu X^-.\varphi(X) \equiv \varphi^3(\top).$$

Proof. We can write $\nu X.\varphi(X)$ as $\nu X.\alpha(\square\beta(X))$. We have that $\beta(\nu X.\alpha(\square\beta(X)))$ is equivalent to $\nu Y.\beta(\alpha(\square Y))$ (see Lemma 3.8 of [11]). And so

$$\begin{aligned} \varphi^3(\top) &\equiv \alpha(\square\beta(\alpha(\square\beta(\alpha(\square\beta(\top)))))) \\ &\rightarrow \alpha(\square\beta(\alpha(\square\beta(\alpha(\top)))) \\ &\equiv \alpha(\square\nu Y.\beta(\alpha(\square Y))) \\ &\equiv \alpha(\square\beta(\nu X.\alpha(\square\beta(X)))) \\ &\equiv \nu X.\alpha(\square\beta(X)) \\ &\equiv \nu X.\varphi(X) \end{aligned}$$

So $\varphi^3(\top)$ implies $\nu X.\varphi(X)$. As X is positive in φ , $\nu X.\varphi(X)$ implies $\varphi^3(\top)$. Therefore $\nu X.\varphi(X)$ and $\varphi^3(\top)$ are equivalent. $\square$

We now review the machinery developed in Sect. 3 of D’Agostino and Lenzi. It will be used to prove the collapse to the alternation-free fragment over weakly transitive frames. Let Γ, Δ be finite sets of μ -formulas, define

$$\text{Cover}(\Gamma, \Delta) := \left(\bigwedge_{\varphi \in \Gamma} \diamond \varphi \right) \wedge \square \left(\bigvee_{\psi \in \Delta} \psi \right).$$

The class of *bidisjunctive* formulas is the least class containing all literals and closed under:

- disjunctions;
- restricted fixed-points: if φ is bidisjunctive and there is no subformula of φ of the form $X \wedge \psi$, then $\eta X.\varphi$ is bidisjunctive; and
- covers: if Γ, Δ are finite sets of bidisjunctive formulas and σ is a conjunction of literals, then $\sigma \wedge \text{Cover}(\Gamma, \Delta)$ is also bidisjunctive.

A *special* formula is a well-named bidisjunctive Π_2^μ -formula where (in the tree representation of φ) there is a node labeled by a Cover between any $\nu X.\varphi$ and $\mu Y.\psi$. D’Agostino and Lenzi [11] proved that any Π_2^μ -formula is equivalent (over all frames) to a special formula.

We say a μ -formula φ is in weakly universal prenex form iff it is of the form

$$\nu X_k^- \dots \nu X_1^- . \psi$$

and ψ does not contain any weakly universal occurrence of ν . Let $\varphi := \nu X_1^- . \psi$ be in weakly universal prenex form. By Lemma 2, φ is equivalent to $\varphi^* := \psi(\psi'(\psi''(\top)))$ over weakly transitive frames. Here, ψ' is obtained by renaming all bound variables X of ψ with fresh variables X' (and ψ'' from ψ'), so φ^* is well named. If φ is special, it follows from the definition of special formulas that so is φ^* . Thus any weakly universal special formula is equivalent to a special formula without weakly universal occurrences of ν . Furthermore, any well named special formula is equivalent to a well named special formula in weakly universal form by Lemma 3.23 of [11].

Theorem 1. *The alternation hierarchy collapses to the alternation-free fragment over frames of $\mathbf{wK4}$ (with relational semantics).*

Proof. Let φ be a μ -formula in Π_2^μ . We can suppose φ is an special formula without weakly universal occurrences of ν . Let $\nu X.\psi_1$ be a subformula of φ . Suppose that there is a (maximal) subformula $\mu Y.\psi_2$ of $\nu X.\psi_1$. As φ is special, there is a (minimal) $\text{Cover}(\Gamma, \Delta) \in \text{Sub}(\nu X.\psi_1)$ such that $\mu Y.\psi_2$ is a subformula of some formula in Γ or Δ . $\mu Y.\psi_2$ is not a subformula of a formula in Δ as this would make X weakly universal. So Γ is

$$\Gamma = \{\theta_1 \vee (\theta_2 \vee (\dots (\theta_k \vee \mu Y.\psi_2) \dots))\} \cup \Sigma.$$

$\text{Cover}(\Gamma, \Delta)$ is equivalent to $\text{Cover}(\Gamma', \Delta)$ with

$$\Gamma' = \{\theta_1 \vee (\theta_2 \vee (\dots (\theta_k \vee \psi_2(\psi_2(\perp))) \dots))\} \cup \Sigma,$$

as $\diamond$ and $\vee$ commute and Lemma 1. Therefore, if we substitute $\psi_2(\psi_2(\perp))$ for $\mu Y.\psi_2$ in φ we get a formula φ' which is equivalent to φ .

We can then suppose that φ is equivalent to a μ -formula φ' such that: if $\nu X.\psi_1 \in \text{Sub}(\varphi')$ then there is no $\mu Y.\psi_2 \in \text{Sub}(\nu X.\psi_1)$ with a free occurrence of X . As we can build φ' from $\Sigma_1^\mu \cup \Pi_1^\mu$ using only substitutions allowed in the construction of both Σ_2^μ and Π_2^μ , φ' is Δ_2^μ . That is, φ' is alternation-free and φ is equivalent to an alternation-free formula. $\square$

4 The Collapse over Topological Semantics

We now define derivative topological models $\mathcal{X} = \langle W, \tau, V \rangle$. As in a Kripke model, W is the set of possible worlds and V is the valuation function. τ is a topology on W . We define the valuation $\|\varphi\|^\mathcal{X}$ inductively as in relational semantics. The definition of the valuation of conjunctions, disjunctions, negations, and fixed-points is the same as in relational semantics. We think of the $\diamond$ modality as the *Cantor derivative*:

$$w \in \|\diamond\varphi\|^\mathcal{X} \text{ iff } w \text{ is a limit point of } \|\varphi\|^\mathcal{X},$$

where w is a limit point of a set $X \subseteq W$ iff it is the limit of a sequence of points in X .

Interior topological semantics is also studied in the literature, where the semantics for $\|\Box_i\varphi\|^{\mathcal{X}}$ defined to be the interior of $\|\varphi\|^{\mathcal{X}}$. The closure of a set X consists of the points in X and the limit points of X . Therefore, the interior modality can be defined using derivative topological semantics: $\Box_i\varphi := \varphi \wedge \Diamond\neg\varphi$. This implies derivative topological semantics is more expressive than interior topological semantics. For more information on topological semantics see [5, 29].

On relational semantics, **wK4** is complete for the weakly transitive frames. Esakia [13] showed that **wK4** is also complete for topological spaces with $\Diamond$ interpreted as the Cantor derivative.

Let $\mathcal{X} = \langle W, \tau, V \rangle$ be a topological model and $M = \langle W, R, V \rangle$ be a Kripke model. $\mathcal{X}$ and M are equivalent iff $\|\varphi\|^{\mathcal{X}} = \|\varphi\|^M$ for all μ -formula φ . We say topological space is *Alexandroff* iff arbitrary intersections of open sets are open. Each Alexandroff topological model is equivalent an (irreflexive) weak transitive model.

We will use the following result to relate the alternation hierarchies on relational and topological semantics:

Theorem 2 (Baltag et al. [5]). *The μ -calculus with topological semantics has the finite model property. That is, if a formula is satisfiable by some topological model, then it is satisfiable by a finite topological model.*

This theorem is proved by using “final” canonical models, as the standard canonical models are not amenable to μ -calculus.

We prove the collapse of the alternation hierarchy on topological semantics using the collapse over weakly transitive Kripke frames.

Theorem 3. *The alternation hierarchy collapses to its alternation-free fragment on topological semantics.*

Proof. Fix a μ -formula φ and a topological model $\mathcal{X} = \langle W, \tau, V \rangle$. Suppose φ is not equivalent to the alternation-free formula φ' . The finite model property implies that there is a finite topological model M_f where φ is not equivalent to φ' . Any finite topological model is Alexandroff. So M_f is equivalent to a Kripke model extending a weakly transitive frame. And so φ and φ' are not equivalent over weakly transitive frames. As this argument holds for any alternation-free φ' , then alternation hierarchy doesn't collapse to its alternation-free fragment over weakly transitive frames. This contradicts Theorem 1. $\square$

5 The Collapse over **S4.2**, **S4.3**, **S4.3.2**, **S4.4** and **KD45**

We first show that the alternation hierarchy doesn't collapse to modal logic over **S4.2** and **S4.3**. This was already known for **S4**. After that we show that the alternation hierarchy collapses to modal logic over **S4.3.2**, **S4.4** and **KD45**.

Theorem 4. *The alternation hierarchy collapses to its alternation-free fragment over **S4.2** and **S4.3**.*

Proof. First note that frames of S4.2 are transitive, and frames of S4.3 satisfy S4.2. Therefore, the collapse to the alternation-free fragment for S4 implies the collapse for S4.2 and S4.3; and the non-collapse to modal logic for S4.3 implies the non-collapse for S4.2. The collapse to the alternation-free fragment for transitive frames is already established [1, 11]. So we only need to show that S4.3 does not collapse to modal logic.

Let $\varphi(X) := \Diamond(P \wedge \Diamond(\neg P \wedge X))$. We show $\nu X.\varphi(X)$ is not equivalent to any modal formula. For a contradiction, suppose $\nu X.\varphi(X)$ is equivalent to a modal formula φ' . Given a model $M = \langle W, R, V \rangle$ over an S4.3 frame $F = \langle W, R \rangle$ and $w_0 \in W$, $\nu X.\varphi$ holds on w_0 iff there is a sequence $\{w_i\}_{i \in \omega}$ such that $w_i R w_{i+1}$, $w_{2i+1} \in \llbracket P \rrbracket$ and $w_{2i+2} \in \llbracket \neg P \rrbracket$. For $n \in \omega$, define a model $M_n = \langle W_n, R_n, V_n \rangle$ by

- $W_n = \{w_1, w_2, \dots, w_{2n}\}$;
- $R_n = \{\langle w_i, w_j \rangle \mid 0 \leq w_i \leq w_j \leq 2n - 1\}$; and
- $V_n(P) = \{w_i \mid i \text{ is odd}\}$.

M_n has a path of length $2n$ with P and $\neg P$ appearing alternately, but no such path is infinite; that is, $M_n, w_1 \models \varphi^n(\top) \wedge \neg \varphi'$ (Fig. 1).

Define $L = S4.3 + \neg \varphi' + \{\top, \varphi^1, \varphi^2, \dots\}$. Note that the compactness theorem holds for S4.3, as its frames are first-order definable. As each finite subtheory of L is satisfied by some M_n , L has a model with an S4.3 frame. S4.3 also has the finite model property, and so there is a finite model (M_f, w_f) of L with S4.3 frame. An application of the Pigeonhole Principle implies that $\nu X.\varphi$ holds on (M_f, w_f) , as M_f is finite and has paths of arbitrary length starting from w_f where P and $\neg P$ hold alternately. Therefore (M_f, w_f) satisfies both $\nu X.\varphi$ and $\neg \nu X.\varphi$, a contradiction. Therefore $\nu X.\varphi$ is not equivalent to any modal formula over frames of S4.3. $\square$

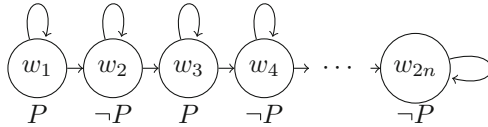


Fig. 1. M_n is model of $\varphi^n \wedge \neg \varphi'$.

The collapse to modal logic over equivalence relations was proved by Alberucci and Facchini [1]. The proof idea is as follows. Fixed a μ -formula $\varphi(X)$, a model $M = \langle W, R, V \rangle$ and $s \in W$. Define $\varphi^0(X) := X$ and $\varphi^{n+1} := \varphi(\varphi^n(X))$. If $w \in \llbracket \varphi^2(\top) \rrbracket^M$, then we can “glue” winning strategies for $\forall$ on the evaluation game for $M, w \models \varphi^2(\top)$ to define a winning strategy for $\forall$ on the evaluation game for $M, w \models \varphi^3(\top)$. One then shows that $\llbracket \varphi^2(\top) \rrbracket \subseteq \llbracket \varphi^3(\top) \rrbracket$. Since $\llbracket \varphi^3(\top) \rrbracket \subseteq \llbracket \varphi^2(\top) \rrbracket$ holds, we have $\llbracket \nu X.\varphi(X) \rrbracket = \llbracket \varphi^2(\top) \rrbracket$ too.

Let $F = \langle W, R \rangle$ be a transitive frame, the strongly connected component of $w \in W$ is the set $\{w' \mid w R w' \wedge w' R w\}$. The following lemma is used to “glue” the strategies:

Lemma 3 (Alberucci and Facchini [1]). *Let M be a transitive Kripke model, w' be a member of the strongly connected component of w , φ be a μ -formula, and $\Delta \in \{\Box, \Diamond\}$. Then $w \in \|\Delta\varphi\|^M$ iff $w' \in \|\Delta\varphi\|^M$.*

The lemma follows from the transitivity of M and the definitions of $\Box, \Diamond$.

Theorem 5. *The alternation hierarchy collapses to modal logic over S4.3.2 and S4.4.*

Proof. Any frame of S4.4 is also a frame of S4.3.2, so we only need to prove the collapse over S4.3.2.

Let $F = \langle W, R \rangle$ satisfy S4.3.2. Without loss of generality, we can suppose that W is partitioned into two sets INI, FIN such that $\text{INI} \cup \text{FIN} = W$, $\text{INI} \cap \text{FIN} = \emptyset$; xRx' for all $x, x' \in \text{INI}$; yRy' for all $y, y' \in \text{FIN}$; and $xRy \wedge \neg yRx$ for all $x \in \text{INI}$, $y \in \text{FIN}$. This follows by the frame condition of S4.3 and the fact that the evaluation of a formula on a world w depends only on worlds accessible from w .

Let $M = \langle W, R, V \rangle$ be a model over F and $\varphi(X)$ be a formula of μ -calculus. We suppose that $\varphi(X)$ is well-named and of the form $\alpha(\Delta\beta(X))$. We show that if $w \in \|\varphi^3(\top)\|$ then $w \in \|\varphi^4(\top)\|$, which implies $\nu X.\varphi$ is equivalent to $\varphi^3(\top)$.

Let $w \in \|\varphi^3(\top)\|$. Then the player $\mathbf{V}$ has a winning strategy σ in the evaluation game starting with $\langle \varphi^3(\top), w \rangle$. We now define a winning strategy σ' for $\mathbf{V}$ in the evaluation game for $w \in \|\varphi^4(\top)\|$. σ' starts by emulating σ : if $\sigma(\psi(\top), v) = \langle \psi'(\top), v' \rangle$ then $\sigma'(\psi(\varphi(\top)), v) = \langle \psi'(\varphi(\top)), v' \rangle$. If we get to a position $\langle \Delta\beta(\varphi^2(\top)), w'' \rangle$, then we must have passed by a position $\langle \Delta\beta(\varphi^3(\top)), w' \rangle$ since φ is well-named.

Suppose w' and w'' are both in INI or FIN . By our choice of σ , the equivalent run starting at $\langle \varphi^3(\top), w \rangle$ where $\mathbf{V}$ uses σ is also an initial segment of a winning run:

$$\langle \varphi^3(\top) \rangle \rightarrow \dots \rightarrow \langle \Delta\beta(\varphi^2(\top)), w' \rangle \rightarrow \dots \rightarrow \langle \Delta\beta(\varphi(\top)), w'' \rangle.$$

Therefore $w' \in \|\Delta\beta(\varphi^2(\top))\|$. By Lemma 3, $w'' \in \|\Delta\beta(\varphi^2(\top))\|$, and so there is a winning strategy $\sigma_{w''}$ for the game starting on $\langle \Delta\beta(\varphi^2(\top)), w'' \rangle$; $\mathbf{V}$ then uses $\sigma_{w''}$ until the game is over.

If $w' \in \text{INI}$ and $w'' \in \text{FIN}$, we keep emulating σ . Suppose we end on a position of the form $\langle \Delta\beta(\varphi(\top)), w''' \rangle$. We must have $w''' \in \text{FIN}$. As above, the corresponding run starting at $\langle \varphi^3(\top), w \rangle$ where $\mathbf{V}$ uses σ is also a partial segment of a winning run:

$$\langle \varphi^3(\top) \rangle \rightarrow \dots \rightarrow \langle \Delta\beta(\varphi(\top)), w' \rangle \rightarrow \dots \rightarrow \langle \Delta\beta(\top), w'' \rangle.$$

So $w'' \in \|\Delta\beta(\varphi(\top))\|$ and by Lemma 3 $w''' \in \|\Delta\beta(\varphi(\top))\|$. $\mathbf{V}$ then changes to a strategy $\sigma_{w'''}$ which is winning for the evaluation game of $w''' \in \|\Delta\beta(\varphi(\top))\|$. Any run following σ' is going to be a win for $\mathbf{V}$: we either switch to a winning strategy on a simpler game or essentially play a game following σ which is winning. Therefore $w \in \|\varphi^4(\top)\|$. $\square$

Alberucci and Facchini showed that over equivalence relations, $\|\varphi^2(\top)\| = \|\varphi^3(\top)\|$ for any Kripke model and μ -formula. We use this result directly to prove the collapse over KD45.

Theorem 6. *The alternation hierarchy collapses to modal logic over KD45.*

Proof. Let $F = \langle W, R \rangle$ satisfy KD45. Without loss of generality, we may suppose that there are w_{INI} and $\text{FIN} \subseteq W$ such that $W \setminus \text{FIN} = \{w_{\text{INI}}\}$; yRy' for all $y, y' \in \text{FIN}$; and $w_{\text{INI}}Ry \wedge \neg yRw_{\text{INI}}$ for every $y \in \text{FIN}$.

Let $M = \langle W, R, V \rangle$ be a model on the frame F . As M restricted to FIN is an equivalence relation, $\|\varphi^2(\top)\| \cap \text{FIN} = \|\varphi^3(\top)\| \cap \text{FIN}$. If $w_{\text{INI}} \in \|\varphi^2(\top)\|$ and $w_{\text{INI}} \notin \|\mu X.\varphi\|$ then $w_{\text{INI}} \notin \|\varphi^3(\top)\|$ must hold. Therefore, $\|\nu X.\varphi\| = \|\varphi^3(\top)\|$. $\square$

Note that we could also use the same technique to show the collapse over S4.3.2 and S4.4, but we would get $\|\varphi^3(\top)\| = \|\varphi^4(\top)\|$; a slightly worse upper bound on the speed of the collapse.

6 Epistemic Logic and Ignorance

In this section, we analyze the meaning of some μ -formulas from the point of view of Epistemic Logic. The formulas we consider describe degrees of ignorance. Building on work of Stalnaker [28], we argue that logics “closer” to S4.2 allow greater degrees of ignorance compared to logics “closer” to S5.

Write K for $\Box$ and $\hat{K}$ for $\Diamond$, and define $B\varphi := \hat{K}K\varphi$. We will work with only one agent, so we read $K\varphi$ as “the agent knows that φ ”, $\hat{K}\varphi$ as “the agent considers φ (epistemically) possible” and $B\varphi$ as “the agent believes that φ ”.

Following Stalnaker [28], we suppose that K satisfies S4.2, B satisfies KD45, and that by belief we mean strong belief:

- $K\varphi \rightarrow B\varphi$;
- $B\varphi \rightarrow KB\varphi$; and
- $\neg B\varphi \rightarrow K\neg B\varphi$.

We also have that if K and B satisfy these properties then $B\varphi$ is equivalent to $\hat{K}K\varphi$, so we can use either basic set of assumptions. A proof of this result can be found in [4].

Van der Hoek and Lomuscio [30] defined the ignorance modality:

$$I\varphi := \leftrightarrow \neg K\varphi \wedge \neg K\neg\varphi.$$

We read $I\varphi$ as “the agent is ignorant *whether* φ holds”. Fine [16] studied high-order versions of the ignorance modality: $I^1\varphi := I\varphi$, $I^{n+1}\varphi := I(I^n\varphi)$. He showed that $\neg KI^n\varphi$ is valid on all S4 frames for any formula φ and $n \geq 2$. One can also think about the ignorance modality $I\varphi$ as the epistemic version of the contingency modality $\nabla\varphi := \neg\Box\varphi \wedge \neg\Box\neg\varphi$. See Fan *et al.* [15] for the connection between ignorance and contingency.

Ignorance is a traditional concept in epistemology, already discussed by Plato in his *Theaetetus*. It is also an active research topic within epistemic logic [10, 14, 26]; Peels and Blaauw [25] compile many recent papers on ignorance.

Fix a μ -sentence φ , we define formulas α_φ^n for $n \in \omega \cup \{\infty\}$. We read α_φ^n as “the agent has n -th degree ignorance whether φ ”. Define:

- $\alpha_\varphi(X) := \hat{K}(\varphi \wedge X) \wedge \hat{K}(\neg\varphi \wedge X)$;
- $\alpha_\varphi^1 := \alpha_\varphi(\top)$;
- $\alpha_\varphi^{n+1} := \alpha_\varphi(\alpha_\varphi^n)$; and
- $\alpha_\varphi^\infty := \nu X. \alpha_\varphi(X)$.

The monotonicity of α_φ implies that any degree of ignorance implies the weaker degrees of ignorance. That is, if $i \leq j$, then $\alpha_\varphi^j \rightarrow \alpha_\varphi^i$. In particular, α_φ^1 is equivalent to $I\varphi$, so we may say that our α_φ^i are refinements of van der Hoek and Lomuscio's ignorance modality.

We briefly review the systems of epistemic logic we are considering before analyzing the degrees of ignorance. For more details, see Stalnaker [28]. S4.2 is Stalnaker's [28] and Lenzen's [20] logic of knowledge. S4.3 is Lehrer and Paxson's [19] logic of knowledge, with knowledge being undefeated justified true belief. S4.3.2 appear in non-monotonic contexts, see Schwarz and Truszczyński [27]. S4.4 is the logic as knowledge as true belief. S5 is the standard logic for multi-agent epistemic logic.

Over S4.2 and S4.3, the formulas

$$\alpha_\varphi^\infty, \alpha_\varphi^1, \alpha_\varphi^2, \alpha_\varphi^3, \dots$$

are pairwise disjoint. We now look at formulas of the form $\alpha_\varphi^n \wedge \neg\alpha_\varphi^{n+1}$. The formula $\alpha_\varphi^1 \wedge \neg\alpha_\varphi^2$ is equivalent to the agent having a false belief and, in all accessible worlds, believing that they know φ . Similarly, $\alpha_\varphi^2 \wedge \neg\alpha_\varphi^3$ holds when the agent has a true belief but thinks it is possible to have a false belief. We can generalize this to other $\alpha_\varphi^i \wedge \neg\alpha_\varphi^{i+1}$, and express higher degrees of self-doubt.

When the alternation hierarchy collapses to modal logic, we do not have infinitely many degrees: α_φ^∞ is equivalent to a modal formula. Over S4.3.2 and S4.4, α_φ^2 is equivalent to α_φ^∞ , so we have only two degrees of ignorance: α_φ^1 , the agent has a false belief; and α_φ^2 , the agent has no belief. In S5, α_φ^1 is equivalent to α_φ^∞ and we have only one degree of ignorance; here, $B\varphi$ is equivalent to $K\varphi$ and the agent cannot have wrong beliefs.

In settings other than S5, having no belief implies a high degree of ignorance, but it is possible to have a high degree of ignorance and a belief at the same time, that is: for any $1 \leq n < \infty$, $\neg(B\varphi \vee B\neg\varphi)$ implies α_φ^∞ ; and $\alpha_\varphi^\infty \wedge B\varphi$ and $\alpha_\varphi^\infty \wedge B\neg\varphi$ are satisfiable. In S5, ignorance and lack of belief are equivalent.

Stalnaker's [28] criticism of S4.3 and S4.3.2 is that in both systems false belief can deny knowledge the agent may be justified in having: in S4.3, a false belief denies *some* knowledge; in S4.3.2, a false belief denies *all* non-trivial knowledge. From our point of view, S4.3.2 allows us to express only a few degrees of ignorance, while S4.3 allows us to express infinitely many degrees.

One can also do a similar analysis to belief and doubt. Olsson and Proietti [24] defined a doubt modality D by $D\varphi := \neg B\varphi \wedge \neg B\neg\varphi$. We can then define degrees of doubt by substituting belief for knowledge in the definition of degrees of ignorance. Formally, fix φ and define:

- $\delta_\varphi(X) := \hat{B}(\varphi \wedge X) \wedge \hat{B}(\neg\varphi \wedge X)$;
- $\delta_\varphi^1 := \delta_\varphi(\top)$;

- $\delta_\varphi^{n+1} := \delta_\varphi(\delta_\varphi^n)$; and
- $\delta_\varphi^\infty := \nu X. \delta_\varphi(X)$.

A short argument shows that δ_φ^1 and δ_φ^∞ are equivalent over frames of KD45. Therefore we can only define one degree of doubt in our framework.

Our analysis of degrees of ignorance and disbelief contrasts with existing discussions of degrees of knowledge and belief. Olsson and Proietti [24] argue that belief and doubt have many degrees but knowledge has only one. They also argue that ignorance has only one degree. Hetherington [18] argues that knowledge also has many degrees.

References

1. Alberucci, L., Facchini, A.: The modal μ -calculus hierarchy over restricted classes of transition systems. *J. Symbolic Logic* **74**(4), 1367–1400 (2009). <https://doi.org/10.2178/jsl/1254748696>
2. Alberucci, L., Facchini, A.: On modal μ -calculus and gödel-löb logic. *Stud. Logica* **91**(2), 145–169 (2009). <https://doi.org/10.1007/s11225-009-9170-9>
3. Arnold, A.: The μ -calculus alternation-depth hierarchy is strict on binary trees. *RAIRO-Theor. Inform. Appl.* **33**(4–5), 329–339 (1999). <https://doi.org/10.1051/ita:1999121>
4. Aucher, G.: Principles of knowledge, belief and conditional belief. In: Rebuschi, M., Batt, M., Heinzmann, G., Lihoreau, F., Musiol, M., Trognon, A. (eds.) *Interdisciplinary Works in Logic, Epistemology, Psychology and Linguistics. LAR*, vol. 3, pp. 97–134. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-03044-9_5
5. Baltag, A., Bezhanishvili, N., Fernández-Duque, D.: The topological mu-calculus: completeness and decidability. In: 2021 36th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), pp. 1–13 (2021). <https://doi.org/10.1109/lics52264.2021.9470560>
6. Blackburn, P., De Rijke, M., Venema, Y.: *Modal Logic*. Cambridge University Press (2001). cbo9781107050884
7. Bradfield, J.C.: Simplifying the modal mu-calculus alternation hierarchy. In: *Annual Symposium on Theoretical Aspects of Computer Science*, pp. 39–49. Springer (1998). <https://doi.org/10.1007/bfb0028547>
8. Bradfield, J.C.: The modal mu-calculus alternation hierarchy is strict. *Theoret. Comput. Sci.* **195**(2), 133–153 (1998). [https://doi.org/10.1016/s0304-3975\(97\)00217-x](https://doi.org/10.1016/s0304-3975(97)00217-x)
9. Bradfield, J., Walukiewicz, I.: The μ -calculus and model checking. In: *Handbook of Model Checking*, pp. 871–919. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-10575-8_26
10. Carrara, M., Chiffi, D., De Florio, C., Pietarinen, A.: We don’t know we don’t know: asserting ignorance. *Synthese* **198**(4), 3565–3580 (2021). <https://doi.org/10.1007/s11229-019-02300-y>
11. D’Agostino, G., Lenzi, G.: On the μ -calculus over transitive and finite transitive frames. *Theoret. Comput. Sci.* **411**(50), 4273–4290 (2010). <https://doi.org/10.1016/j.tcs.2010.09.002>
12. D’Agostino, G., Lenzi, G.: On modal μ -calculus over reflexive symmetric graphs. *J. Log. Comput.* **23**(3), 445–455 (2012). <https://doi.org/10.1093/logcom/exs028>

13. Esakia, L.: Intuitionistic logic and modality via topology. *Ann. Pure Appl. Logic* **127**(1–3), 155–170 (2004). <https://doi.org/10.1016/j.apal.2003.11.013>
14. Fan, J.: A logic for disjunctive ignorance. *J. Philos. Log.* **50**(6), 1293–1312 (2021). <https://doi.org/10.1007/s10992-021-09599-4>
15. Fan, J., Wang, Y., van Ditmarsch, H.: Contingency and knowing whether. *Rev. Symbolic Logic* **8**(1), 75–107 (2015). <https://doi.org/10.1017/S1755020314000343>
16. Fine, K.: Ignorance of ignorance. *Synthese* **195**(9), 4031–4045 (2017). <https://doi.org/10.1007/s11229-017-1406-z>
17. Grädel, E., Thomas, W., Wilke, T. (eds.): *Automata Logics, and Infinite Games*. LNCS, vol. 2500. Springer, Heidelberg (2002). <https://doi.org/10.1007/3-540-36387-4>
18. Hetherington, S.: *Good knowledge, bad knowledge: on two dogmas of epistemology*. Clarendon Press (2001). <https://doi.org/10.1093/acprof:oso/9780199247349.001.0001>
19. Lehrer, K., Paxson, T.: Knowledge: undefeated justified true belief. *J. Philos.* **66**(8), 225–237 (1969). <https://doi.org/10.2307/2024435>
20. Lenzen, W.: Recent work in epistemic logic. *Acta Philos. Fennica* **30**, 1–219 (1978)
21. Lenzi, G.: A hierarchy theorem for the μ -calculus. In: Meyer, F., Monien, B. (eds.) *ICALP 1996*. LNCS, vol. 1099, pp. 87–97. Springer, Heidelberg (1996). https://doi.org/10.1007/3-540-61440-0_119
22. Lenzi, G.: Recent results on the modal μ -calculus: a survey. *Rendiconti dell'Istituto di Matematica dell'Università di Trieste* **42**, 235–255 (2010)
23. McKinsey, J.C.C., Tarski, A.: The algebra of topology. *Ann. Math.* 141–191 (1944). <https://doi.org/10.2307/1969080>
24. Olsson, E. J., Proietti, C.: Explicating ignorance and doubt: a possible worlds approach. In: *The Epistemic Dimensions of Ignorance*, pp. 81–95. Cambridge University Press (2016). <https://doi.org/10.1017/9780511820076.005>
25. Peels, R., Blaauw, M.: *The Epistemic Dimensions of Ignorance*. Cambridge University Press, Cambridge (2016). <https://doi.org/10.1017/9780511820076>
26. Ranalli, C., van Woudenberg, R.: Collective ignorance: an information theoretic account. *Synthese* **198**(5), 4731–4750 (2019). <https://doi.org/10.1007/s11229-019-02367-7>
27. Schwarz, G., Truszczyński, M.: Modal logic $s4f$ and the minimal knowledge paradigm. In: *Proceedings of the 4th Conference on Theoretical Aspects of Reasoning about Knowledge*, pp. 184–198. Morgan Kaufmann Publishers Inc. (1992). <https://doi.org/10.5555/1029762.1029779>
28. Stalnaker, R.: On logics of knowledge and belief. *Philos. Stud. Int. J. Philos. Anal. Tradit.* **128**(1), 169–199 (2006). <https://doi.org/10.1007/s11098-005-4062-y>
29. Van Benthem, J., Bezhanishvili, G.: Modal Logics of Space. In: *Handbook of spatial logics*, pp. 217–298. Springer (2007). https://doi.org/10.1007/978-1-4020-5587-4_5
30. Van Der Hoek, W., Lomuscio, A.: A logic for ignorance. *Electron. Notes Theor. Comput. Sci.* **85**(2), 117–133 (2004). https://doi.org/10.1007/978-3-540-25932-9_6



Embedding Kozen-Tiuryn Logic into Residuated One-Sorted Kleene Algebra with Tests

Igor Sedlár^(✉)  and Johann J. Wannenburg 

Czech Academy of Sciences, Institute of Computer Science, Prague, Czech Republic
{sedlar,wannenburg}@cs.cas.cz

Abstract. Kozen and Tiuryn have introduced the substructural logic S for reasoning about correctness of while programs (ACM TOCL, 2003). The logic S distinguishes between tests and partial correctness assertions, representing the latter by special implicational formulas. Kozen and Tiuryn’s logic extends Kleene algebra with tests, where partial correctness assertions are represented by equations, not terms. Kleene algebra with codomain, KAC, is a one-sorted alternative to Kleene algebra with tests that expands Kleene algebra with an operator that allows to construct a Boolean subalgebra of tests. In this paper we show that Kozen and Tiuryn’s logic embeds into the equational theory of the expansion of KAC with residuals of Kleene algebra multiplication and the upper adjoint of the codomain operator.

Keywords: Hoare logic · Kleene algebra with codomain · Kleene algebra with tests · Partial correctness · Substructural logic

1 Introduction

Kleene algebra with tests [10], KAT, is a simple algebraic framework for verifying properties of propositional while programs. KAT is two-sorted, featuring a Boolean algebra of tests embedded into a Kleene algebra of programs. The equational theory of KAT is PSPACE-complete [2] and KAT subsumes Propositional Hoare logic, PHL [11]. A PHL partial correctness assertion $\{b\}p\{c\}$, meaning that c holds after each terminating execution of program p in a state satisfying b , is represented in KAT by the equation $bp\bar{c} = 0$ or, equivalently, $bp = bpc$. The universal Horn theory (or the quasi-equational theory) of KAT consisting of quasi-equations of the form $E \longrightarrow p = q$ where E is a finite set of equations of the form $r = 0$ embeds into the equational theory of KAT [13] and is therefore PSPACE-complete as well.

Kozen and Tiuryn [14] introduce a substructural logic S that extends KAT and represents partial correctness assertions as implicational formulas $bp \Rightarrow c$. They argue that the implicational rendering of partial correctness assertions has certain advantages over the equational one, e.g. it facilitates a better distinction

between local and global properties. PHL embeds into $\mathbf{S}$ and $\mathbf{S}$ is PSPACE-complete [12]. Kozen and Tiuryn prove that $\mathbf{S}$ induces a left-handed KAT structure on programs. However, a more thorough discussion of the relation of $\mathbf{S}$ to residuated extensions of Kleene algebra, such as Pratt's action logic [18], or to substructural logics in general is not provided. We believe that a deeper investigation of these connections would shed more light on the landscape of program logics and algebras.

In this paper we prove an embedding result that contributes to this. We note that a simple extension of KAT with a residual $\rightarrow$ of the Kleene algebra multiplication does not enable an obvious embedding of $\mathbf{S}$: implication formulas of $\mathbf{S}$ are test-like in the sense that they entail the multiplicative unit 1, but terms $p \rightarrow b$ of residuated KAT are not test-like, even if b is. To avoid this problem, we work with Kleene algebra with codomain, KAC [3, 5], a one-sorted alternative to KAT featuring a test-forming codomain operator c . In particular, we introduce an extension of KAC with both residuals of Kleene algebra multiplication and the upper adjoint e of the codomain operator, which we call SKAT. We show that $c(p \rightarrow e(b))$ behaves like $p \Rightarrow b$ of $\mathbf{S}$. Our main technical result is that $\mathbf{S}$ embeds into the equational theory of SKAT*, the class of algebras in SKAT that are based on *-continuous Kleene algebras.

The paper is structured as follows. Section 2 recalls KAT and $\mathbf{S}$. Section 3 introduces a residuated version of KAT. We show that a naive translation of the implication connective of $\mathbf{S}$ into the residual operation does not constitute an embedding. In Sect. 4 we recall KAC and in Sect. 5 we introduce SKAT and SKAT*. Section 6 establishes the embedding result. Section 7 concludes the paper and lists some interesting open problems.

2 KAT and $\mathbf{S}$

In this section we recall KAT (Sect. 2.1) and we outline $\mathbf{S}$ (Sect. 2.2).

2.1 Kleene Algebra with Tests

We assume that the reader is familiar with the notion of an *idempotent semiring*.

Definition 1. A Kleene algebra [9] is an idempotent semiring $(K, \cdot, +, 1, 0)$ expanded with an operation $*$: $K \rightarrow K$ such that

$$1 + xx^* \leq x^* \tag{1}$$

$$1 + x^*x \leq x^* \tag{2}$$

$$y + xz \leq z \implies x^*y \leq z \tag{3}$$

$$y + xz \leq z \implies yx^* \leq z \tag{4}$$

A Kleene algebra is *-continuous iff it satisfies the condition

$$xy^*z = \sum_{n \geq 0} xy^n z \tag{5}$$

where $y^0 = 1$ and $y^{n+1} = y^n y$. $\sum X$ is the supremum of the set $X \subseteq K$ with respect to the partial order induced by Kleene algebra addition $+$.

It follows from the definition that x^* is the least element z such that $1 \leq z$, $xz \leq z$ and $zx \leq z$. A standard example of a Kleene algebra is a relational Kleene algebra where K is a set of binary relations over some set S , $\cdot$ is relational composition, $+$ is set union, $*$ is reflexive transitive closure, 1 is identity on S and 0 is the empty set. A relational Kleene algebra where K is the power set of some $S \times S$ will be called *fully relational*. Another standard example is the Kleene algebra of regular languages over a finite alphabet, where $*$ corresponds to finite iteration (Kleene star). Relational Kleene algebras and Kleene algebras of regular languages are both $*$ -continuous. Part of the definition of $*$ -continuous Kleene algebras is the requirement that suprema of sets of elements of the form $xy^n z$ exist. Not each Kleene algebra is $*$ -continuous [8], but the equational theories of Kleene algebras and $*$ -continuous Kleene algebras coincide [9].

Definition 2. A Kleene algebra with tests [10] is a structure $(K, B, \cdot, +, *, 1, 0, \bar{\cdot})$ such that

- $(K, \cdot, +, *, 1, 0)$ is a Kleene algebra,
- $B \subseteq K$, and
- $(B, \cdot, +, \bar{\cdot}, 1, 0)$ is a Boolean algebra.

A Kleene algebra with tests is $*$ -continuous if its underlying Kleene algebra is.

Every Kleene algebra is a Kleene algebra with tests; take $B = \{0, 1\}$ and define $\bar{0} = 1$, $\bar{1} = 0$ and $\bar{x} = x$ for $x \notin \{1, 0\}$ ¹. A standard example of a ($*$ -continuous) Kleene algebra with tests is a relational Kleene algebra expanded with a Boolean subalgebra of the *negative cone*, i.e. the elements $x \leq 1$, which in the relational case are subsets of the identity relation. The class of Kleene algebras with tests is denoted as KAT and the class of $*$ -continuous Kleene algebras with tests is denoted as KAT*. As in the test-free case, the equational theory of KAT is identical to the equational theory of KAT* [13].

Kleene algebras with tests are able to represent while programs and facilitate equational reasoning about their partial correctness:

- **skip** = 1
- $p; q = pq$
- **if** b **then** p **else** $q = (bp) + (\bar{b}q)$
- **while** b **do** $p = (bp)^* b$
- $\{b\} p \{c\}$ corresponds to $bp\bar{c} = 0$

¹ In contrast to the standard formulation of Kleene algebra with tests [10], we consider $\bar{\cdot}$ to be defined on all elements of K .

2.2 Substructural Logic of Partial Correctness

This section outlines the substructural logic S [14] that allows to represent a partial correctness assertion by a formula (term) instead of an equation as in KAT. As with KAT, the logic S is many-sorted. Let $\mathsf{B} = \{\mathsf{b}_i \mid i \in \omega\}$ be the set of test variables and let $\mathsf{P} = \{\mathsf{p}_i \mid i \in \omega\}$ be the set of program variables. We define the following sorts of syntactic objects:

tests	$b, c := \mathsf{b}_i \mid 0 \mid b \Rightarrow c$
programs	$p, q := \mathsf{p}_i \mid b \mid p \oplus q \mid p \otimes q \mid p^+$
formulas	$f, g := b \mid p \Rightarrow f$
environments	$\Gamma, \Delta := \epsilon \mid \Gamma, p \mid \Gamma, f$
sequents	$\Gamma \vdash f$

We define $1 := 0 \Rightarrow 0$, $\neg b := b \Rightarrow 0$ and $p^* := 1 \oplus p^+$. We will sometimes write pq instead of $p \otimes q$ and $\bar{b}$ instead of $\neg b$. Let $\mathsf{E} = \mathsf{B} \cup \mathsf{P}$ and let Ex_{S} , the set of S -expressions, be the union of the sets of formulas, programs and environments.

Kozen and Tiuryn introduce three kinds of semantics for their language: semantics based on guarded strings, traces and binary relations, respectively. We will work only with binary relational semantics.

Definition 3. A Kozen–Tiuryn model is a pair $M = (W, V)$, where W is a non-empty set and $V : \mathsf{E} \rightarrow 2^{W \times W}$ such that $V(\mathsf{b}) \subseteq \text{id}_W$ for all $\mathsf{b} \in \mathsf{B}$.

For each Kozen–Tiuryn model M , we define the M -interpretation function $[\]_M : Ex_{\mathsf{S}} \rightarrow 2^{W \times W}$ as follows:

- $[\mathsf{b}]_M = V(\mathsf{b})$
- $[\mathsf{p}]_M = V(\mathsf{p})$
- $[0]_M = \emptyset$
- $[b \Rightarrow c]_M = \{(s, s) \mid (s, s) \notin [b]_M \text{ or } (s, s) \in [c]_M\}$
- $[p \oplus q]_M = [p]_M \cup [q]_M$
- $[p \otimes q]_M = [p]_M \circ [q]_M$
- $[p^+]_M = [p]_M^+$
- $[p \Rightarrow f]_M = \{(s, s) \mid \forall t. (s, t) \in [p]_M \implies (t, t) \in [f]_M\}$
- $[\epsilon]_M = \text{id}_W$
- $[\Gamma, \Delta]_M = [\Gamma]_M \circ [\Delta]_M$

(Here $^+$ denotes transitive closure and $\circ$ denotes relational composition.)

A sequent $\Gamma \vdash f$ is valid in M iff, for all $s, t \in W$, if $(s, t) \in [\Gamma]_M$, then $(t, t) \in [f]_M$.

Observe that $[f]_M \subseteq \text{id}_W$ for all formulas f ; if $(s, s) \in [f]_M$, then we may say that formula f is true in s . Note that $[bp \Rightarrow c]_M$ is the set of (s, s) such that, for all t , if $(s, s) \in [b]_M$ and $(s, t) \in [p]_M$, then $(t, t) \in [c]_M$. Hence, $bp \Rightarrow c$ represents a partial correctness assertion: the formula is true in s iff b is true in s and p connects s with a state t only if c is true in t .

Figure 1 shows the sequent proof system for S. A sequent $\Gamma \vdash f$ is *provable in S* iff there is a finite sequence of sequents that ends with $\Gamma \vdash f$ each of which is either of the form (Id) or (I0), or is derived from previous sequents using some of the inference rules.

Theorem 1 (Kozen and Tiuryn). $\Gamma \vdash f$ is provable in S iff $\Gamma \vdash f$ is valid in all models M.

$\text{(Id)} \quad b \vdash b$	$\text{(I0)} \quad \Gamma, 0, \Delta \vdash f$
$\text{(TC)} \quad \frac{\Gamma, b, \Delta \vdash f \quad \Gamma, \bar{b}, \Delta \vdash f}{\Gamma, \Delta \vdash f}$	$\text{(cut)} \quad \frac{\Gamma \vdash g \quad \Gamma, g, \Delta \vdash f}{\Gamma, \Delta \vdash f}$
$\text{(R}\Rightarrow\text{)} \quad \frac{\Gamma, p \vdash f}{\Gamma \vdash p \Rightarrow f}$	$\text{(I}\Rightarrow\text{)} \quad \frac{\Gamma, p, f, \Delta \vdash g}{\Gamma, p \Rightarrow f, p, \Delta \vdash g}$
$\text{(I}\otimes\text{)} \quad \frac{\Gamma, p, q, \Delta \vdash f}{\Gamma, p \otimes q, \Delta \vdash f}$	$\text{(E}\otimes\text{)} \quad \frac{\Gamma, p \otimes q, \Delta \vdash f}{\Gamma, p, q, \Delta \vdash f}$
$\text{(I}\oplus\text{)} \quad \frac{\Gamma, p, \Delta \vdash f \quad \Gamma, q, \Delta \vdash f}{\Gamma, p \oplus q, \Delta \vdash f}$	$\text{(E}\oplus_1\text{)} \quad \frac{\Gamma, p \oplus q, \Delta \vdash f}{\Gamma, p, \Delta \vdash f}$
$\text{(I}^+\text{)} \quad \frac{g, p \vdash f \quad g, p \vdash g}{g, p^+ \vdash f}$	$\text{(E}\oplus_2\text{)} \quad \frac{\Gamma, p \oplus q, \Delta \vdash f}{\Gamma, q, \Delta \vdash f}$
$\text{(E}^+\text{)} \quad \frac{\Gamma, p^+, \Delta \vdash f}{\Gamma, p, \Delta \vdash f}$	$\text{(Wf)} \quad \frac{\Gamma, \Delta \vdash g}{\Gamma, f, \Delta \vdash g}$
$\text{(CC}^+\text{)} \quad \frac{\Gamma, p^+, \Delta \vdash f}{\Gamma, p^+, p^+, \Delta \vdash f}$	$\text{(Wp)} \quad \frac{\Gamma \vdash f}{p, \Gamma \vdash f}$

Fig. 1. The sequent proof system for S.

S can be seen as a substructural logic [19]. From that point of view, S is an extension of the Multiplicative-additive Lambek calculus with the transitive closure operator $^+$. However, S contains some rules not usual from the substructural logic perspective, namely, the sort-specific weakening rules (Wf) and (Wp), and the implication-formula rules (TC) and (I $\Rightarrow$). It is therefore interesting to inquire how S relates to mainstream substructural logics. The embedding result of Sect. 6 is helpful in this respect.

3 Residuated KAT

The crucial difference between KAT and S is that the latter contains an implication operator $\Rightarrow$ that is used to formalize partial correctness assertions as formulas (terms) of the language. Therefore, one may wonder if S can be thought of as KAT expanded with a residual of Kleene algebra multiplication. In this section we introduce residuated KAT, RKAT, and we show that a naive mapping from S to RKAT is not an embedding.

We note that residuated Kleene algebras with tests were studied also in [6]. In that framework, however, it is assumed by definition that the negative cone of the underlying Kleene algebra forms a Boolean algebra of tests. Kozen [10] argues, however, that this approach is not entirely satisfactory: tests are usually easily decidable (unlike other propositions, such as halting assertions), and not each Kleene algebra has a Boolean negative cone. Our definition of residuated Kleene algebras with tests adds residuals to KAT defined as usual.

Definition 4. A residuated idempotent semiring is a structure of the form $(K, \cdot, +, \rightarrow, \hookrightarrow, 1, 0)$ where

- $(K, \cdot, +, 1, 0)$ is an idempotent semiring, and
- $\rightarrow, \hookrightarrow$ are binary operations on K satisfying the residuation laws

$$xy \leq z \iff x \leq y \rightarrow z \iff y \leq x \hookrightarrow z \quad (6)$$

In residuated idempotent semirings, $\rightarrow$ is called the *right residual* of $\cdot$ and $\hookrightarrow$ is called the *left residual*. An idempotent semiring is called *right-residuated* (*left-residuated*) if it is a reduct of a residuated idempotent semiring with only the right (left) residual. Note that residuated idempotent semirings necessarily have a top element: $x0 \leq 0$ implies $x \leq 0 \rightarrow 0$ and $0x \leq 0$ implies $x \leq 0 \hookrightarrow 0$.

The study of residuated semirings has some precedent. Pratt adds residuals to Kleene algebras to obtain *action algebras* [18], and Kozen adds a lattice meet operation to action algebras to obtain *action lattices* [7]; see also [6]. Equational theories of both action algebras and action lattices are undecidable [15].

Definition 5. A right-residuated Kleene algebra with tests is a structure of the form $(K, B, \cdot, +, *, \rightarrow, \bar{\cdot}, 1, 0)$ where

- $(K, B, \cdot, +, *, \bar{\cdot}, 1, 0) \in \text{KAT}$, and
- $(K, \cdot, +, \rightarrow, 1, 0)$ is a right-residuated idempotent semiring.

We will usually call right-residuated Kleene algebras with tests just residuated. The class of residuated Kleene algebras with tests will be denoted as RKAT.

One may be tempted to define $\bar{x}$ as $x \rightarrow 0$, but this temptation is better resisted. Note that $0 \in B$ in every Kleene algebra with tests, but $0 \rightarrow 0$ is the top element. Hence, if we have a residuated Kleene algebra with tests where B does not contain the top element, then B is not closed under the operation $x \rightarrow 0$. An example of such an algebra is given in Fig. 2, where $B = \{0, 1\}$ and $+$ is join behaving as indicated in the diagram on the left: $0 \rightarrow 0 = \top$ but $\bar{0} = 1$.

For a similar reason, formulas $p \Rightarrow b$ of $\mathbf{S}$ do not directly translate into $p \rightarrow b$; the former entail 1, but the latter do not, as we have seen. However, a consideration of the relational case sheds some light on the matter. Recall that for $R \subseteq S \times S$ and $B \subseteq \text{id}_S$,

$$\begin{aligned} R \rightarrow B &= \{(s, u) \mid \forall v. (u, v) \in R \implies (s, v) \in B\} \\ R \Rightarrow B &= \{(u, u) \mid \forall v. (u, v) \in R \implies (v, v) \in B\} \end{aligned}$$

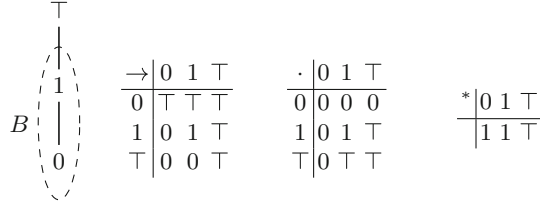


Fig. 2. A residuated Kleene algebra with tests where $x \rightarrow 0$ is not test complementation.

Now consider two operations $c, e : 2^{S \times S} \rightarrow 2^{S \times S}$ such that

$$c(R) = \{(u, u) \mid \exists s. (s, u) \in R\} \quad (7)$$

$$e(R) = \{(s, u) \mid (u, u) \in R\} \quad (8)$$

Note that c and e form a *Galois connection* in the sense that, for all binary relations Q and R on S ,

$$c(R) \subseteq Q \iff R \subseteq e(Q). \quad (9)$$

We have the following observation.

Proposition 1. *For all $R \subseteq S \times S$ and $B \subseteq \text{id}_S$,*

$$c(R \rightarrow e(B)) = R \Rightarrow B \quad (10)$$

Proof. If $(u, u) \in R \Rightarrow B$, then $(u, u) \in R \rightarrow e(B)$ since $(v, v) \in B$ iff $(u, v) \in e(B)$ by (8). But then obviously $(u, u) \in c(R \rightarrow e(B))$. Hence, $R \Rightarrow B \subseteq c(R \rightarrow e(B))$. Conversely, assume that $(u, u) \in c(R \rightarrow e(B))$ and $(u, v) \in R$, then $(s, u) \in R \rightarrow e(B)$ by (7) and so $(s, v) \in e(B)$. It follows that $(v, v) \in B$. Hence, $(u, u) \in R \Rightarrow B$ and so $c(R \rightarrow e(B)) \subseteq R \Rightarrow B$. $\square$

Hence, it seems that expanding RKAT with abstractions of c and e would make it possible to express the implication operator of S. An abstraction of c was added to semirings and Kleene algebras in the works on Kleene algebra with (co)domain [3, 5]. We discuss Kleene algebra with codomain in the next section.

4 Kleene Algebra with Codomain

Kleene algebra with codomain, KAC, is a one-sorted alternative to KAT: instead of assuming the existence of a Boolean subalgebra of tests, KAC introduces a unary *codomain* operation c such that the set of $\mathsf{c}(x)$ for x in the underlying Kleene algebra forms a Boolean algebra. One of the motivations for studying one-sorted alternatives to Kleene algebra with tests is that they seem to be better suited to automated theorem proving (see [5], p. 194). Our presentation follows [3, 5], where a symmetric variant of KAC, namely, Kleene algebra with *domain*, was studied.

Kleene algebras with codomain add to Kleene algebras a unary operator that generalizes the *anticodomain* operator on binary relations:

$$a(R) = \{(u, u) \mid \neg \exists s. (s, u) \in R\}. \quad (11)$$

Note that, assuming (7), $c(R) = a(a(R))$.

Definition 6. A Kleene algebra with codomain is an algebra of the form $(K, \cdot, +, *, a, 1, 0)$ where $a : K \rightarrow K$ such that

$$x \cdot a(x) \leq 0 \quad (12)$$

$$a(x \cdot y) \leq a(a(x) \cdot y) \quad (13)$$

$$a(x) + a(a(x)) = 1 \quad (14)$$

We define $c(x) := a(a(x))$.

Note that the relational anticodomain operator (11) satisfies the above equations.

Proposition 2. In every Kleene algebra with codomain, the operator c satisfies the following codomain equations:

$$x \leq x \cdot c(x) \quad (15)$$

$$c(xy) = c(c(x) \cdot y) \quad (16)$$

$$c(x) \leq 1 \quad (17)$$

$$c(0) = 0 \quad (18)$$

$$c(x + y) = c(x) + c(y) \quad (19)$$

Moreover, c has the following properties:

$$c(c(x)) = c(x) \quad (20)$$

$$c(x) \cdot c(x) = c(x) \quad (21)$$

$$c(xy) \leq c(y) \quad (22)$$

$$x \leq xc(y) \implies c(x) \leq c(y) \quad (23)$$

Proof. For the codomain equations, see [5], Theorem 8.7, and the more detailed proof of the theorem in Appendix D of [4]. The arguments there are given for the domain and antidomain operators, but similar arguments work for codomain and anticodomain.

The fixpoint property (20) follows from $c(xy) = c(c(x) \cdot y)$ and idempotence (21) follows from the fixpoint property, $x \leq x \cdot c(x)$, and $c(x) \leq 1$. The codomain restriction property (22) is established by noting that $c(xy) \leq c(c(x)y) \leq c(1y) \leq c(y)$. The least preserver property (23) is established by noting that $x \leq xc(y)$ implies $c(x) \leq c(xc(y)) \leq c(c(y)) \leq c(y)$. $\square$

Proposition 3. Let $c(K) = \{x \mid x \in K \ \& \ \exists y. (x = c(y))\}$. In every Kleene algebra with codomain,

1. $(c(K), \cdot, +, 1, 0)$ is a subalgebra of $(K, \cdot, +, 1, 0)$;
2. $(c(K), \cdot, +, 1, 0)$ is a bounded distributive lattice;
3. $c(x) + a(c(x)) = 1$ and $c(x) \cdot a(c(x)) = 0$.

Hence, $c(\mathcal{K}) = (c(K), \cdot, +, a, 1, 0)$ is a Boolean algebra.

Proof. 1. follows from $c(1) = 1$, $c(0) = 0$, $c(x) + c(y) = c(x + y)$, and $c(x) \cdot c(y) = c(c(x) \cdot c(y))$. We verified the latter using Prover 9 [16]. 2. follows from 1., together with $c(x) \leq 1$ and $c(x) = c(x)c(x)$ of Proposition 2. 3. follows from the axioms for a . $\square$

It can be shown that every full relational KAT can be seen as a KAC where a is the relational anticodomain operation, $a(R) := \{(u, u) \mid \neg \exists s. (s, u) \in R\}$. Moreover, Proposition 3 shows that every KAC gives rise to a KAT. A more detailed argument involving a translation from the language of KAT into the language of KAC then shows that the equational theory of KAT embeds into the equational theory of KAC. We omit the details.

5 SKAT: Residuated KAC with a Galois Connection

In this section, we introduce SKAT, an extension of KAC with both residuals of Kleene algebra multiplication and the upper adjoint of the codomain operator. The $*$ -continuous variant of SKAT will be denoted as SKAT*. The next section shows that S embeds into the equational theory of SKAT*.

Definition 7. An S-type Kleene algebra with codomain is a structure of the form $(K, \cdot, +, \rightarrow, \hookrightarrow, *, a, e, 1, 0)$ where

- $(K, \cdot, +, \hookrightarrow, \rightarrow, *, 1, 0)$ is a residuated Kleene algebra,
- $(K, \cdot, +, *, a, 1, 0)$ is a Kleene algebra with codomain, and
- e is a unary operation on K that satisfies the following:

$$a(a(e(x))) \leq x \quad (24)$$

$$x \leq e(a(a(x))) \quad (25)$$

$$e(x) \leq e(x + y) \quad (26)$$

An S-type Kleene algebra with codomain is $*$ -continuous iff its underlying Kleene algebra is $*$ -continuous. The class of S-type Kleene algebras with codomain will be denoted as SKAT and the class of the $*$ -continuous ones as SKAT*.

We will call S-type Kleene algebras with codomain simply SKAT-algebras. We will usually omit bracketing between elements of $\{a, e\}$, writing $ec(x)$ instead of $e(c(x))$, for instance.

Every SKAT-algebra is a Kleene algebra with codomain by definition, and so the test algebra $(c(\mathcal{K}), \cdot, +, a, 1, 0)$ of each SKAT-algebra $\mathcal{K}$ is a Boolean algebra. It is well known that residuated Kleene algebras form a variety [18], and so SKAT forms a variety.

Proposition 4. *The following hold in each SKAT-algebra:*

$$c(x) \leq y \iff x \leq e(y) \quad (27)$$

$$c(x \rightarrow y) \leq x \rightarrow xc(y) \quad (28)$$

$$a(x) = c(c(x) \rightarrow 0) \quad (29)$$

Proof. (27) follows directly from the SKAT axioms concerning e . (28) is established as follows. Take any z such that $c(z) = z$; note that $c(x \rightarrow y)$ is of this form. We reason as follows:

$$\begin{aligned} z \leq c(x \rightarrow y) &\implies zx \leq c(x \rightarrow y)x \stackrel{(16)}{\implies} c(zx) \leq c((x \rightarrow y)x) \\ &\implies c(zx) \leq c(y) \implies zxc(zx) \leq c(z)xc(y) \implies zx \leq xc(y) \\ &\implies z \leq x \rightarrow xc(y) \implies c(z) \leq c(x \rightarrow xc(y)) \implies z \leq c(x \rightarrow xc(y)). \end{aligned}$$

(29) is established as follows. Note that $ac(x) = a(x)$ by (13) and $c(x)a(x) = a(x)c(x)$ by Proposition 3. Hence, $a(x) \leq c(x) \rightarrow 0$ by (12) and residuation. It follows that $ca(x) \leq c(c(x) \rightarrow 0)$ and so $a(x) \leq c(c(x) \rightarrow 0)$.

The converse inequality follows from

$$xc(y) = 0 \implies c(x) \leq a(y) \quad (30)$$

In particular, $c(c(x) \rightarrow 0)c(x) = 0$ by (28), and so $cc(c(x) \rightarrow 0) \leq a(x)$ by (30), which means that $c(c(x) \rightarrow 0) \leq a(x)$ by the fixpoint property (20). The quasi-equation (30) is established similarly as the quasi-equation (26) on p. 386 of [4]. If $xc(y) = 0$, then $a(c(x)c(y)) = 1$ and so $c(x)c(y) = 0$. But then

$$c(x) = c(x)(c(y) + a(y)) = c(x)c(y) + c(x)a(y) = c(x)a(y) \leq a(y).$$

□

We note that an extension of Kleene algebra with domain with a right residual of Kleene algebra multiplication is considered in [4, 5]. However, the goal there is to induce a Heyting algebra of tests, and $\rightarrow$ is assumed to be a partial operation, defined only on the test algebra. It is also assumed that the test algebra is closed under $\rightarrow$, which we may transpose to the setting that uses codomain instead of domain as follows:

$$c(c(x) \rightarrow c(y)) = c(x) \rightarrow c(y) \quad (31)$$

In our setting, $\rightarrow$ is a total operation which is not included in the signature of (Boolean) test algebras. It can also be shown that (31) does not hold; see Fig. 3. Operations $\cdot$ and $*$ are defined as in Fig. 2, $\hookrightarrow$ is identical to $\rightarrow$. It is easily verified that $c(0) \rightarrow c(0) = \top$, and $c(\top) = 1$.

6 The Embedding Result

In this section we prove that S embeds into the equational theory of $SKAT^*$.

$$\begin{array}{c} \top \\ | \\ 1 \\ | \\ 0 \end{array} \quad \rightarrow \begin{array}{c|c} 0 & 1 \ \top \\ \hline 0 & \top \ \top \ \top \\ 1 & 0 \ 1 \ \top \\ \top & 0 \ 0 \ \top \end{array} \quad \begin{array}{c|c} \mathbf{a} & 0 \ 1 \ \top \\ \hline 1 & 0 \ 0 \end{array} \quad \begin{array}{c|c} \mathbf{e} & 0 \ 1 \ \top \\ \hline 0 & \top \ \top \end{array}$$

Fig. 3. A SKAT-algebra falsifying (31).

Let $\mathbf{X} = \{x_i \mid i \in \omega\}$ be a countable set of variables. The set of terms Tm is the set of expressions formed using the following grammar:

$$p, q := x_i \mid 1 \mid 0 \mid p \cdot q \mid p + q \mid p \rightarrow q \mid p^* \mid c(p) \mid e(p).$$

(We abuse the notation by re-using the same variables that range over programs in S, but this will hopefully cause no confusion.) An equation is an ordered pair of terms, denoted as $p \approx q$. We use $p \preceq q$ as short for $p + q \approx q$. If $\mathcal{K} \in \mathbf{SKAT}$, then a $\mathcal{K}$ -valuation is any homomorphism from Tm into $\mathcal{K}$. An equation $p \approx q$ is satisfied by a $\mathcal{K}$ -valuation $[\]$ iff $[p] = [q]$. The equational theory of SKAT (SKAT*) is the set of equations that are satisfied by all $\mathcal{K}$ valuations where $\mathcal{K} \in \mathbf{SKAT}$ ($\mathcal{K} \in \mathbf{SKAT}^*$).

Definition 8. We define $Tr : Ex_S \rightarrow Tm$ as follows:

- $Tr(p_n) = x_{2n}$
- $Tr(b_n) = c(x_{2n+1})$
- $Tr(0) = c(0)$
- $Tr(b \Rightarrow c) = c(Tr(b) \rightarrow e(Tr(c)))$
- $Tr(p \oplus q) = Tr(p) + Tr(q)$
- $Tr(p \otimes q) = Tr(p) \cdot Tr(q)$
- $Tr(p^+) = Tr(p) \cdot Tr(p)^*$
- $Tr(p \Rightarrow f) = c(Tr(p) \rightarrow e(Tr(f)))$
- $Tr(\epsilon) = 1$
- $Tr(\Gamma, \Delta) = Tr(\Gamma) \cdot Tr(\Delta)$

Theorem 2. A sequent $\Gamma \vdash f$ is provable in S iff $c(Tr(\Gamma)) \leq Tr(f)$ belongs to the equational theory of SKAT*.

Theorem 2 is established in two steps. First we show that every model M of S with a valuation $[\]$ can be transformed into a $\mathcal{M} \in \mathbf{SKAT}^*$ and a $\mathcal{M}$ -valuation $\llbracket \]$ such that $[\alpha] = \llbracket Tr(\alpha) \rrbracket$ for all $\alpha \in Ex_S$. Then we show by induction on the length of S-proofs that if $\Gamma \vdash f$ is provable in S, then $c(Tr(\Gamma)) \preceq Tr(f)$ belongs to the equational theory of SKAT*.

Lemma 1. Let $M = (W, V)$ be a model of S and let $[\]$ be a M -interpretation. Then there is $\mathcal{M} \in \mathbf{SKAT}^*$ and a $\mathcal{M}$ -valuation $\llbracket \]$ such that, for all $\alpha \in Ex_S$,

$$[\alpha] = \llbracket Tr(\alpha) \rrbracket \tag{32}$$

Proof. Fix $M = (W, V)$. Let $\mathcal{M}$ be the full relational residuated Kleene algebra of binary relations on W ; define $\mathbf{a}(R) = \{(u, u) \mid \neg \exists s. (s, u) \in R\}$ and $\mathbf{e}(R) = \{(s, u) \mid (u, u) \in R\}$; note that $\mathbf{c}(R) = \{(u, u) \mid \exists s. (s, u) \in R\}$ ². It is easily checked that $\mathcal{M} \in \mathbf{SKAT}^*$. In order to prove (24), note that $(s, t) \in \mathbf{ce}(R)$ only if $s = t$ and there is u such that $(u, t) \in \mathbf{e}(R)$, which holds only if $s = t$ and $(t, t) \in R$. In order to prove (25), note that $(s, t) \in R$ implies $(t, t) \in \mathbf{c}(R)$, which implies $(s, t) \in \mathbf{ec}(R)$. Finally, $\mathbf{e}$ is obviously monotonic, which entails (26).

Define $\llbracket \cdot \rrbracket$ as the unique $\mathbf{SKAT}$ homomorphism from Tm to $\mathcal{M}$ satisfying the following, for all $n \in \omega$:

- $\llbracket \mathbf{x}_{2n} \rrbracket = [\mathbf{p}_n]$;
- $\llbracket \mathbf{x}_{2n+1} \rrbracket = [\mathbf{b}_n]$.

We will prove (32) by induction on α . First we prove that (32) holds for all tests b by induction on b . The base case is established as follows. We know by definition that $\llbracket Tr(\mathbf{b}_n) \rrbracket = \mathbf{c}(\llbracket \mathbf{x}_{2n+1} \rrbracket) = \mathbf{c}[\mathbf{b}_n]$. The latter is $\{(u, u) \mid \exists s. (s, u) \in [\mathbf{b}_n]\} = [\mathbf{b}_n]$, since $[\mathbf{b}_n] \subseteq \text{id}_W$. The induction step is established as follows. $\llbracket Tr(0) \rrbracket = \mathbf{c}[\llbracket 0 \rrbracket] = \emptyset = [0]$ and the other case uses (10) of Proposition 1 (we are working with a relational Kleene algebra):

$$\begin{aligned} \llbracket Tr(b \Rightarrow c) \rrbracket &= \mathbf{c}(\llbracket Tr(b) \rrbracket \rightarrow \mathbf{e}\llbracket Tr(c) \rrbracket) \\ &= \mathbf{c}([b] \rightarrow \mathbf{e}[c]) \\ &\stackrel{(10)}{=} [b \Rightarrow c] \end{aligned}$$

This shows that (32) holds for all tests b . The fact that it holds for all programs is established by easy induction (Tr virtually commutes with Kleene algebra operations). Next, we show that (32) holds for all formulas. The only thing to prove is the induction step for $\alpha = p \Rightarrow f$. The claim can be established using (10) as before. The claim for environments is trivial. $\square$

Lemma 2. *If $\Gamma \vdash f$ is provable in $\mathbf{S}$, then $\mathbf{c}(Tr(\Gamma)) \preceq Tr(f)$ belongs to the equational theory of $\mathbf{SKAT}^*$.*

Proof. We reason by induction on the length of proofs. We note that the proof of this lemma, namely the claim for the (I^+) rule, is the only point where the assumption of $*$ -continuity seems to be required.

A useful fact to note is that $Tr(f)$ is of the form $\mathbf{c}(p)$ for all formulas f . By a translation of a sequent $\Gamma \vdash f$ we mean the equation $\mathbf{c}(Tr(\Gamma)) \preceq Tr(f)$. We will prove that translations of axioms of $\mathbf{S}$ are in the equational theory of $\mathbf{SKAT}^*$, and that if translations of all premises of an inference rule of $\mathbf{S}$ are in the equational theory of $\mathbf{SKAT}^*$, then so is the consequence of the rule. (If $p \preceq q$ is in the equational theory of $\mathbf{SKAT}^*$, then we say that $p \preceq q$ is valid.)

(Id) $\mathbf{c}(Tr(b)) \preceq Tr(b)$ is valid since $\mathbf{cc}(x) \leq \mathbf{c}(x)$ by (20).

² Recall that $R \rightarrow Q = \{(s, u) \mid \forall v. (u, v) \in R \implies (s, u) \in Q\}$ and $R \hookrightarrow Q = \{(s, u) \mid \forall v. (v, s) \in R \implies (v, u) \in Q\}$.

(I0) $c(Tr(\Gamma) \cdot Tr(0) \cdot Tr(\Delta)) \preceq Tr(f)$ is valid since $Tr(0) = c(0) = 0$.

$$(TC) \frac{c(Tr(\Gamma)Tr(b)Tr(\Delta)) \preceq Tr(f) \quad c(Tr(\Gamma)Tr(b \Rightarrow 0)Tr(\Delta)) \preceq Tr(f)}{c(Tr(\Gamma)Tr(\Delta)) \preceq Tr(f)}$$

preserves validity thanks to (19) and (29):

$$\begin{aligned} & c(xc(y)z) \leq a \text{ and } c(xc(c(y) \rightarrow 0)z) \leq a \\ & \stackrel{(19)}{\implies} c(xc(y)z + xc(c(y) \rightarrow 0)z) \leq a \\ & \stackrel{(29)}{\implies} c(xc(y)z + xa(y)z) \leq a \\ & \implies c(x(c(y) + a(y))z) \leq a \\ & \stackrel{(14)}{\implies} c(xz) \leq a \end{aligned}$$

$$(cut) \frac{c(Tr(\Gamma)) \preceq Tr(g) \quad c(Tr(\Gamma)Tr(g)Tr(\Delta)) \preceq Tr(f)}{c(Tr(\Gamma)Tr(\Delta)) \preceq Tr(f)}$$

preserves validity thanks to (15) and (19):

$$\begin{aligned} & c(x) \leq a_2 \text{ and } c(xa_2y) \leq a_1 \\ & \implies xc(x)y \leq xa_2y \text{ and } c(xa_2y) \leq a_1 \\ & \stackrel{(15)}{\implies} xy \leq xa_2y \text{ and } c(xa_2y) \leq a_1 \\ & \stackrel{(19)}{\implies} c(xy) \leq c(xa_2y) \text{ and } c(xa_2y) \leq a_1 \\ & \implies c(xy) \leq a_1 \end{aligned}$$

$$(R \Rightarrow) \frac{c(Tr(\Gamma)Tr(p)) \preceq Tr(f)}{c(Tr(\Gamma)) \preceq Tr(p \Rightarrow f)} \text{ preserves validity thanks to (27) and (19):}$$

$$\begin{aligned} & c(xy) \leq z \stackrel{(27)}{\implies} xy \leq e(z) \\ & \implies x \leq y \rightarrow e(z) \stackrel{(19)}{\implies} c(x) \leq c(y \rightarrow e(z)) \end{aligned}$$

$$(I \Rightarrow) \frac{c(Tr(\Gamma)Tr(p)Tr(f)Tr(\Delta)) \preceq Tr(g)}{c(Tr(\Gamma)Tr(p \Rightarrow f)Tr(p)Tr(\Delta)) \preceq Tr(g)} \text{ preserves validity thanks to (28)}$$

and the fact that (27) entails $cec(y) \leq c(y)$:

$$c(x \rightarrow ec(y))x \leq (x \rightarrow xcec(y))x \leq xcec(y) \leq xc(y)$$

(Recall that $Tr(p \Rightarrow f) = c(q \rightarrow ec(r))$ for some terms q, r .)

(I $\otimes$) follows directly from the definition of Tr , and so do (E $\otimes$), (I $\oplus$), (E $\oplus_1$) and (E $\oplus_2$).

$$(I^+) \frac{c(Tr(g)Tr(p)) \preceq Tr(f) \quad c(Tr(g)Tr(p)) \preceq Tr(g)}{c(Tr(g)Tr(p)Tr(p)^*) \preceq Tr(f)}.$$

It can be proven by induction on n , using (16), that if $c(yx) \leq c(z)$ and $c(yx) \leq y$, then $\forall n \geq 0. c(yxx^n) \leq c(z)$. By (27), $\forall n \geq 0. yxx^n \leq ec(z)$, and so by *-continuity $yx x^* \leq ec(z)$, which means that $c(yxx^*) \leq c(z)$. (Note that *-continuity entails that infinite suprema of elements of the form xy^*z exist.) In fact, this is the only point where we need to assume *-continuity.

(E⁺) follows from the fact that $x \leq xx^*$ in Kleene algebras; and (CC⁺) follows from the fact that $x^*xx^* \leq x^*$.

(Wf) $\frac{c(Tr(\Gamma)Tr(\Delta)) \preceq Tr(g)}{c(Tr(\Gamma)Tr(f)Tr(\Delta)) \preceq Tr(g)}$ preserves validity thanks to $c(x) \leq 1$.

(Wp) $\frac{c(Tr(\Gamma)) \preceq Tr(f)}{c(Tr(p)Tr(\Gamma)) \preceq Tr(f)}$ follows from (22).

This concludes the proof of the lemma. $\square$

Now we are in the position to prove Theorem 2.

Proof. If $\Gamma \vdash f$ is not provable in S , then there is a model M and an interpretation function $[]$ such that $(w, u) \in [\Gamma]$ and $(u, u) \notin [f]$ by Kozen and Tiuryn's relational completeness theorem. By Lemma 1, there is $\mathcal{M} \in \text{SKAT}^*$ and a $\mathcal{M}$ -valuation $\llbracket \cdot \rrbracket$ such that $(u, u) \in \llbracket c(Tr(\Gamma)) \rrbracket$ and $(u, u) \notin \llbracket Tr(f) \rrbracket$. Hence, $c(Tr(\Gamma)) \preceq Tr(f)$ is not in the equational theory of SKAT^* . Conversely, if $\Gamma \vdash f$ is provable in S , then $c(Tr(\Gamma)) \preceq Tr(f)$ is in the equational theory of SKAT^* by Lemma 2. $\square$

7 Conclusion

We have shown in this paper that Kozen and Tiuryn's substructural logic of partial correctness S embeds into the equational theory of an extension of $*$ -continuous Kleene algebra with codomain with both residuals of the Kleene algebra multiplication and the upper adjoint of the codomain operator, SKAT^* . We believe that this result sheds more light on the landscape of program logics and algebras.

A number of interesting problems remain open:

1. Does S embed into the equational theory of SKAT ?
2. Which residuated Kleene algebras extend to algebras in SKAT or SKAT^* ? Is the equational theory of SKAT (SKAT^*) a conservative extension of the equational theory of residuated ($*$ -continuous) Kleene algebras?
3. Related to the previous problem, is the equational theory of SKAT^* identical to the equational theory of SKAT ? The equational theory of $*$ -continuous residuated Kleene algebras is not identical to the equational theory of residuated Kleene algebras [1].
4. What are the free algebras in SKAT and SKAT^* ? McLean [17] shows that regular sets of labelled pointed trees form free Kleene algebras with domain, and a version of his construction may turn out to apply to our case as well.
5. What is the complexity of the equational theories of SKAT and SKAT^* ? The equational theory of $*$ -continuous residuated Kleene algebras is Π_1^0 -complete (hence not r.e.) and the equational theory of residuated Kleene algebras is Σ_1^0 -complete.

6. Consider the set of terms generated by the following restricted grammar:

$$p, q := x_i \mid 1 \mid 0 \mid p \cdot q \mid p + q \mid p^* \mid c(p) \mid c(p \rightarrow ec(q)) .$$

Is the fragment of the equational theory of SKAT restricted to terms generated by this grammar decidable?

7. What are the Gentzen systems sound and (weakly) complete for SKAT, in the style of [6]?

Acknowledgement. We are grateful to two anonymous reviewers for valuable suggestions. The first author is grateful to Vít Punčochář for discussions that led to the work on this paper, and to the audience at the Proof Theory Seminar of the Steklov Mathematical Institute for illuminating discussion after his talk on 14 February 2022. The work of the second author was carried out within the project *Supporting the Internationalization of the Institute of Computer Science of the Czech Academy of Sciences* (no. CZ.02.2.69/0.0/0.0/18_053/0017594), funded by the Operational Programme Research, Development and Education of the Ministry of Education, Youth and Sports of the Czech Republic. The project is co-funded by the EU.

References

1. Buszkowski, W.: On action logic: equational theories of action algebras. *J. Logic Comput.* **17**(1), 199–217 (2006). <https://doi.org/10.1093/logcom/exl036>
2. Cohen, E., Kozen, D., Smith, F.: The complexity of Kleene algebra with tests. Technical report TR96-1598, Computer Science Department, Cornell University (1996)
3. Desharnais, J., Möller, B., Struth, G.: Kleene algebra with domain. *ACM Trans. Comput. Logic* **7**(4), 798–833 (2006). <https://doi.org/10.1145/1183278.1183285>
4. Desharnais, J., Struth, G.: Modal semirings revisited. In: Audebaud, P., Paulin-Mohring, C. (eds.) *MPC 2008*. LNCS, vol. 5133, pp. 360–387. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-70594-9_19
5. Desharnais, J., Struth, G.: Internal axioms for domain semirings. *Sci. Comput. Program.* **76**(3), 181–203 (2011). <https://doi.org/10.1016/j.scico.2010.05.007>. special issue on special issue on the Mathematics of Program Construction (MPC 2008)
6. Jipsen, P.: From semirings to residuated Kleene lattices. *Stud. Logica* **76**(2), 291–303 (2004). <https://doi.org/10.1023/B:STUD.0000032089.54776.63>
7. Kozen, D.: On action algebras. In: *Logic and Information Flow*, pp. 78–88. MIT Press (1994)
8. Kozen, D.: On Kleene algebras and closed semirings. In: Rován, B. (ed.) *MFCS 1990*. LNCS, vol. 452, pp. 26–47. Springer, Heidelberg (1990). <https://doi.org/10.1007/BFb0029594>
9. Kozen, D.: A completeness theorem for Kleene algebras and the algebra of regular events. *Inf. Comput.* **110**(2), 366–390 (1994). <https://doi.org/10.1006/inco.1994.1037>
10. Kozen, D.: Kleene algebra with tests. *ACM Trans. Program. Lang. Syst.* **19**(3), 427–443 (1997). <https://doi.org/10.1145/256167.256195>
11. Kozen, D.: On Hoare logic and Kleene algebra with tests. *ACM Trans. Comput. Logic* **1**(1), 60–76 (2000). <https://doi.org/10.1145/343369.343378>

12. Kozen, D.: Automata on guarded strings and applications. Cornell University, Technical report (2001)
13. Kozen, D., Smith, F.: Kleene algebra with tests: completeness and decidability. In: van Dalen, D., Bezem, M. (eds.) Computer Science Logic. CSL 1996. LNCS, vol. 1258, pp. 244–259. Springer, Berlin, Heidelberg (1997). https://doi.org/10.1007/3-540-63172-0_43
14. Kozen, D., Tiuryn, J.: Substructural logic and partial correctness. ACM Trans. Comput. Logic **4**(3), 355–378 (2003). <https://doi.org/10.1145/772062.772066>
15. Kuznetsov, S.: Action logic is undecidable. ACM Trans. Comput. Logic **22**(2), 1–26 (2021). <https://doi.org/10.1145/3445810>
16. McCune, W.: Prover9 and Mace4 (2005–2010). <https://www.cs.unm.edu/mccune/prover9/>
17. McLean, B.: Free Kleene algebras with domain. J. Logical Algebraic Meth. Program. **117**, 100606 (2020). <https://doi.org/10.1016/j.jlamp.2020.100606>
18. Pratt, V.: Action logic and pure induction. In: van Eijck, J. (ed.) JELIA 1990. LNCS, vol. 478, pp. 97–120. Springer, Heidelberg (1991). <https://doi.org/10.1007/BFb0018436>
19. Restall, G.: An Introduction to Substructural Logics. Routledge, London (2000)



The Limits to Gossip: Second-Order Shared Knowledge of All Secrets is Unsatisfiable

Hans van Ditmarsch¹  and Malvin Gattinger²  

¹ Open University of the Netherlands, Heerlen, The Netherlands
hans.vanditmarsch@ou.nl

² University of Amsterdam, Amsterdam, The Netherlands
malvin@w4eg.eu

Abstract. It is known that without synchronization via a global clock one cannot obtain common knowledge by communication. Moreover, it is folklore that without exchanging higher-level information arbitrary higher-level shared knowledge cannot be achieved.

Here we make this result precise. We use epistemic logic to formally define “everyone knows that everyone knows that everyone knows all secrets” and then prove that this statement is unsatisfiable.

Keywords: Epistemic logic · Gossip · Higher-order knowledge

1 Introduction

Consider a group of agents that initially each know a unique secret. The agents then make one-to-one phone calls during which they always share all the secrets they know. So-called *gossip protocols* in this simple model provide an efficient way to spread information using peer-to-peer communication.

Even when the gossiping agents only exchange secrets and no additional information, they also obtain higher-order knowledge: they learn what other agents know, or even what other agents know about yet other agents, and so on. But it is known that without synchronization via a global clock one cannot obtain common knowledge by peer-to-peer communication [5]. It is folklore that only the first level of shared knowledge can be achieved, but not the higher levels. In this article we make this result precise using epistemic logic.

We start with two examples that illustrate interesting cases of when and how agents can obtain higher-order knowledge.

Example 1. Suppose we have a set of four agents $\{a, b, c, d\}$. Consider the sequence of calls $ab.cd.ac.bd.ad.bc.ab.cd$ and the results shown in Table 1. After the fourth call bd everyone is an expert, i.e. they know all secrets. In both of the last two calls ab and cd the two agents involved become so-called super experts, i.e. they know that everyone knows all secrets. The example shows in particular that two agents can become super experts at the same time.

Table 1. Results of $ab.cd.ac.bd.ad.bc.ab.cd$. A lower case y in column x means x knows the secret of y ; an upper case Y means x knows that y is an expert. Therefore, “abcd” denotes an expert and “ABCD” denotes a super expert.

	a	b	c	d	initial state
$\xrightarrow{ab}$	ab	ab	c	d	
$\xrightarrow{cd}$	ab	ab	cd	cd	
$\xrightarrow{ac}$	abcd A C	ab	abcd A C	cd	
$\xrightarrow{bd}$	abcd A C	abcd B D	abcd A C	abcd B D	everyone is an expert
$\xrightarrow{ad}$	abcd A CD	abcd B D	abcd A C	abcd AB D	
$\xrightarrow{bc}$	abcd A CD	abcd BCD	abcd ABC	abcd AB D	
$\xrightarrow{ab}$	abcd ABCD	abcd ABCD	abcd ABC	abcd AB D	a, b become super experts
$\xrightarrow{cd}$	abcd ABCD	abcd ABCD	abcd ABCD	abcd ABCD	c, d become super experts

Table 2. Results of $ac.ad.ac.bc.ac$ including a lucky call.

	a	b	c	d	initial state
$\xrightarrow{ac}$	a c	b	a c	d	
$\xrightarrow{ad}$	a cd	b	a c	a cd	
$\xrightarrow{ac}$	a cd	b	a cd	a cd	a learns d from c
$\xrightarrow{bc}$	a cd	abcd BC	abcd BC	a cd	
$\xrightarrow{ac}$	abcd ABC	abcd BC	abcd ABC	a cd	a is <i>lucky</i> about b

More interestingly, an agent may learn that another agent is an expert without calling that agent. We will call this a lucky call and this notion plays a key role in the proof of our main result. In a synchronous setting where agents observe that other calls happen, lucky calls happen frequently. But in the asynchronous setting we consider here agents only observe their own calls. This limits what agents learn about each other and makes lucky calls noteworthy.

Example 2. Again suppose we have a set of four agents $\{a, b, c, d\}$ and consider the call sequence $ac.ad.ac.bc.ac$, with results shown in Table 2. Here agent a learns in the final call ac that a, b, c are experts. Because b is not involved in this call we say that this a *lucky call* and say that a is *lucky about* b . We will show that this sequence is typical: in any call sequence each agent can only once be lucky and at most $n - 2$ agents out of n agents (for $n \geq 4$) can be lucky.

Our main contribution here is the proof that “everyone knows that everyone knows that everyone knows all secrets” is unsatisfiable. On the way to this result we also provide a new definition of the “causal cone” of an agent or a subset of agents in a call sequence. This article is structured as follows. We discuss related work in Sect. 2 and provide definitions in Sect. 3. In Sects. 4 we define causal cones and in Sect. 5 we use them to characterise lucky calls. Section 6 contains our main result.

2 Related Work

The “gossip problem” as introduced above is also known as the “telephone problem” and goes back (at least) to the article [9] from 1971. The main classical result is that only a linear amount of calls ($2n - 4$ if we have n agents) is needed to ensure that *everyone knows all secrets*. We refer to [6] for a survey of variants of the gossip problem, for example over different graphs or using broadcasting.

Most of the classical results assume a central scheduler, i.e. an authority that decides in which order calls should be made. More recently, decentralised gossip has been studied, where agents decide on their own whom they should call next, and multiple logics have been developed to analyse the gossip problem and different protocols [1–3]. Some of these logics include not only statements to say that agents know a secret, but they also provide general “an agent knows that φ ” modalities common in epistemic logics, and thereby allow us to discuss the higher-order knowledge effects of gossip.

The question which higher-order knowledge can be achieved by communication between agents goes back to the classic problem of the Byzantine Generals [8]. In general it is impossible to achieve *common knowledge* in an asynchronous distributed system, as shown in [5]. However, *shared knowledge* can be achieved with messages of the form “I know that this other agents knows that ...”. This has been studied in [7] where agents always tell each other *all* they know. Concretely, among n agents shared knowledge of level k can be achieved with $(k + 1)(n - 2)$ many calls. Here we only allow agents to exchange secrets and do not allow them to exchange any other kind of information.

Our result is most related to [1, 3] who both investigate when the truth of formulas stabilises during gossip protocol execution, including the case of the most general gossip protocol where any call can be made at any time and where agents only observe their own calls (setting $\langle \bullet, \diamond, \beta \rangle$ in [3], also called asynchronous ANY in [4]). This is the same setting as ours.

By different methods the authors of [1] and those of [3] demonstrate that making new calls no longer affects the truth of epistemic formulas at some stage. Although the objectives of these publications were different, namely decidability of logics for gossip or correctness of gossip protocols, there is some overlap in methods. They show that in any (fairly scheduled) call sequence, with the standard call semantics that only secrets are exchanged in a call, at some stage further calls have no informational effect—such calls are redundant. This was relevant to observe for gossip protocols where the goal was that all agents became experts, because it showed that, in principle, even if one were to consider epistemic goals such as knowing that others are experts, things would eventually come to a stop. But they did not consider any specific epistemic goals.

In [1] only epistemic formulas of depth 1 were considered (the crucial result is [1, Lemma 21]), but here we focus on depth 2 and higher which in [1] is only mentioned as a generalization for future work.

The authors of [3] considered arbitrary epistemic formulas. However, there the comparison to our result stops: [3, Prop. 5.5, Cor. 5.6] shows that formulas of any epistemic depth remain true forever or false forever after call sequences

of certain length (bounded by a polynomial in terms of the number of agents). Hence a formula like $EEExp_A$ must remain true forever or false forever after further extending call sequences. But the authors of [3] did not investigate specific formulas. Here we show that $EEExp_A$ remains false forever.

3 Syntax and Semantics

We assume a finite set of at least four agents $A = \{a, b, c, d, \dots\}$ throughout this article. This assumption is needed for our main results. Some lemmas may also hold for less than four agents, but these are boundary cases of little interest.

Each agent holds a single secret. The agents communicate with each other through telephone calls. During a call between two agents x and y , they exchange all the secrets that they knew before the call.

A *call* is a pair of agents $(x, y) \in A \times A$ for which we write xy . Agent x is the *caller* and agent y is the *callee*. Given call xy , call yx is the *dual call*. An agent x is *involved* in a call yz iff $y = x$ or $z = x$. In this contribution the direction of the call does not matter, so it only matters if an agent is involved in a call. We will therefore arbitrarily write xy or yx for the call between x and y , where we often prefer the lexicographic order of agents. A *call sequence* is defined by induction: the empty sequence ϵ is a call sequence. If σ is a call sequence and xy is a call, then $\sigma.xy$ is a call sequence. We write $|\sigma|$ to denote the length of a call sequence.

Given call sequences τ, σ , by induction on the length of σ we further define that τ is a *subsequence* of σ . This is the inductive definition: $\epsilon \subseteq \epsilon$, and if $\tau \subseteq \sigma$ then $\tau, \tau.ab \subseteq \sigma.ab$ and $\tau, ab.\tau \subseteq ab.\sigma$.

If $\sigma = \rho.\tau$, then ρ is a *prefix* of σ , denoted as $\rho \sqsubseteq \sigma$, and τ is the *complement* of ρ in σ , where τ is also denoted $\sigma \setminus \rho$.

Definition 1 (Language). For a finite set of agents A the language $\mathcal{L}$ is given by $\varphi ::= b_a \mid \neg\varphi \mid \varphi \wedge \varphi \mid K_a\varphi$ where $a, b \in A$. Let $\rightarrow$ and $\vee$ be defined as usual.

The φ are called *formulas*. The atomic formula b_a reads as ‘agent a has the secret of b ’ or ‘agent a knows the secret of b ’. The formula $K_a\varphi$ reads ‘agent a knows that φ is true’. We also define $E\varphi := \bigwedge_{a \in A} K_a\varphi$ and read it as ‘everyone knows that φ ’ ($E\varphi$ is also known as *shared* or *mutual* knowledge of φ).

Agent a is an *expert* if she knows all the secrets, formally $\bigwedge_{b \in A} b_a$, abbreviated as Exp_a . *Everyone is an expert* is represented by the formula $Exp_A := \bigwedge_{a \in A} \bigwedge_{b \in A} b_a$. Agent a is a *super expert* if she knows that everyone is an expert, formally $K_a Exp_A$. Therefore, $EEExp_A$ means that everyone is a super expert and $EEExp_A$ means that everyone knows that everyone is a super expert. The main result we prove in this article is that the latter cannot be achieved, i.e. that $EEExp_A$ is not satisfiable.

Let $B \subseteq A$. If $a \in B$ knows the secrets of all the agents in B , a is a *B expert*, and if a knows that all the agents in B know the secrets of all the agents in B , a is a *B super expert*. We find it convenient to have some additional notation for such matters: $\bigwedge_{b \in B} K_b\varphi$ is denoted $E_B\varphi$, and ‘ $a \in B$ knows the secrets of all the

agents in B' is denoted $Exp_a(B)$ with similar notational variations as before, so that $E_B Exp_B(B)$ denotes that all in B are B super experts (and for $B = A$ we omit the parameter).

The epistemic relation defined below models that agents only observe the calls they are involved in. In particular there is no global clock and the conditions are asynchronous, meaning agents do not know how many calls have taken place.

Definition 2 (Epistemic relation). *Let $a \in A$. The epistemic relation $\sim_a$ is the smallest equivalence relation between call sequences such that:*

- $\epsilon \sim_a \epsilon$
- if $\sigma \sim_a \tau$ and $a \notin \{b, c\}$, then $\sigma.bc \sim_a \tau$
- if $\sigma \sim_a \tau$, and for all c , $\sigma \models c_b$ iff $\tau \models c_b$, then $\sigma.ab \sim_a \tau.ab$

Definition 3 (Semantics). *Let call sequence σ and formula $\varphi \in \mathcal{L}$ be given. We define $\sigma \models \varphi$ by induction on the structure of φ . Moreover, we define the valuation of atoms by induction on σ , for any $a, b \in A$ with $a \neq b$.*

$$\begin{array}{ll}
\epsilon \models a_b & \text{iff } a = b \\
\sigma.ab \models c_a & \text{iff } \sigma \models c_a \text{ or } \sigma \models c_b & \text{for all } c \in A \\
\sigma.ab \models c_b & \text{iff } \sigma \models c_a \text{ or } \sigma \models c_b & \text{for all } c \in A \\
\sigma.ab \models c_d & \text{iff } \sigma \models c_d & \text{for all } c, d \in A \text{ with } d \notin \{a, b\} \\
\sigma \models \neg\varphi & \text{iff } \sigma \not\models \varphi \\
\sigma \models \varphi \wedge \psi & \text{iff } \sigma \models \varphi \text{ and } \sigma \models \psi \\
\sigma \models K_a \varphi & \text{iff } \tau \models \varphi \text{ for all } \tau \text{ such that } \sigma \sim_a \tau
\end{array}$$

A formula φ is valid, notation $\models \varphi$, iff for all call sequences σ we have $\sigma \models \varphi$. We abbreviate the set of secrets know by a after σ with $a(\sigma) := \{c \in A \mid \sigma \models c_a\}$.

If in call ab agent a or b becomes an expert, then the other agent must also be an expert after this call. In contrast, if in a call ab agent a or b becomes a super expert, then the other agent does not have to be a super expert after this call.

4 Causal Relation and Causal Cone

We now introduce additional notation for specific subsequences of calls. The goal is to make it easy to select and reason about those calls that are relevant for a specific subset of agents.

Definition 4 (Causal relation). *For any sequence σ and calls $ab, cd \in \sigma$, we write $ab < cd$ iff σ has shape $\sigma_1.ab.\sigma_2.cd.\sigma_3$ (i.e. this occurrence of the call ab in σ is before this occurrence of the call cd in σ). We define the relation $\ll_0$ over call occurrences in σ by $ab \ll_0 cd :\Leftrightarrow ab < cd$ and $\{a, b\} \cap \{c, d\} \neq \emptyset$. Let $\ll$ be the reflexive transitive closure of $\ll_0$. Calls ab and cd are causally related iff $ab \ll cd$.*

Definition 5 (Causal cone). Given a call sequence σ and a set of agents $B \subseteq A$, the causal cone $\sigma_{\ll}^B$ is defined by induction:

$$\epsilon_{\ll}^B := \epsilon \quad (\sigma.ab)_{\ll}^B := \begin{cases} \sigma_{\ll}^{B \cup \{a,b\}}.ab & \text{if } a \in B \text{ or } b \in B \\ \sigma_{\ll}^B & \text{otherwise} \end{cases}$$

For $\sigma_{\ll}^{\{a,b\}}$ we write $\sigma_{\ll}^{ab}$. Furthermore, let $\sigma_{\not\ll}^B$ be the complement of $\sigma_{\ll}^B$ in σ .

Intuitively, given a call sequence $\sigma.ab$, the sequence $\sigma_{\ll}^{ab}$ is the subsequence of σ of which all calls are *causally related* to the final call ab (it is the causal cone of ab), in other words $\sigma_{\ll}^{ab}$ is the subsequence consisting of all $cd \in \sigma$ such that $cd \ll ab$ in $\sigma.ab$. Also note that we can identify $\sigma_{\ll}^a$ with the causal cone of the last call in σ involving a . In other words, $\sigma_{\ll}^a$ determines what a knows after σ .

The complement $\sigma_{\not\ll}^B$ are the calls that do *not* determine what a knows after σ . For example, $\sigma_{\not\ll}^{ab}$ consists of all $cd \in \sigma$ that do not determine what a or b know after σ .

The causal relations between calls are only interesting when there are at least four agents. If there are two or three agents, all calls are causally related.

Lemma 1. Let call sequence σ , group $B \subseteq A$ of agents, and agent $a \in B$ be given. Then (i) $\sigma \sim_a \sigma_{\ll}^B$ and also (ii) $\sigma \sim_a \sigma_{\ll}^B \cdot \sigma_{\not\ll}^B$.

Proof. We prove (i) by induction on the length of σ . Note that we declared B and $a \in B$ after σ , hence these may occur differently in our inductive assumption. For $B \cup \{a\}$ we write $B + a$ and similarly for $B \cup \{b, c\}$ we write $B + bc$.

Case $\sigma = \epsilon$. This is by definition as $\epsilon_{\ll}^B = \epsilon$.

Case $\sigma = \tau.ab$. By the definition of $\ll$ we have $(\tau.ab)_{\ll}^B = \tau_{\ll}^{B+b}.ab$ (we recall that $a \in B$). By inductive assumption we have $\tau \sim_a \tau_{\ll}^{B+b}$ and also $\tau \sim_b \tau_{\ll}^{B+b}$, so that agent b holds the same secrets after both. Therefore, by definition of $\sim_a$, we have $\tau.ab \sim_a \tau_{\ll}^{B+b}.ab$. Combining this we obtain $\tau.ab \sim_a (\tau.ab)_{\ll}^B$.

Case $\sigma = \tau.bc$ with $b, c \neq a$ and $b \in B$ or $c \in B$. By the definition of $\ll$ and because b or c is in B , $(\tau.bc)_{\ll}^B = \tau_{\ll}^{B+bc}.bc$. By induction, $\tau \sim_a \tau_{\ll}^{B+bc}$, and from that, the fact that $b, c \neq a$, and the definition of $\sim_a$ we also obtain $\tau.bc \sim_a \tau_{\ll}^{B+bc}.bc$, and therefore $\tau.bc \sim_a (\tau.bc)_{\ll}^B$.

Case $\sigma = \tau.bc$ with $b, c \notin B$. We have that $\tau.bc \sim_a \tau$ by the definition of $\sim_a$, because $b, c \neq a$. By inductive assumption, $\tau \sim_a \tau_{\ll}^B$. By the definition of $\ll$ and because $b, c \neq a$, $\tau_{\ll}^B = (\tau.bc)_{\ll}^B$. Combining all this we obtain $\tau.bc \sim_a (\tau.bc)_{\ll}^B$.

We now prove (ii). From $\sigma \sim_a \sigma_{\ll}^B$ it follows that $\sigma \sim_a \sigma_{\ll}^B \cdot \sigma_{\not\ll}^B$ by the definition of $\sim_a$ and the observation that a does not occur in any call in $\sigma_{\not\ll}^B$. $\square$

An instantiation of Lemma 1 is that $\sigma \sim_a \sigma_{\ll}^a \cdot \sigma_{\not\ll}^a$ and $\sigma \sim_a \sigma_{\ll}^a$: a considers it possible that all not causally related calls, if any, take place after her last call.

Moreover, note that we have $(\sigma.ab)_{\ll}^{ab} = \sigma_{\ll}^{ab}.ab$ and $(\sigma.ab)_{\not\ll}^{ab} = \sigma_{\not\ll}^{ab}$ by Definition 5. Hence Lemma 1 also implies the following corollary.

Corollary 1. Let call sequence σ and call ab be given. Then $\sigma.ab \sim_a \sigma_{\ll}^{ab}.ab \cdot \sigma_{\not\ll}^{ab}$ and also $\sigma.ab \sim_a \sigma_{\ll}^{ab}.ab$.

5 Lucky Calls

In the introduction we informally introduced the notion of a lucky call. The following definition makes this notion precise and the goal of this section is to characterise when lucky calls can happen.

Definition 6. *An agent a is lucky in a call ab if she learns in that call that another agent c is an expert. Formally, given a call sequence $\sigma.ab$ and $c \notin \{a, b\}$, agent a is lucky about c in ab iff we have $\sigma \not\models K_a \text{Exp}_c$ and $\sigma.ab \models K_a \text{Exp}_c$.*

In subsequent proofs we show and use that only $A - b$ super experts, who know that all but one agent b know the secrets of all but one agent b , can be lucky. We stress that our results only concern asynchrony. In a synchronous setting, agents are lucky all the time and this is nothing special.

Lemma 2. *When two agents become experts in a call, they cannot be lucky.*

Proof. Let σ and a, b be given such that $\sigma \not\models \text{Exp}_a$, $\sigma \not\models \text{Exp}_b$, and $\sigma.ab \models \text{Exp}_a$ as well as $\sigma.ab \models \text{Exp}_b$. We show that for all $c \neq a, b$, $\sigma.ab \not\models K_a \text{Exp}_c$.

From Corollary 1 it follows that $\sigma.ab \sim_a \sigma_{\ll}^{ab}.ab$. No call in $\sigma_{\ll}^{ab}$ may contain an agent who is an expert, as the causal relation would then have made a or b an expert before call ab . Therefore $\sigma_{\ll}^{ab} \not\models \text{Exp}_c$ for all $c \in A$ other than a or b , and because of that and the semantics of calls also $\sigma_{\ll}^{ab}.ab \not\models \text{Exp}_c$. From that and $\sigma.ab \sim_a \sigma_{\ll}^{ab}.ab$ it follows that $\sigma.ab \models \neg K_a \text{Exp}_c$.

Similarly we show (replace $\sim_a$ by $\sim_b$) that $\sigma.ab \not\models K_b \text{Exp}_c$. $\square$

Lemma 3. *When two agents become experts, neither becomes a super expert.*

Proof. We recall that there at least four agents. By Lemma 2, after the call wherein two agents become experts, they both remain uncertain whether the two or more agents not involved in the call are expert. $\square$

We already showed (Lemma 2) that when two agents become experts in a call, they cannot be lucky. Now we consider the case where one of the agents was already an expert before the call. We want to characterise when the other agent who becomes expert can be lucky.

As an example, recall the call sequence $ac.ad.ac.bc.ac$ from Example 2 wherein a learns that b and c are expert in the final call ac . In other words, a 's final call ac is lucky. After the prefix $ac.ad.ac$, a is a super expert for all agents but one (for $\{a, c, d\}$): a knows that a, c, d know all the secrets of a, c, d . This allows a to learn in the final call ac that *someone* from those all but one agents a, c, d must have called the one agent b and clearly it was not herself. Agent a learns in the final call that c or d called b , not that c called b .

We will now show that this typical case is also the only case.

Although we assume that there are at least four agents, the next result also holds for three agents. In call sequence $ac.bc.ac$, the third call is lucky and a then learns that c and b are experts. In the first call, a becomes a $\{a, c\}$ super expert. In the second call ac , agent a that c must have called b , and thus learns that b and c are experts, and thus becomes a $\{a, b, c\}$ super expert.

Lemma 4. *An agent becoming an expert can only be lucky if she is a super expert for all agents but one. Formally: Let $a, b, c \in A$ and σ be given and suppose $\sigma \not\models \text{Exp}_a$ and $\sigma.ac \models \text{Exp}_a$. Then $\sigma.ac \models K_a \text{Exp}_b$ iff $\sigma \models K_a \text{Exp}_{A-b}(A-b)$.*

Proof. Suppose we have $\sigma \not\models \text{Exp}_a$ and $\sigma.ac \models \text{Exp}_a$. We show the two directions. ($\Rightarrow$): We show the requested by contraposition: if a is not a $A-b$ super expert ($\sigma \not\models K_a \text{Exp}_{A-b}(A-b)$), then a cannot be lucky about b (i.e. $\sigma.ac \not\models K_a \text{Exp}_b$).

Intuitively, there are two ways in which non-expert a can be not a $A-b$ super expert: when she does not know enough or when she knows too much. In the first case a does not know the secret of b but she is not a super expert. In the second case a knows the secret of b , as this is, in a way, ‘more’ than being an $A-b$ super expert who is not an expert, which implies ignorance of b .

Case a does not know b . If $\sigma \models \neg b_a$ then a considers it possible that b has not yet made a call and thus only knows its own secret. In that case, as a is not a $A-b$ super expert, then a considers it possible that there is an agent $d \in A-b$ such that d does not know all of $A-b$ ’s secrets, that is, then there is a $e \in A-b$ (where e may be a or c) such that d does not know the secret of e . Agent a thus considers it possible that the next two calls are $db.dc$ and that b is not involved in further calls. After db , agent b is not an expert because neither b nor d know the secret of e . In dc agent d informs c of the secret of b . This can still be followed by any call sequence τ of calls between the agents of $A-ba$ making c expert before call ac . Altogether we get $\sigma.ac \sim_a \sigma.db.dc.\tau.ac$ and $\sigma.db.dc.\tau.ac \not\models \text{Exp}_b$. Therefore $\sigma.ac \not\models K_a \text{Exp}_b$.

Case a knows b . If $\sigma \models b_a$, then a cannot also be a $A-b$ super expert as this implies that a also knows all other secrets and therefore is an expert, which contradicts our assumption. It remains to show that $\sigma.ac \not\models K_a \text{Exp}_b$.

As non-expert a became expert in call ac , a learns a secret of some agent d in that call. As a already knew the secret of b , we must have $d \neq b$.

First assume there is a last call in σ between a and b . In that call ab , a therefore did not learn the secret of d . So after this call a still considers it possible that d only knows its own secret.

- If after call ab agent a also knows the secret of c , then a considers it possible that c does not know d . If then the subsequent calls are $bc.cd$ and b was not involved in further calls, then after call bc agent b still does not know d so b is not an expert. The part $bc.cd$ can still be followed by any sequence τ of calls between the agents of $A-ba$ making c expert before call ac . Altogether we get $\sigma.ac \sim_a \sigma.bc.cd.\tau.ac$ and $\sigma.bc.cd.\tau.ac \not\models \text{Exp}_b$. Therefore $\sigma.ac \not\models K_a \text{Exp}_b$.
- If after call ab agent a does not know the secret of c , then b also does not know c , and a considers it possible that subsequently $bd.cd$ took place. After bd agent b is not an expert (because b still does not know c). Call cd informs c of the secret of b . This can still be followed by any number of calls between the agents of $A-ba$ making c expert before call ac . Therefore $\sigma.ac \not\models K_a \text{Exp}_b$.

Second, assume there was no call in σ between a and b . Then, given that a knows b and thus knows that a call took place between b and some agent e (where e may be c) in $A-ba$, we again conclude that b did not know d after

that call nor after any call before the last call by a in σ (which happens to be the final call in $\sigma_{\ll}^a$). Agent a considers a call sequence possible (with prefix $\sigma_{\ll}^a$) wherein after her last call b did not make further calls and that b 's secret was instead initially spread by agent e among the $A - ba$, and so on until c became expert. Therefore also in this case, $\sigma.ac \not\models K_a \text{Exp}_b$.

($\Leftarrow$): Suppose $\sigma \models K_a \text{Exp}_{A-b}(A - b)$. Note that we not only have $\sigma \not\models \text{Exp}_a$, but also $\sigma \models K_a \neg \text{Exp}_a$. Moreover, we claim that $\sigma \models \text{Exp}_c$. To see this, note that after σ agent a is an $A - b$ super expert, so a knows that c knows all secrets except b . Therefore, if a becomes expert in the last a call with c , then c must have learnt another secret. This can only be the secret of b . Therefore a learnt that c was an expert after any $\tau \sim_a \sigma$.

In order to show $\sigma.ac \models K_a \text{Exp}_b$, let τ be arbitrary such that $\tau.ac \sim_a \sigma.ac$. By definition of $\sim_a$ we have $\tau \sim_a \sigma$. From $\sigma \models \text{Exp}_c$ we get $c(\tau) = c(\sigma) = A$. It remains to show that $\tau \models \text{Exp}_b$. Also note that $c(\tau) = A$ implies $\tau \models b_c$.

From $\tau \sim_a \sigma$ as well as $\sigma \models K_a \text{Exp}_{A-b}(A - b)$ and $\sigma \models K_a \neg \text{Exp}_a$ we obtain $\tau \models \text{Exp}_{A-b}(A - b)$ and $\tau \models \neg \text{Exp}_a$.

These two imply that $\tau_{\ll}^a \models \neg b_d$ for any $d \neq b$ (b does not occur in $\tau_{\ll}^a$) and therefore in particular that $\tau_{\ll}^a \models \neg b_c$.

From $\tau_{\ll}^a \models \neg b_c$ whereas $\tau \models b_c$ it follows that $\tau_{\ll}^a$ must contain a call bd involving b and some agent d where either $d = c$ or there is a subsequent call ce involving c (where $d, e \neq a$). Because $d(\tau_{\ll}^a) = A - b$ for any such d , in the call bd agent b becomes an expert. (Also, ce explains how agent c became an expert.)

Therefore $\tau \models \text{Exp}_b$, and as τ was arbitrary such that $\tau.ac \sim_a \sigma.ac$ this shows that $\sigma.ac \models K_a \text{Exp}_b$. $\square$

We can conclude from Lemma 4 that if $\sigma \models E_{A-b} \text{Exp}_{A-b}(A - b)$, so when all agents in $A - b$ are $A - b$ super experts, and if also no agent in $A - b$ is expert, all but one of those can be lucky in the same call sequence. To see this, after σ let some agent c call b . Let now all agents in $A - bc$ call c . Then they all also learn in that call that b is expert. Given $|A| = n$ agents, we therefore get $n - 2$ many lucky calls.

Example 3. With 4 agents we can have 2 lucky calls. Recall $ac.ad.ac.bc.ac$ from Example 2 wherein a learns that b and c are expert in the final call ac . Now consider the expanded sequence $ac.ad.ac.cd.bc.ac.cd$: after $ac.ad.ac.cd$, all of a, c, d know that all of a, c, d know all secrets of a, c, d . In penultimate call ac , a learns that b and c are expert, and in final call cd , d learns that b and c are expert.

6 Main Result

Lemma 5. *An agent cannot become an expert and a super expert in the same call.*

Proof. Suppose a becomes expert in ac , i.e. $\sigma \not\models \text{Exp}_a$ and $\sigma.ac \models \text{Exp}_a$. We need to show that $\sigma.ac \not\models K_a \text{Exp}_A$. From $\sigma \not\models \text{Exp}_a$ we know there must be a $d \in A$ such that $\sigma \models \neg d_a$. As there are at least four agents, there must be a $b \notin \{a, c, d\}$.

From $\sigma \models \neg d_a$ we get $\sigma \not\models \text{Exp}_a(A - b)$ and thus $\sigma \not\models K_a \text{Exp}_{(A-b)}(A - b)$. Now by Lemma 4 we have $\sigma.ac \not\models K_a \text{Exp}_b$. This implies $\sigma.ac \not\models K_a \text{Exp}_A$. $\square$

Lemma 6. *An agent becoming a super expert considers it possible that the other agent involved in that call did not become a super expert.*

Proof. Let $\sigma.ab$ be a call sequence wherein agent a becomes super expert in final call ab . From Lemma 5 we conclude that a was already expert after σ .

Suppose that b became expert in the call ab . Then also from Lemma 5 we conclude that b did not become a super expert in the call ab . As a considers the actual call sequence possible, we infer that agent a then considers it possible that agent b is not a super expert after $\sigma.ab$.

Thus we can assume that b already was an expert after σ . As a was expert before call ab , a became super expert by learning in call ab that b is expert.

First suppose a considers possible that b made a lucky call in σ . The lucky call was not with a , as a would then already have known that b is an expert. So let that call be bc with some agent $c \neq a$. If in that call b also learns that d is expert, where $d \neq a, c$, then a considers possible that the call instead of bc was bd . In call bd agent b only learns that d is an expert (we recall that by Lemma 4 agent b must have been an $A - d$ superexpert before that call), and does not learn that c is expert. So a considers a call sequence possible wherein b was not lucky when he became expert, and therefore when becoming expert remained uncertain whether some other agent c is expert.

We continue the argument by reasoning about this agent c .

Suppose a learnt that c is expert in lucky call ad . Then a considers possible that all further calls involving c were instead involving d except for further calls ac . After this call sequence b does not know that c is expert, so b is not a super expert, so a considers possible after $\sigma.ab$ that b is not a super expert.

Now suppose that ac is the first call in σ after which a knows that c is expert. We now distinguish four cases by whether c and b are experts before the call ac .

1 Suppose c was already expert before that call ac .

1.1 If b became expert before call ac in call bd with $d \neq c$, b was ignorant whether c is expert after that call (we assumed bd was not lucky), replace bd by $bd.cd$ in the call sequence between bd and ac , and replace all further occurrences of c except in further calls ac by d . This call sequence is indistinguishable for a and preserves that b does not know that c is expert (after final call ab). Therefore a considers possible after $\sigma.ab$ that b is not a super expert.

1.2 If b became expert after call ac in call bd with $d \neq c$, replace all subsequent occurrences of c except in calls ac by d . (We need not change any calls between ac and bd , as we assumed that b does not learn whether any other agent than d is expert in call bd , which includes agent c .) Then b does not know after final call ab that c is expert. Therefore a considers possible after $\sigma.ab$ that b is not a super expert.

2 Suppose c became expert in that call ac .

2.1 If b became expert before ac in bd with $d \neq c$, agent a considers possible that all further calls after ac involving c were instead by d , except for further

occurrences of ac . We need not change any calls involving c between bd and ac : note that this cannot have been bc as that would have made c expert before ac contrary to our assumption. This call sequence preserves that b does not know whether c is expert after final call ab . Therefore a considers possible after $\sigma.ab$ that b is not a super expert.

2.2 If b became expert after ac in bd with $d \neq c$ agent a considers possible all further calls after bd involving c were instead by d except for future occurrences of ac . Now consider the calls involving c between ac and bd : we need not change any of those as we assumed that b only learnt that d is expert in call bd . This call sequence preserves that b does not know whether c is expert after final call ab . Therefore a considers possible after $\sigma.ab$ that b is not a super expert.

We have exhaustively investigated all cases and this ends the proof. $\square$

In Lemma 6 it is important to observe that the agent c that agent a remains uncertain about is different from the agent b involved in the call wherein she became super expert. This will be used in the final theorem.

Example 4. As an example of Lemma 6, recall the sequence $ab.cd.ac.bd.ad.bc.ab$ from Example 1 above where a and b become super experts in the last call. It is indistinguishable for a from call sequence $ab.cd.ac.bd.ad.ab$ after which b is not a super expert. Alternatively we could replace bc by bd in the actual sequence.

With Lemma 6 we can now prove our main result.

Theorem 1. $EEExp_A$ is unsatisfiable.

Proof. Let ρ be an arbitrary call sequence. We show that $\rho \not\models EEEExp_A$. If $\rho \not\models EEEExp_A$, then clearly $\rho \not\models EEEExp_A$. Hence we assume that $\rho \models EEEExp_A$.

Consider any agent a becoming a super expert in ρ , in other words choose σ and τ such that $\rho = \sigma.ab.\tau$ where in call ab agent a becomes a super expert. We will show that $\rho \not\models K_a K_b Exp_c$ for some $c \in A$.

From Lemma 6 it follows that after $\sigma.ab$ agent a considers possible a call sequence $\sigma'.ab$ after which b is not a super expert. Therefore, b considers possible a call sequence $\sigma''.ab$ that does not satisfy Exp_A , that is, $\sigma''.ab$ does not satisfy Exp_c for some $c \in A$. Clearly we must have that $c \neq a$ (because b learnt that a is an expert in the call ab). Formally, we have: $\sigma.ab \sim_a \sigma'.ab$ and $\sigma'.ab \sim_b \sigma''.ab$ and $\sigma''.ab \models \neg Exp_c$. The last implies $\sigma'' \models \neg Exp_c$.

The sequence $\sigma.ab.\tau$ is indistinguishable for a from $\sigma.ab.\tau'$ where τ' is τ without all calls involving b but not a . This is because no secrets are exchanged in any call in τ , because all agents are already experts (because a is a super expert after $\sigma.ab$).

For the same reason, $\sigma.ab.\tau'$ is indistinguishable for b from $\sigma.ab.\tau''$ where τ'' is τ' restricted to calls involving b . Call sequence τ'' is a finite and possibly empty sequence consisting only of calls ab . We can write ab^n for that, where $n \in \mathbb{N}$ is the number of occurrences of ab in τ (and τ').

First, from $\sigma.ab \sim_a \sigma.ab$ (reflexivity of $\sim_a$), $\tau \sim_a \tau'$, and the fact that all are experts before τ and τ' (so no new secrets are learnt in calls) we get

$\sigma.ab.\tau \sim_a \sigma.ab.\tau'$. Then, from the assumption $\sigma.ab \sim_a \sigma'.ab$, $\tau' \sim_a \tau'$ (reflexivity of $\sim_a$ again), and that all are experts before τ' , we get $\sigma.ab.\tau' \sim_a \sigma'.ab.\tau'$. Finally, from $\sigma.ab.\tau \sim_a \sigma.ab.\tau'$ and $\sigma.ab.\tau' \sim_a \sigma'.ab.\tau'$ and transitivity of $\sim_a$ we then get $\sigma.ab.\tau \sim_a \sigma'.ab.\tau'$.

Similarly, from $\sigma'.ab \sim_b \sigma'.ab$ and $\tau' \sim_b \tau''$ we get $\sigma'.ab.\tau' \sim_b \sigma'.ab.\tau''$, from $\sigma'.ab \sim_b \sigma''.ab$ and $\tau'' \sim_b \tau''$ we get $\sigma'.ab.\tau'' \sim_b \sigma''.ab.\tau''$ (although c is not an expert in σ'' , a and b are, so as above no new secrets are learnt in calls in τ''), and therefore from both we obtain $\sigma'.ab.\tau' \sim_b \sigma''.ab.\tau''$.

From $\sigma'' \not\models Exp_c$ and $\sigma''.ab.\tau'' = \sigma''.ab^{n+1}$ we obtain $\sigma''.ab.\tau'' \not\models Exp_c$. Finally, in view of the above $\sigma'.ab.\tau' \not\models K_b Exp_c$, and also $\sigma.ab.\tau \not\models K_a K_b Exp_c$.

This implies $\rho \not\models EEExp_A$ and because ρ was an arbitrary call sequence, we have shown $EEExp_A$ must be unsatisfiable. $\square$

7 Conclusion and Open Questions

We have shown that “everyone knows that everyone knows all secrets” is the maximum level of shared knowledge that can be reached in asynchronous gossip. Formally, we can reach $EEExp_A$ but the next level $EEEExp_A$ is unsatisfiable.

Our results suggest at least two immediate related questions. First, it remains open whether $K_i EEExp_A$ is satisfiable. That is, can at least one agent learn that everyone is a super-expert? Second, a footnote to [3]: maybe one does not need to consider arbitrary epistemic formulas, but could show that beyond a certain modal depth *any formula* of a certain kind (with positively stacked modalities, i.e. excluding negations before epistemic modalities) is unsatisfiable. That would simplify model checking gossip.

Finally, in ongoing work we study *minimal* call sequences reaching super success. We conjecture that at least $2n - 3$ calls are needed to reach $K_i Exp_A$ for some agent i , and that at least $n - 2 + \binom{n}{2}$ calls are needed to reach $EEExp_A$.

References

1. Apt, K., Wojtczak, D.: Verification of distributed epistemic gossip protocols. J. Artif. Intell. Res. **62**, 101–132 (2018). <https://doi.org/10.1613/jair.1.11204>
2. van Ditmarsch, H., Gattinger, M., Kuijer, L., Pardo, P.: Strengthening gossip protocols using protocol-dependent knowledge. J. Appl. Logics - IfCoLog J. Logics Appl. **6**(1), 157–203 (2019). <https://arxiv.org/abs/1907.12321>
3. van Ditmarsch, H., van der Hoek, W., Kuijer, L.: The logic of gossiping. Artif. Intell. **286**, 103306 (2020). <https://doi.org/10.1016/j.artint.2020.103306>
4. van Ditmarsch, H., Gattinger, M., Ramezani, R.: Everyone knows that everyone knows: gossip protocols for super experts. CoRR abs/2011.13203 (2020). <https://arxiv.org/abs/2011.13203>
5. Halpern, J.Y., Moses, Y.: Knowledge and common knowledge in a distributed environment. J. ACM **37**(3), 549–587 (1990). <https://doi.org/10.1145/79147.79161>
6. Hedetniemi, S.M., Hedetniemi, S.T., Liestman, A.L.: A survey of gossiping and broadcasting in communication networks. Networks **18**(4), 319–349 (1988). <https://doi.org/10.1002/net.3230180406>

7. Herzig, A., Maffre, F.: How to share knowledge by gossiping. *AI Commun.* **30**(1), 1–17 (2017). <https://doi.org/10.3233/aic-170723>
8. Lamport, L., Shostak, R., Pease, M.: The Byzantine generals problem. *ACM Trans. Program. Lang. Syst.* **4**(3), 382–401 (1982). <https://doi.org/10.1145/357172.357176>
9. Tijdeman, R.: On a telephone problem. *Nieuw Archief voor Wiskunde* **3**(19), 188–192 (1971)



Additive Types in Quantitative Type Theory

Vít Šeřl^(✉) and Tomáš Svoboda

Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic
seřl@ksvi.mff.cuni.cz, svoboda@posteo.net

Abstract. Dependent type theories enable us to reason about properties of our programs, while substructural type theories enable us to reason about their resource use. Quantitative type theory seamlessly combines dependent and substructural types by employing positive semirings to keep track of variable usage and computational contexts. Existing treatments of this theory typically focus on multiplicative connectives such as functions or multiplicative pairs. In this work, we extend quantitative type theory with additive zero, unit, unions and dependent pairs, as well as annotated eliminators for various types. We then explain how these additive types can be used to better describe resource use and how they integrate with existing programming techniques. Finally, we implement an interpreter for a language based on these extensions.

1 Introduction

Type systems allow us to describe and exhibit certain program properties in a way that can be checked by a computer. Today, the most expressive type systems are typically based on various *dependent type theories*, such as calculus of constructions or Martin-Löf type theory. A typical example of a dependent type is the vector: a list of elements whose length is part of the type. Functions that operate on these vectors need to respect not only the type of the elements, but also the lengths of the vectors. This restriction further constrains possible implementations, reducing the chance of errors. Thanks to their expressiveness, dependent type theories are found in proof assistants or programming languages where verification is paramount.

However, whereas a proof assistant does not mind that an assumption is used twice, a programmer might want to avoid the duplication of an extensive data structure. *Substructural type systems* can be used to describe how a program manipulates its resources. These type systems restrict when variables can be duplicated or discarded. Combinations of these restrictions give rise to different substructural type systems. A well-known example is the linear type system. In this system, variables come in two flavors: unrestricted and linear. Linear variables cannot be discarded or duplicated and must thus be used exactly once.

This work was supported by the Charles University grant SVV-260588.

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
A. Ciabattini et al. (Eds.): WoLLIC 2022, LNCS 13468, pp. 250–262, 2022.
https://doi.org/10.1007/978-3-031-15298-6_16

Since dependent types and substructural types are not mutually exclusive, there have been attempts at combining them into a single type system. Historically, these attempts often restricted how dependent and substructural types are able to interact, such as only allowing dependency on unrestricted variables.

Quantitative type theory [1] (abbreviated QTT) is a recent type theory which seamlessly combines dependent and substructural types. QTT drops the distinction between linear and unrestricted variables. Instead, each variable is annotated by *multiplicity*, an element of some semiring, which describes how the variable is used. Semiring operations describe how the multiplicity changes when the variable is used in different contexts. As an example, multiplication by zero ensures that a variable occurrence in a type does not count towards its use.

Depending on the semiring, QTT can describe a wider variety of substructural types. Apart from the usual linear and intuitionistic function types, we can have functions that cannot use its parameter in a relevant context, functions that must use its parameter exactly n times, at most n times, and so on. Similar variations also exist for other types, such as pairs or modalities. Moreover, QTT can also describe dependent versions of these types.

Substructural type systems distinguish between two kinds of types: *multiplicative* and *additive*. When constructing a value of a multiplicative type, the resources are split into multiple groups, which are then used to create a part of the value. Additive types do not split the resources. When constructing a value, each part can access all resources. To ensure that these resources are not used more than once, only one part of the resulting value can be accessed.

1.1 Contributions

We introduce the concept of an annotated eliminator, which enables the use of the weakening rule on the bound variables rather than the eliminated expression. We extend the theory with multiple additive types: zero, unit, unions and pairs. In particular, we define dependent additive pairs, which do not exist in the normal substructural type theories. We show how these types can be used in practice and how they interact with existing programming techniques. Finally, we build a bidirectional type system and implement it in an interpreter.¹

1.2 Structure

The work is structured as follows. In the second section, we discuss other approaches to additive types. The third section provides an overview of key concepts: semirings, multiplicities, and weakening. Multiplicative types, together with the annotated eliminator, are introduced in the fourth section. Additive types are introduced in the fifth section. This section also discusses possible uses for these types, focusing primarily on dependent additive pairs. The sixth section details the process of building bidirectional type checking for the modified theory. The seventh section describes the interpreter and its implementation.

¹ <https://github.com/svobot/janus>.

2 Related Work

The idea of combining linear and dependent types is not new. One of the earliest such systems is Cervesato and Pfenning's Linear Logical Framework [5]. In this system, the typing context is split into a *linear* part and an *intuitionistic* part. However, this split also creates some restrictions. As an example, a dependent type cannot mention linear variables. A more recent example of a system that distinguishes between linear and intuitionistic parts is Krishnaswami et al. [8].

Semirings have also been used to keep track of usage. Brunel et al. [4] use semiring annotations with exponential modality. This idea is then generalized to encompass *coeffacts*. The typing context is not split and contains both linear and *discharged* variables, the latter of which contains a semiring annotation.

McBride [10] describes a dependent type theory in which semiring zero is used to represent computational irrelevance. By treating types as a computationally irrelevant context, this theory allows types to depend on both linear and intuitionistic variables. The work also describes weakening as well as a translation into an untyped lambda calculus which erases irrelevant portions of the program.

However, the system described by McBride does not admit substitution. This problem is fixed by Atkey [1], who also extends the theory with dependent multiplicative pairs and booleans, and provides a categorical model.

Note that QTT essentially disables resource checking for the irrelevant fragment, which might be undesirable in some cases. Choudhury et al. [6] describe a graded dependent type system which does not treat types in a special way and can thus be applied to a wider range of semirings. Interestingly, this system contains additive union eliminator which can consume the given value more than once. However, this feature is not applied to any other eliminator in the system.

QTT has also been successfully implemented into programming languages. A prime example is the Idris2 language, developed by Brady et al. [2,3]. Another example is the language Juvix [11], which features dependent additive pairs.

3 Quantitative Type Theory

3.1 Semirings

QTT uses semirings to keep track of variable usage. A semiring is a tuple $(S, +, \cdot, 0, 1)$ where S is a set, $+$ and $\cdot$ are functions of type $S \times S \rightarrow S$, 0 and 1 are elements of S such that $(S, +, 0)$ is a commutative monoid, $(S, \cdot, 1)$ is a monoid, $\cdot$ distributes over $+$, and $0a = 0 = a0$. Since variable usage cannot be negative, the semiring also needs to be *positive*. A positive semiring additionally requires: $a + b = 0$ implies $a = 0$ and $b = 0$; $ab = 0$ implies $a = 0$ or $b = 0$.

Different semirings give rise to different variations of QTT. As an example, the $(\mathbb{N}, +, \cdot, 0, 1)$ semiring tracks exact usage. In this work, we use the zero-one-many semiring $(\{0, 1, \omega\}, +, \cdot, 0, 1)$ where ω represents more than one use. In this semiring, $\omega \cdot \omega = \omega$ and $\rho + \omega = \omega$, the rest follows from the semiring properties. After we introduce weakening, we will be able to identify ω with any use, much like unrestricted variables in linear type systems.

3.2 Typing Rules

The standard typing judgment $\Gamma \vdash M : T$ does not provide enough information to correctly account for variable usage. Firstly, the typing context Γ needs to be extended. Instead of containing pairs (x, T) (written as $x : T$) where x is a name of a variable and T is its type, the context will now contain triples (x, T, π) (written as $x \overset{\pi}{:} T$) where π is an element of the semiring.

Secondly, the judgment itself needs to track the usage context. There are circumstances where variable occurrences do not count toward that variable's usage. In the extended judgment $\Gamma \vdash M \overset{?}{:} T$, the usage context is given by σ , an element of the semiring. In theory, any element can be used. However, this work follows the description given by Atkey [1] where σ can only be semiring's zero (irrelevant context) or one (relevant context).

Finally, the typing rules need to correctly handle extended typing contexts. The variable rule associates the multiplicity of the variable in the typing context with the multiplicity in the judgment itself. Other multiplicities must be zero.

$$\frac{0\Gamma_1, x \overset{?}{:} S, 0\Gamma_2 \vdash}{0\Gamma_1, x \overset{?}{:} S, 0\Gamma_2 \vdash x \overset{?}{:} S} \text{VAR}$$

In a linear type system, we need to distinguish between linear functions (written as $A \multimap B$) that must use its parameter exactly once, and unrestricted functions (written as $A \rightarrow B$). In QTT, these two types are a special case of a more general function type, $(x \overset{\pi}{:} S) \rightarrow T$. Functions of this type can use its parameter according to the multiplicity π . In our system, linear functions correspond to $(x \overset{!}{:} S) \rightarrow T$ and unrestricted functions to $(x \overset{?}{:} S) \rightarrow T$. This restriction is reflected in one of the typing rules for function application.

$$\frac{\Gamma_1 \vdash M \overset{?}{:} (x \overset{\pi}{:} S) \rightarrow T \quad \Gamma_2 \vdash N \overset{!}{:} S}{\Gamma_1 + \sigma\pi\Gamma_2 \vdash M N \overset{?}{:} T[N/x]} \rightarrow\text{E}_1$$

The rule follows a naming scheme used throughout this work: a type, a class (formation, introduction, or elimination), and, if necessary, an index. The variable usages from the function and the argument are added together. Additionally, since the function can use its parameter π -times, the usage from the argument needs to be scaled by π . Scaling of the context by π , $\pi\Gamma$, is done by multiplying the multiplicity of each variable by π . Adding two contexts, $\Gamma_1 + \Gamma_2$, is done by adding the multiplicities of matching variables.

3.3 Weakening

In this system, each expression has a unique variable usage. In some situations, such rigidity is undesirable. In linear type theories, unrestricted variables can be used any amount of times, including once. As mentioned previously, we would like to use ω from the $\{0, 1, \omega\}$ semiring to represent unrestricted use. The system thus needs to be able to treat the multiplicities 0 and 1 as ω , which can be accomplished by adding an ordering to the semiring.

The ordering, $\leq$, needs to be reflexive, transitive, and needs to respect the semiring operations. A suitable ordering for our semiring is defined by $0 \leq \omega$

and $1 \leq \omega$. $0 \leq 1$ is not present since it would allow us to treat unused variables as linear. We can now extend this ordering to typing contexts: $\Gamma_1 \leq \Gamma_2$ iff the multiplicity of each variable in Γ_1 is less than or equal to the multiplicity of the matching variable in Γ_2 . The weakening rule states that relaxing a typing context does not change the validity of the judgment.

$$\frac{\Gamma_1 \vdash M \overset{?}{:} T \quad \Gamma_1 \leq \Gamma_2}{\Gamma_2 \vdash M \overset{?}{:} T} \text{W}_{\text{EAK}}$$

4 Multiplicative Types

4.1 Multiplicative Pairs

When introducing a multiplicative pair, the typing context is split into two parts and each part is then used to construct one element of the pair. When eliminating a multiplicative pair, both elements of the pair must be used to ensure no resources are discarded.

In QTT, the multiplicative pair type $(x \overset{?}{:} S) \otimes T$ additionally contains multiplicity annotation π which specifies how many times can the first element be used. The type of the second element can also depend on the value of the first element. The following rules summarize formation, introduction, and elimination. The constant $\mathcal{U}$ is the type of types.

$$\begin{array}{c} \frac{0\Gamma \vdash S \overset{0}{:} \mathcal{U} \quad 0\Gamma, x \overset{0}{:} S \vdash T \overset{0}{:} \mathcal{U}}{0\Gamma \vdash (x \overset{?}{:} S) \otimes T \overset{0}{:} \mathcal{U}} \otimes\text{-F} \\[2ex] \frac{\frac{\sigma\pi=0 \quad 0\Gamma \vdash M \overset{0}{:} S}{\Gamma \vdash N \overset{?}{:} T[M/x]} \otimes\text{-I}_0 \quad \frac{\Gamma_1 \vdash M \overset{1}{:} S \quad \Gamma_2 \vdash N \overset{?}{:} T[M/x]}{\sigma\pi\Gamma_1 + \Gamma_2 \vdash (M, N) \overset{?}{:} (x \overset{?}{:} S) \otimes T} \otimes\text{-I}_1 \\[2ex] \frac{\Gamma_1 \vdash M \overset{\sigma}{:} (x \overset{\pi}{:} S) \otimes T \quad 0\Gamma_1, z \overset{0}{:} (x \overset{\pi}{:} S) \otimes T \vdash U \overset{0}{:} \mathcal{U} \quad \Gamma_2, x \overset{\sigma}{:} S, y \overset{\sigma}{:} T \vdash N \overset{\sigma}{:} U[(x, y)/z]}{\Gamma_1 + \Gamma_2 \vdash \mathbf{let} (x, y) = M \mathbf{in} N \overset{?}{:} U[M/z]} \otimes\text{-E} \end{array}$$

4.2 Multiplicative Unit

Unit types also come in two variants. The multiplicative unit uses no resources and although it cannot be discarded, it can always be eliminated.

$$\begin{array}{c} \frac{0\Gamma \vdash}{0\Gamma \vdash \mathbf{1} \overset{0}{:} \mathcal{U}} \mathbf{1}\text{-F} \quad \frac{0\Gamma \vdash}{0\Gamma \vdash () \overset{?}{:} \mathbf{1}} \mathbf{1}\text{-I} \\[2ex] \frac{\Gamma_1 \vdash M \overset{?}{:} \mathbf{1} \quad 0\Gamma_1, x \overset{0}{:} \mathbf{1} \vdash S \overset{0}{:} \mathcal{U}}{\Gamma_1 + \Gamma_2 \vdash \mathbf{let} () = M \mathbf{in} N \overset{?}{:} S[M/x]} \mathbf{1}\text{-E} \end{array}$$

4.3 Exponential Modality

Multiplicative units and pairs can be combined to form the exponential type $!_{\pi}S$, which is defined as $(- \stackrel{\pi}{\vdash} S) \otimes \mathbf{1}$. Values of this type are introduced and eliminated by combining the introductions and eliminations of the constituent parts. Introduction, $!x$, is defined as $(x, ())$. Elimination, **let** $!x = M$ **in** N , is defined as **let** $(x, i) = M$ **in** **let** $() = i$ **in** N .

4.4 Extended Eliminator

However, the multiplicative pair eliminator does not interact well with weakening. Suppose the typing context contains $p \stackrel{\varpi}{\vdash} (- \stackrel{\dagger}{\vdash} S) \otimes T$ (abbreviated $S \otimes T$). Unless we want to eliminate p multiple times, we first need to apply the weakening rule to obtain $p \stackrel{\dagger}{\vdash} S \otimes T$. Now the elimination **let** $(x, y) = p$ **in** N can be used, which adds the variables x and y to the context. Notice that their multiplicities are 1 and thus the variables cannot be duplicated or discarded.

The expression **let** $(x, y) = p$ **in** (y, y) does not seem to violate any resource constraints. Indeed, if p is unrestricted, so should be its elements. The only way around this problem is to know beforehand that the elements of p will be used in this way and change the type to $!_{\omega}S \otimes !_{\omega}T$.

We solve this problem by adding a *multiplicity annotation* to the eliminator, allowing it to consume the pair more than once. To ensure no resources are discarded, the multiplicity is propagated to the variables representing the elements of the pair. In the example, the weakening rule can now be used on x instead of p , allowing us to discard it in the expression (y, y) .

$$\frac{\Gamma_1 \vdash M \stackrel{\sigma}{\vdash} (x \stackrel{\pi}{\vdash} S) \otimes T \quad \frac{0\Gamma_1, z \stackrel{0}{\vdash} (x \stackrel{\pi}{\vdash} S) \otimes T \vdash U \stackrel{0}{\vdash} \mathcal{U} \quad \Gamma_2, x \stackrel{\rho}{\vdash} \sigma \stackrel{\rho\sigma}{\vdash} S, y \stackrel{\rho\sigma}{\vdash} T \vdash N \stackrel{\sigma}{\vdash} U[(x, y)/z]}{\rho\Gamma_1 + \Gamma_2 \vdash \mathbf{let}_{\rho} (x, y) = M \mathbf{in} N \stackrel{\sigma}{\vdash} U[M/z]} \otimes\text{-E}$$

Since multiplicative pairs have a single introduction, the eliminator does not provide any computationally relevant information apart from the two elements. In particular, the eliminator functions properly even when the annotation is zero.

With this modification, the exponential type can be introduced as needed and does not have to be added beforehand. Indeed, we can easily transform ω copies of $S \otimes T$ into a single copy of $!_{\omega}S \otimes !_{\omega}T$ with the following function.

$$\vdash \lambda p. \mathbf{let}_{\omega} (x, y) = p \mathbf{in} (!x, !y) \stackrel{\dagger}{\vdash} (- \stackrel{\varpi}{\vdash} S \otimes T) \rightarrow !_{\omega}S \otimes !_{\omega}T$$

In fact, using this extended eliminator, we can show that $!_{\omega}(S \otimes T)$ and $!_{\omega}S \otimes !_{\omega}T$ are isomorphic.

$$\begin{aligned} &\vdash \lambda p'. \mathbf{let} !p = p' \mathbf{in} \mathbf{let}_{\omega} (x, y) = p \mathbf{in} (!x, !y) \\ &\quad \stackrel{\dagger}{\vdash} (- \stackrel{\dagger}{\vdash} !_{\omega}(S \otimes T)) \rightarrow !_{\omega}S \otimes !_{\omega}T \\ &\vdash \lambda p. \mathbf{let}_1 (x', y') = p \mathbf{in} \mathbf{let} !x = x' \mathbf{in} \mathbf{let} !y = y' \mathbf{in} !(x, y) \\ &\quad \stackrel{\dagger}{\vdash} (- \stackrel{\dagger}{\vdash} !_{\omega}S \otimes !_{\omega}T) \rightarrow !_{\omega}(S \otimes T) \end{aligned}$$

These definitions can also be used with dependent pairs. However, we have to account for the change of the type of the first element. We can construct the type $!_{\omega}((x \vdash S) \otimes T)$ under the assumption $x \vdash S \vdash T \vdash \mathcal{U}$. However, if the type of the first element is $!_{\omega}S$, it needs to be unwrapped before it can be used in T . The second type is then $(x \vdash !_{\omega}S) \otimes !_{\omega}T[(\text{let } !y = x \text{ in } y)/x]$.

We can also add the multiplicity annotation to the unit eliminator. In the $\{0, 1, \omega\}$ semiring, the only useful annotation is 1, which corresponds to the regular eliminator. If the context contains ω copies of a unit, no elimination is necessary due to weakening. This annotation can be useful with other semirings.

5 Additive Types

5.1 Additive Unions

Disjoint unions are an ideal example of an additive type. When constructing a value of this type, the typing context is passed without any changes. Disjoint unions have two introductions and one elimination, which performs case analysis.

$$\begin{array}{c}
 \frac{0\Gamma \vdash S \vdash \mathcal{U} \quad 0\Gamma \vdash T \vdash \mathcal{U}}{0\Gamma \vdash S \oplus T \vdash \mathcal{U}} \oplus\text{-F} \\
 \\
 \frac{\Gamma \vdash M \vdash S \quad 0\Gamma \vdash T \vdash \mathcal{U}}{\Gamma \vdash \text{inl } M \vdash S \oplus T} \oplus\text{-I}_1 \quad \frac{\Gamma \vdash M \vdash T \quad 0\Gamma \vdash S \vdash \mathcal{U}}{\Gamma \vdash \text{inr } M \vdash S \oplus T} \oplus\text{-I}_2 \\
 \\
 \frac{\rho \neq 0 \quad \Gamma_1 \vdash M \vdash S \oplus T \quad 0\Gamma_1, z \vdash S \oplus T \vdash U \vdash \mathcal{U} \quad \Gamma_2, x \vdash S \vdash N \vdash U[\text{inl } x/z] \quad \Gamma_2, y \vdash T \vdash O \vdash U[\text{inr } y/z]}{\rho\Gamma_1 + \Gamma_2 \vdash \text{case}_{\rho} M \text{ of } \{\text{inl } x \rightarrow N, \text{inr } y \rightarrow O\} \vdash U[M/z]} \oplus\text{-E}
 \end{array}$$

The eliminator is annotated with multiplicity, just like in the case of multiplicative pairs. The requirement that $\rho \neq 0$ is crucial. In that case, although the elements of the disjoint union cannot be used in a relevant context, the union as a whole carries computationally relevant information: the choice of introduction.

We can use the extended eliminator to show that $!_{\omega}(S \oplus T)$ and $!_{\omega}S \oplus !_{\omega}T$ are isomorphic. Like before, only one direction requires this extension.

$$\begin{array}{l}
 \vdash \lambda s'. \text{let } !s = s' \text{ in case}_{\omega} s \text{ of } \{\text{inl } x \rightarrow \text{inl } !x, \text{inr } y \rightarrow \text{inr } !y\} \\
 \vdash (- \vdash !_{\omega}(S \oplus T)) \rightarrow !_{\omega}S \oplus !_{\omega}T \\
 \vdash \lambda s. \text{case}_1 s \text{ of } \{\text{inl } x' \rightarrow \text{let } !x = x' \text{ in } \text{inl } !x, \text{inr } y' \rightarrow \text{let } !y = y' \text{ in } \text{inr } !y\} \\
 \vdash (- \vdash !_{\omega}S \oplus !_{\omega}T) \rightarrow !_{\omega}(S \oplus T)
 \end{array}$$

5.2 Additive Zero

The empty type has no introductions and a single elimination.

$$\frac{0\Gamma \vdash}{0\Gamma \vdash \perp : \mathcal{U}} \perp\text{-F} \qquad \frac{\Gamma \vdash M \text{?} \perp \quad 0\Gamma, x \text{?} \perp \vdash S \text{?} \mathcal{U}}{\Gamma \vdash \mathbf{case} M \text{ of } \{\} \text{?} S[M/x]} \perp\text{-E}$$

5.3 Additive Pairs

Defining the additive version of a pair type presents a problem. We cannot split the resources but we still need to construct both elements of the pair. The only option is to give the same resources to both elements, which seemingly results in resource duplication. We can avoid such duplication by restricting the eliminator to only ever access one of the elements.

Such a type can be constructed with the tools we already have. Since the eliminator of unions shares the typing context between both cases, we can use it to introduce an additive pair. The content of the union is not important, only the introduction is. We thus use the simplest union type: $\mathbf{1} \oplus \mathbf{1}$ (written as $\mathbf{2}$).

To allow the elements of the pair to have different types, we need to use a dependent type for the result of the elimination. We thus define the type of additive pairs $S \& T$ as $(b : \mathbf{2}) \rightarrow \mathbf{case}_1 b \text{ of } \{\mathbf{inl} \ t \rightarrow S, \mathbf{inr} \ f \rightarrow T\}$.

Introduction, $\langle x, y \rangle$, is defined as $\lambda b. \mathbf{case}_1 b \text{ of } \{\mathbf{inl} \ t \rightarrow x, \mathbf{inr} \ f \rightarrow y\}$. We also need to consume the units t and f , but the eliminators are left out for brevity. Eliminations, $\mathbf{fst} M$ and $\mathbf{snd} M$, are defined as $M(\mathbf{inl}())$ and $M(\mathbf{inr}())$. Importantly, we cannot extract both elements of an additive pair without applying the function multiple times. This restriction guarantees resource soundness.

However, while this encoding has the necessary properties to qualify as an additive pair, it is not without issues. In dependent type theories, we often use the identity type to ask whether two values are equal. Functions are notoriously hard to compare, since these theories often lack the function extensionality principle.

Moreover, the definition of $S \& T$ does not allow any dependency between S and T . Both types are on separate branches of the additive union eliminator and thus must be independent. Notice that dependency between S and T is, in principle, possible. Even though we are only allowed to access one element of the pair in a relevant context, this restriction does not extend to irrelevant contexts.

Instead of using a combination of unions and functions, additive pairs, $(x : S) \& T$, are defined explicitly. The variable x can occur in the type T . We use the same syntax for introduction, $\langle x, y \rangle$, and elimination, $\mathbf{fst} M$ and $\mathbf{snd} M$.

$$\frac{0\Gamma \vdash S \text{?} \mathcal{U} \quad 0\Gamma, x \text{?} S \vdash T \text{?} \mathcal{U}}{0\Gamma \vdash (x : S) \& T \text{?} \mathcal{U}} \&\text{-F} \qquad \frac{\Gamma \vdash M \text{?} S \quad \Gamma \vdash N \text{?} T[M/x]}{\Gamma \vdash \langle M, N \rangle \text{?} (x : S) \& T} \&\text{-I}$$

$$\frac{\Gamma \vdash M \text{?} (x : S) \& T}{\Gamma \vdash \mathbf{fst} M \text{?} S} \&\text{-E}_1 \qquad \frac{\Gamma \vdash M \text{?} (x : S) \& T}{\Gamma \vdash \mathbf{snd} M \text{?} T[\mathbf{fst} M/x]} \&\text{-E}_2$$

5.4 Additive Unit

The multiplicative unit can be viewed as a neutral element to multiplicative pairs. Indeed, the types $(x : S) \otimes \mathbf{1}$ and S are isomorphic. Similarly, an additive unit, $\top$, should be a neutral element to additive pairs.

One side of the isomorphism is trivial. When given $p : S \& \top$, we can extract a value of type S by using **fst** p . The other direction presents a problem. When given $s : S$, we need to construct a pair $\langle s, \langle \rangle \rangle$ where $\langle \rangle$ is the additive unit introduction. However, the use of the variable s is inconsistent between the two elements. The introduction $\langle \rangle$ must thus count as a use of the variable s . The additive unit represents some unknown, but unused resources. And since there is no general way to consume unknown resources, the unit must not be eliminated.

We can define $\top$ as a function $(e : \perp) \rightarrow \mathbf{case} \ e \ \mathbf{of} \ \{\}$. A value of the empty type can be used to consume any variable, for example by eliminating it into a function and then applying that function to the desired variables. This function cannot be eliminated as it would require us to provide a value of the empty type.

We can also define the type $\top$ and its introduction $\langle \rangle$ explicitly. Notice the introduction works in any context Γ , not just 0Γ . No elimination is present.

$$\frac{0\Gamma \vdash}{0\Gamma \vdash \top : \mathcal{U}} \top\text{-F} \qquad \frac{0\Gamma \vdash}{\Gamma \vdash \langle \rangle : \top} \top\text{-I}$$

5.5 Usage

Since an additive zero, multiplicative pairs, and additive unions are commonly used types in functional programming, we focus on possible uses of an additive unit and additive pairs. Additionally, we restrict ourselves to linear uses of these types. If values of these types can be discarded or duplicated, they are functionally equivalent to the normal, intuitionistic pairs and units.

The uses of the additive unit are rather limited. Once a value is introduced, it can never be removed. Using it to signal some kind of resource error is also not ideal, since such errors cannot be handled. However, because the additive unit is a neutral element to additive pairs, it can be used as the base case for type operations. Suppose we have a type of lists, **List**, together with two introductions, **Cons** and **Nil**, and an eliminator, **foldr**. When given a list of types, we can construct an *additive tuple* of these types as follows.

$$\vdash \mathbf{foldr} \ \mathcal{U} \ \mathcal{U} \ (\lambda T \ R. (\_ : T) \& R) \ \top : (\_ : \mathbf{List} \ \mathcal{U}) \rightarrow \mathcal{U}$$

In functional programming, higher-order functions are frequently used in place of control flow statements. Take for example the **if** function. It takes a value of type **2**, two values of some type T and returns a value of type T . We make the function polymorphic by quantifying over the type T , which gives us the type $(T : \mathcal{U}) \rightarrow \mathbf{2} \rightarrow T \rightarrow T \rightarrow T$.

If we attempt to mark the other parameters with multiplicities, we soon run into a problem. The parameter of type **2** is used exactly once, but of the two

following parameters, only one is used and the other is discarded. One possible solution is to use the multiplicity ω and rely on weakening to discard one of the values. This change severely limits this function's usefulness in linear contexts.

The requirement that exactly one value must be used can be expressed using an additive pair, which allows us to give the **if** function a fully linear type: $(T \multimap \mathcal{U}) \rightarrow (b \multimap \mathbf{2}) \rightarrow (c \multimap (\_ : T) \& T) \rightarrow T$. Indeed, whenever we need to express this kind of 1-of- n requirement, we can use the aforementioned additive tuple.

A *paramorphism* is a recursion scheme where the function handling the recursive step has access to the result of the recursion and the remainder of the original structure. To ensure the elements of a given structure are consumed once, the combining function must either use the recursive result or the remainder, but not both. Since the base case of the recursion might not be reached, we must be allowed to discard it. We define the list paramorphism, **para**, via **foldr**.

$$\begin{aligned} &\vdash \lambda S T c n l. \mathbf{snd} (\mathbf{foldr} S ((\_ : \mathbf{List} S) \& T) \\ &\quad (\lambda a b. \langle \mathbf{Cons} a (\mathbf{fst} b), c a b \rangle) \langle \mathbf{Nil}, n \rangle l) \\ &\quad \varpi (S \multimap \mathcal{U}) \rightarrow (T \multimap \mathcal{U}) \rightarrow (c \varpi (a \multimap S) \rightarrow (b \multimap (\_ : \mathbf{List} S) \& T) \rightarrow T) \rightarrow \\ &\quad (n \varpi T) \rightarrow (l \multimap \mathbf{List} S) \rightarrow T \end{aligned}$$

Note that in practice, **para** would either be given directly or implemented recursively. The **foldr** implementation needlessly reconstructs the original list.

We can use this recursion scheme to implement an insertion operation for sorted lists. Once we find the point of insertion, we do not need to traverse through the rest of the list and we can use the remainder. At first glance, this operation cannot be linear: the elements need to be compared and then used in the resulting list. To solve this problem, we can use a comparison operation that also returns the compared values, such as **leq** $\varpi (p \multimap A \otimes A) \rightarrow \mathbf{2} \otimes A \otimes A$.

$$\begin{aligned} &\vdash \lambda v l. \mathbf{para} A ((\_ \multimap A) \rightarrow \mathbf{List} A) \\ &\quad (\lambda a w i. \mathbf{let}_1 (b, p) = \mathbf{leq} (a, i) \mathbf{in} \mathbf{let}_1 (a', i') = p \mathbf{in} \\ &\quad \quad \mathbf{if} (\mathbf{List} A) b \langle \mathbf{Cons} a' (\mathbf{snd} w i'), \mathbf{Cons} i' (\mathbf{Cons} a' (\mathbf{fst} w)) \rangle) \\ &\quad (\lambda i. \mathbf{Cons} i \mathbf{Nil}) l v \varpi (v \multimap A) \rightarrow (l \multimap \mathbf{List} A) \rightarrow \mathbf{List} A \end{aligned}$$

A dependent additive pair type can be used to generalize the type of **para** and allow the type of the result to depend on the input list. This function can be defined via a dependent **foldr**, but the implementation has to carry a proof that the original and reconstructed lists are the same, and is skipped for brevity.

$$\begin{aligned} &\vdash \mathbf{para} \varpi (S \multimap \mathcal{U}) \rightarrow (T \multimap (l \multimap \mathbf{List} S) \rightarrow \mathcal{U}) \rightarrow \\ &\quad (c \varpi (a \multimap S) \rightarrow (b \multimap (r : \mathbf{List} S) \& T r) \rightarrow T (\mathbf{Cons} a (\mathbf{fst} b))) \rightarrow \\ &\quad (n \varpi T \mathbf{Nil}) \rightarrow (l \multimap \mathbf{List} S) \rightarrow T l \end{aligned}$$

6 Bidirectional Type System

In contrast with the declarative presentation of the typing rules, there also exists the *bidirectional* form, which reduces the number of explicit type annotations

the user of a language with these typing rules is expected to provide, thus the implementation of the typing algorithm is based on a bidirectional form of the rules.

In bidirectional presentation of the typing rules, judgments come in two types: the *type checking* judgment, $\Gamma \vdash M \Leftarrow^c S$, and the *type synthesising* judgment, $\Gamma \vdash M \Rightarrow^s S$. The difference is that in type checking the type S is an input of the judgment and the derivation proves that the term M can indeed be of type S ; whereas, in type synthesis the type S is an output derived from the term M .

Every term M that synthesises type S can also have its type checked using S , thus, in bidirectional formulation of typing rules, we need to decide which terms can have their type synthesised and which terms can only have its type checked. In our work we follow the Pfenning recipe [7]:

If the rule is an *introduction* rule, make the principal judgment *checking* and if the rule is an *elimination* rule, make the principal judgment *synthesising*.

A *principal judgment* is the judgment containing the logical connective that is being introduced or eliminated.

In the bidirectional form, the order in which the premises are derived is important, because one judgment may synthesise a type that is later used in a different judgment. We derive them from left to right. For example, in the following rule we first synthesise the type $(x \vdash S) \rightarrow T$ from M and only then can we use the S to check N with.

$$\frac{\Gamma_1 \vdash M \Rightarrow^s (x \vdash S) \rightarrow T \quad \Gamma_2 \vdash N \Leftarrow^c S}{\Gamma_1 + \sigma\pi\Gamma_2 \vdash M N \Rightarrow^s T[N/x]} \rightarrow\text{-E}_1$$

The rules above are translated into their bidirectional form using the recipe. We introduce an additional type-synthesising term, $M : S$, and an associated rule that forms it from two type-checking terms, M and S .

$$\frac{0\Gamma \vdash S \Leftarrow^0 \mathcal{U} \quad \Gamma \vdash M \Leftarrow^c S}{\Gamma \vdash M : S \Rightarrow^s S} \text{ANN}$$

7 Implementation

This work is accompanied by an interpreter of a language based on hitherto described extensions to QTT. To implement an efficient typing algorithm, the typing rules had to be modified somewhat, as we describe below.

7.1 Language Interpreter

The accompanying implementation is written in Haskell and, while originally based on the dependently typed λ -calculus interpreter *LambdaPi* [9], has had all its parts rewritten from scratch, making use of modern Haskell packages:

parser is written using the parser combinator library *Megaparsec*; text-based user interface is based on the *Haskeline* library, wrapped in *Repline*; and the pretty-printer uses the *Prettyprinter* package.

The Haskell modules that constitute our implementation are documented throughout, using Haddock-style comments.

7.2 Typing Algorithm

To translate the typing rules into an efficient algorithm, we would need to have a way of a priori deciding how to split the resources in a given context when deriving multiple premises; for example, in the rule $\rightarrow\text{-E}_1$, we need to split the context $\Gamma_1 + \sigma\pi\Gamma_2$ into Γ_1 and Γ_2 . Instead, we consider an additional output for our judgments: recorded resource *usage* of the judged term. Usage accumulates the resource consumption from throughout the term and the typing algorithm uses it to check that an appropriate amount of every resource is available. Thus, in the example of rule $\rightarrow\text{-E}_1$, there is no need to know how to split $\Gamma_1 + \sigma\pi\Gamma_2$, instead the premises produce usages Ψ from M and Ω from N , which get combined into $\Psi + \Omega$ and returned as the usage of $M N$. The addition on usages works analogously to contexts.

To judge that the resource usage of a term is appropriate for the given context, we check that the collected usage of a variable is less than or equal to, with regards to the given semiring ordering $\leq$, the multiplicity with which it occurs in the context of a judgment. For *global variables*, i.e. variables that are free in the whole term, this check is done at the end of the typing algorithm, but for the *local variables*, i.e. variables that have a binding occurrence somewhere in the term, the check is done after accumulating its usage from its scope.

Since the usage check takes the weakening into consideration, we no longer need the explicit **WEAK** rule. Consequently, our typing rules almost completely *syntax-directed*; that is, the derivation rule which can be used is determined by the syntax of the term. There are two exceptions to this: first, when constructing a multiplicative pair, we first check if $\sigma\pi = 0$ and use the rule $\otimes\text{-I}_0$, otherwise we use $\otimes\text{-I}_1$; and second, when applying a function, we synthesise the type of the function first and then check if $\sigma\pi = 0$, using the rule $\rightarrow\text{-E}_0$ if true and $\rightarrow\text{-E}_1$ otherwise.

8 Conclusion

Additive types are an important part of substructural type systems. Indeed, one of the most commonly used types in functional programming is the disjoint union, which is typically only available in the additive version. However, other additive types are used rather sparingly. QTT allows us to explore these types in a fully dependent setting.

We present an extended theory containing an additive unit and dependent additive pairs. Thanks to the annotated eliminators, the theory is also more expressive in the presence of the weakening rule. A variety of examples show

how these extensions may be used. We demonstrate the viability of the theory by providing a bidirectional type system and its implementation in an interpreter.

We hope that this work encourages both existing and future treatments of QTT to consider these additive types as first class citizens, rather than relying on an inconvenient and limited encoding via functions.

References

1. Atkey, R.: Syntax and semantics of quantitative type theory. In: Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, pp. 56–65. Association for Computing Machinery, New York (2018). <https://doi.org/10.1145/3209108.3209189>
2. Brady, E.: Idris 2: quantitative type theory in practice. In: Möller, A., Sridharan, M. (eds.) 35th European Conference on Object-Oriented Programming (ECOOP 2021). Leibniz International Proceedings in Informatics (LIPIcs), vol. 194, pp. 9:1–9:26. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Dagstuhl (2021). <https://doi.org/10.4230/LIPIcs.ECOOP.2021.9>
3. Brady, E., et al.: Idris2 (2022). <https://github.com/idris-lang/Idris2>
4. Brunel, A., Gaboardi, M., Mazza, D., Zdancewic, S.: A core quantitative coefficient calculus. In: Shao, Z. (ed.) ESOP 2014. LNCS, vol. 8410, pp. 351–370. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54833-8_19
5. Cervesato, I., Pfenning, F.: A linear logical framework. *Inf. Comput.* **179**(1), 19–75 (2002). <https://doi.org/10.1006/inco.2001.2951>
6. Choudhury, P., Eades, H., Eisenberg, R., Weirich, S.: A graded dependent type system with a usage-aware semantics. *Proc. ACM Program. Lang.* **5**(POPL) (2021). <https://doi.org/10.1145/3434331>
7. Dunfield, J., Krishnaswami, N.: Bidirectional typing. *ACM Comput. Surv.* **54**(5) (2021). <https://doi.org/10.1145/3450952>
8. Krishnaswami, N., Pradic, P., Benton, N.: Integrating linear and dependent types. *SIGPLAN Not.* **50**(1), 17–30 (2015). <https://doi.org/10.1145/2775051.2676969>
9. Löh, A., McBride, C., Swierstra, W.: A tutorial implementation of a dependently typed lambda calculus. *Fund. Inform.* **102**(2), 177–207 (2010). <https://doi.org/10.3233/fi-2010-304>
10. McBride, C.: I got plenty o’Nuttin’. In: Lindley, S., McBride, C., Trinder, P., Sannella, D. (eds.) *A List of Successes That Can Change the World*. LNCS, vol. 9600, pp. 207–233. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-30936-1_12
11. The Anoma Team: Juvix (2022). <https://github.com/anoma/juvix>



Strongly First Order, Domain Independent Dependencies: The Union-Closed Case

Pietro Galliani^(✉) 

Free University of Bozen-Bolzano, Bolzano, Italy
pietro.galliani@unibz.it

Abstract. Team Semantics generalizes Tarski’s Semantics by defining satisfaction with respect to sets of assignments rather than with respect to single assignments. Because of this, it is possible to use Team Semantics to extend First Order Logic via new kinds of connectives or atoms – most importantly, via *dependency atoms* that express dependencies between different assignments.

Some of these extensions are more expressive than First Order Logic proper, while others are reducible to it. In this work, I provide necessary and sufficient conditions for a dependency atom that is *domain independent* (in the sense that its truth or falsity in a relation does not depend on the existence in the model of elements that do not occur in the relation) and *union closed* (in the sense that whenever it is satisfied by all members of a family of relations it is also satisfied by their union) to be *strongly first order*, in the sense that the logic obtained by adding them to First Order Logic is no more expressive than First Order Logic itself.

Keywords: Team semantics · Dependence logic · First order logic

1 Introduction

Team Semantics is a generalization of Tarskian Semantics in which formulas are satisfied or not satisfied by sets of assignments (called *teams*) rather than by single assignments. This semantics was developed by Hodges in [18] as a compositional alternative to the game-theoretic semantics for Independence-Friendly Logic [16], an extension of first order logic (related to *Branching Quantifier Logic* [15]) that adds to it *slashed quantifiers* $\exists x/\vec{y}$, with the meaning of “there exists some x , chosen independently from all variables in $\vec{y}$, such that...”.

As observed by Jouko Väänänen, in Team Semantics this notion of dependence/independence between variables corresponds precisely to database-theoretic *functional dependence* [1]. This observation led to the development of *Dependence Logic* [24], which separates the notion of dependency restriction from the notion of quantification by means of (functional) *dependence atoms* $= (\vec{y}, x)$ meaning “the value of x is *determined by* the values of $\vec{y}$ ”; and, later, to the development of an entire family of novel extensions of First Order Logic likewise obtained by adding to it new atoms to express forms of dependence/independence between variables.

The properties of some of the logics thus generated are, by now, fairly well studied; but much is still not known about the general classification of the logics obtainable from First Order Logic in this manner. In particular, no full answer yet exists to the following question: which dependencies between variables, if added to First Order Logic as new atoms, generate a logic that is more expressive than First Order Logic, and which ones instead fail to do so? Aside from being a natural starting point for a general classification of these extensions of First Order Logic, this question is of philosophical interest: after a fashion, a full answer to it would clarify how much we can get away with adding to First Order Logic (in terms of novel atoms specifying dependencies between variables) before the resulting logic becomes unrecoverably second-order in nature.

This paper continues a series of works [5–10] that attempt to solve this problem with respect to restricted classes of dependencies, as part of a program to eventually reach a general characterization. In particular, here we will focus on dependencies that have the two properties of being *domain-independent* and *union-closed*, which are shared by many of the dependencies of interest in the literature, and provide a full solution to the problem for their case.

2 Preliminaries

Definition 1 (Team). *Let $\mathfrak{M}$ be a first order model with domain M and let V be a set of variables. Then a team over $\mathfrak{M}$ with domain V is a set of assignments $s : V \rightarrow M$. Given such a team X and some tuple of variables $\vec{v} = (v_1 \dots v_k) \in V^k$, we will write $X(\vec{v})$ for the $|\vec{v}|$ -ary relation $\{s(\vec{v}) : s \in X\} \subseteq M^{|\vec{v}|}$, where $s(\vec{v})$ is the tuple $(s(v_1) \dots s(v_k))$.*

Definition 2 (Team Semantics for First Order Logic). *Let $\mathfrak{M}$ be a first order model with domain M , let ϕ be a first order formula in Negation Normal Form¹ over the signature of $\mathfrak{M}$, and let X be a team over $\mathfrak{M}$ whose domain contains the free variables of ϕ . Then we say that X satisfies ϕ in $\mathfrak{M}$, and we write $\mathfrak{M} \models_X \phi$, if this follows from the following rules:*

- TS-lit:** *If ϕ is a first order literal, $\mathfrak{M} \models_X \phi$ if and only if, for all assignments $s \in X$, $\mathfrak{M} \models_s \phi$ in the usual sense of Tarskian Semantics;*
- TS- $\vee$:** *$\mathfrak{M} \models_X \phi_1 \vee \phi_2$ if and only if $X = Y \cup Z$ for two $Y, Z \subseteq X$ such that $\mathfrak{M} \models_Y \phi_1$ and $\mathfrak{M} \models_Z \phi_2$;*
- TS- $\wedge$:** *$\mathfrak{M} \models_X \phi_1 \wedge \phi_2$ if and only if $\mathfrak{M} \models_X \phi_1$ and $\mathfrak{M} \models_X \phi_2$;*
- TS- $\exists$:** *$\mathfrak{M} \models_X \exists v \psi$ if and only if there exists some function² $H : X \rightarrow \mathcal{P}(M) \setminus \{\emptyset\}$ such that $\mathfrak{M} \models_{X[H/v]} \psi$, where $X[H/v] = \{s[m/v] : s \in X, m \in H(s)\}$;*
- TS- $\forall$:** *$\mathfrak{M} \models_X \forall v \psi$ if and only if $\mathfrak{M} \models_{X[M/v]} \psi$, where $X[M/v] = \{s[m/v] : s \in X, m \in M\}$.*

¹ In this work we will assume that all expressions are in Negation Normal Form. Also, we will use the usual abbreviations of First Order Logic, such as $\vec{x} = \vec{y}$ for $\bigwedge_i (x_i = y_i)$ where $\vec{x} = (x_1 \dots x_k)$ and $\vec{y} = (y_1 \dots y_k)$ are tuples of variables of the same length.

² $\mathcal{P}(M)$ represents the powerset $\{A : A \subseteq M\}$ of M .

A sentence ϕ is true in a model $\mathfrak{M}$ if and only if $\mathfrak{M} \models_{\{\epsilon\}} \phi$, where $\{\epsilon\}$ is the team containing only the empty assignment ϵ . In that case, we write that $\mathfrak{M} \models \phi$.

Over First Order Logic, Team Semantics reduces to Tarskian Semantics:

Proposition 1 ([24], Corollary 3.32). *Let $\mathfrak{M}$ be a first order model, let ϕ be a first order formula in Negation Normal Form over the signature of $\mathfrak{M}$, and let X be a team over $\mathfrak{M}$ whose domain contains the free variables of ϕ . Then $\mathfrak{M} \models_X \phi$ if and only if, for all $s \in X$, $\mathfrak{M} \models_s \phi$ in the sense of Tarskian Semantics.*

In particular, if ϕ is a first order sentence, $\mathfrak{M} \models \phi$ in the sense of Team Semantics if and only if $\mathfrak{M} \models \phi$ in the sense of Tarskian Semantics.

However, the framework of Team Semantics allows one to extend First Order Logic in novel ways, for example via *functional dependence atoms* [24], *inclusion and exclusion atoms* [4], *anonymity atoms* [3, 22, 23]³ or *independence atoms* [13]

- TS- $(\cdot; \cdot)$** : If $\vec{x}$ and $\vec{y}$ are tuples of variables in the domain of X , $\mathfrak{M} \models_X (\vec{x}; \vec{y})$ if and only if, for all $s, s' \in X$, $s(\vec{x}) = s'(\vec{x}) \Rightarrow s(\vec{y}) = s'(\vec{y})$;
- TS- $\subseteq$** : If $\vec{x}$ and $\vec{y}$ are tuple of variables in the domain of X of the same length, $\mathfrak{M} \models_X \vec{x} \subseteq \vec{y}$ if and only if $X(\vec{x}) \subseteq X(\vec{y})$;
- TS- $\mid$** : If $\vec{x}$ and $\vec{y}$ are tuple of variables in the domain of X of the same length, $\mathfrak{M} \models_X \vec{x} \mid \vec{y}$ if and only if $X(\vec{x}) \cap X(\vec{y}) = \emptyset$;
- TS- Υ** : If $\vec{x}$ and $\vec{y}$ are tuples of variables in the domain of X , $\mathfrak{M} \models_X \vec{x} \Upsilon \vec{y}$ if and only if for every $s \in X$ there exists some $s' \in X$ with $s(\vec{x}) = s'(\vec{x})$ but $s(\vec{y}) \neq s'(\vec{y})$;
- TS- $\perp$** : If $\vec{x}$ and $\vec{y}$ are tuples of variables in the domain of X , $\mathfrak{M} \models_X \vec{x} \perp \vec{y}$ if and only if $X(\vec{x}\vec{y}) = X(\vec{x}) \times X(\vec{y})$, i.e., for all $s, s' \in X$ there is some $s'' \in X$ with $s''(\vec{x}) = s(\vec{x})$ and $s''(\vec{y}) = s'(\vec{y})$.

The logics obtained by adding these dependencies to First Order Logic **FO** are all more expressive than it: in particular, (Functional) Dependence Logic **FO**($(\cdot; \cdot)$), Exclusion Logic **FO**($\mid$) and Independence Logic **FO**($\perp$) are all equivalent to⁴ Existential Second Order Logic [4, 12, 13, 24], while Inclusion Logic **FO**($\subseteq$) and Anonymity Logic **FO**(Υ) are both equivalent to the positive fragment of Greatest Fixpoint Logic [3, 11]. On the other hand, the *nonemptiness atoms*

TS-NE: $\mathfrak{M} \models_X \text{NE}(\vec{v})$ if and only if⁵ $X(\vec{v}) \neq \emptyset$

³ Anonymity atoms were called “non-dependence atoms” and briefly studied in Sect. 4.6 of [3], in which their equidefinability with inclusion atoms was also proved; but their interpretation and their properties are discussed more in depth in [22, 23].

⁴ In the sense that every sentence of any of these logics is equivalent to some Existential Second Order sentence, and vice versa. As we will see, it is however not the case that every *formula* of **FO**($\perp$) is equivalent to some formula of **FO**($\mid$).

⁵ The choice of $\vec{v}$ in the atom $\text{NE}(\vec{v})$ is indifferent as long as it is contained in the domain of X . Thus, one could define NE as a single 0-ary dependency instead of a family of dependencies of all arities, as we do here for simplicity and uniformity.

do not increase the expressive power of First Order Logic if added to it, as per Theorem 11 of [5],⁶ even though it follows easily from Proposition 1 that such an atom is not definable by any first order formula over Team Semantics.⁷

Extensions of First Order Logic with Team Semantics via additional connectives have also been studied. One such connective that will be of some importance in this work is the *Global* (or *Boolean*) *Disjunction*

TS- $\sqcup$: $\mathfrak{M} \models_X \phi \sqcup \psi$ if and only if $\mathfrak{M} \models_X \phi$ or $\mathfrak{M} \models_X \psi$.

There is by now a considerable amount of research, that cannot be summarized here, concerning the classification of logics based on Team Semantics. One useful tool for such research has been the following general definition of dependency [21]:

Definition 3 (Dependency) *Let $k \in \mathbb{N}$. Then a k -ary dependency $\mathbf{D}$ is a class, closed under isomorphisms, of models $\mathfrak{A} = (A, R)$, where R is a k -ary relation over A . Given a first order model $\mathfrak{M}$ with domain M , a team X over it and a tuple $\vec{x}$ of variables in the domain of X such that $|\vec{x}| = k$,*

TS- $\mathbf{D}$: $\mathfrak{M} \models_X \mathbf{D}\vec{x}$ if and only if $(M, X(\vec{x})) \in \mathbf{D}$.

Given any such dependency $\mathbf{D}$, we write $\mathbf{FO}(\mathbf{D})$ for the logic obtained by adding $\mathbf{D}$ to First Order Logic with Team Semantics; and likewise, if $\mathcal{D}$ is a family of such dependencies we write $\mathbf{FO}(\mathcal{D})$ for the logic obtained by adding all dependencies in $\mathcal{D}$ to First Order Logic with Team Semantics.

This definition is somewhat more general than the examples given so far in that it allows an instance $\mathbf{D}\vec{x}$ of a dependency $\mathbf{D}$ to be satisfied or not satisfied by a team X over a model $\mathfrak{M}$ depending on the existence in $\mathfrak{M}$ of elements that do not occur in $X(\vec{x})$. In other words, all dependencies mentioned so far (and nearly all other dependencies considered in the literature) are *domain independent* in the sense of the following definition from [19]:⁸

Definition 4 (Domain Independence) *A k -ary dependency $\mathbf{D}$ is domain independent if and only if, for all models $\mathfrak{A} = (A, R)$,*

$$(A, R) \in \mathbf{D} \Leftrightarrow (Fl\bar{d}(R), R) \in \mathbf{D}$$

where $Fl\bar{d}(R) = \bigcup_{i=1}^k \{\vec{a}_i : \vec{a} \in R\}$ is the set of all elements that occur in an position of any tuple of R .

An example of a dependency that is not domain-independent is given by the *Totality Atoms* from [5]

TS-All: $\mathfrak{M} \models_X \mathbf{All}(\vec{x})$ if and only if $X(\vec{x}) = M^{|\vec{x}|}$

⁶ Theorem 1 in this work.

⁷ Of course there is a first order sentence $\exists \vec{x} R\vec{x}$ that says that R is not empty; however, there is no first order formula $\phi(\vec{w})$ such that $\mathfrak{M} \models_X \phi(\vec{w})$ if and only if $X(\vec{w}) \neq \emptyset$.

⁸ In [19], this property is called “Universe Independence”.

that state that a certain tuple of variables $\vec{x}$ takes *all* possible values in the current team. Domain independence is a very natural condition that is satisfied by the majority of the dependencies studied in the literature, and it can be argued that it should be included in the definition of a “proper” dependency.

All dependencies mentioned so far, including totality, are also *First Order* in the sense that, if seen as properties of relations, they are first order definable:

Definition 5 (First Order Dependency) *A k -ary dependency $\mathbf{D}$ is First Order if and only if there exists a first order sentence $\mathbf{D}(R)$ such that*

$$(A, R) \in \mathbf{D} \Leftrightarrow (A, R) \models \mathbf{D}(R)$$

for all models $\mathfrak{A} = (A, R)$, where R is a k -ary relation.

Ultimately, the fact that adding first order dependencies to First Order Logic may lead to logics more expressive than First Order Logic itself is a consequence of the second-order nature of the Team Semantics rules **TS- $\vee$** and **TS- $\exists$** for disjunction and existential quantification.⁹

There are some general classes of dependencies that have also been useful in the study of Team Semantics:

Definition 6 (Empty Team Property, Downwards, Union- and Upwards Closures)

Let $\mathbf{D}$ be a k -ary dependency. Then $\mathbf{D}$

- *has the Empty Team Property if $(A, \emptyset) \in \mathbf{D}$ for all domains A ;*
- *is Downwards Closed if whenever $(A, R) \in \mathbf{D}$ and $S \subseteq R$ we also have that $(A, S) \in \mathbf{D}$;*
- *is Union Closed if whenever $(R_i)_{i \in I}$ is a family of relations over A such that $(A, R_i) \in \mathbf{D}$ for all $i \in I$ then $(A, \bigcup_i R_i) \in \mathbf{D}$ as well;*
- *is Upwards Closed if whenever $(A, R) \in \mathbf{D}$ and $R \subseteq S$ then $(A, S) \in \mathbf{D}$.*

Note that union closure implies the empty team property, because for $I = \emptyset$ we have that $\bigcup_{i \in I} R_i = \emptyset$.

Table 1 summarizes the properties of the dependencies mentioned so far.¹⁰

It can be seen, by structural induction on formulas, that if $\mathcal{D}$ is a family of downwards closed dependencies then all formulas $\phi \in \mathbf{FO}(\mathcal{D})$ are downwards closed, in the sense that if $\mathfrak{M} \models_X \phi$ and $Y \subseteq X$ then $\mathfrak{M} \models_Y \phi$. Likewise, if all dependencies in $\mathcal{D}$ are union closed and $\mathfrak{M} \models_{X_i} \phi \in \mathbf{FO}(\mathcal{D})$ for all $i \in I$, where $(X_i)_{i \in I}$ is a family of teams over $\mathfrak{M}$ with the same domain, then $\mathfrak{M} \models_{\bigcup_{i \in I} X_i} \phi$; and if all dependencies in $\mathcal{D}$ have the empty team property then $\mathfrak{M} \models_{\emptyset} \phi$ for all models $\mathfrak{M}$ and all formulas $\phi \in \mathbf{FO}(\mathcal{D})$ over the signature of $\mathfrak{M}$. This implies at once that, for example, independence atoms are not definable in terms of

⁹ The **FO**T logic of [20], which is a Team Semantics-based logic with no higher order quantifications in the rules for its connectives, is no stronger than First Order Logic and can define all first order dependencies.

¹⁰ **NE** and **All** are not union closed because the union of the empty family of relations is the empty relation, which does not satisfy these dependencies. If we restricted the definition of union closure to nonempty families I , they would be union closed.

Table 1. Closure properties of dependencies

Dependency	Empty team	Downwards	Union	Upwards	Domain Ind.
$=(\cdot; \cdot)$	+	+	−	−	+
$\subseteq$	+	−	+	−	+
$ $	+	+	−	−	+
Υ	+	−	+	−	+
$\perp$	+	−	−	−	+
NE	−	−	−	+	+
All	−	−	−	+	−

exclusion atoms: even though every $\mathbf{FO}(\perp)$ sentence is equivalent to some $\mathbf{FO}(|)$ sentence and vice versa, there exists no $\mathbf{FO}(|)$ formula that is equivalent to the independence atom $\vec{x} \perp \vec{y}$. This discrepancy between definability and expressivity in Team Semantics – i.e., the fact that logics based on Team Semantics may be equiexpressive despite the corresponding atoms not being reciprocally definable – is one of the most intriguing phenomena of Team Semantics.

The property of upwards closure, instead, is not preserved by Team Semantics as first order literals are not upwards closed. However,

Theorem 1 ([5], Theorem 11). *Let $\mathcal{D}$ be a family of upwards closed, first order dependencies. Then every sentence of $\mathbf{FO}(\mathcal{D})$ is equivalent to some first order sentence.*

In other words, the set $\mathcal{D}^\dagger$ of all upwards closed first order dependencies is *strongly first order* in the sense of the following definition:

Definition 7 (Strongly First Order Dependencies). *Let $\mathbf{D}$ be a dependency [let $\mathcal{D}$ be a family of dependencies]. Then $\mathbf{D}$ [$\mathcal{D}$] is strongly first order if and only if every sentence of $\mathbf{FO}(\mathbf{D})$ [$\mathbf{FO}(\mathcal{D})$] is equivalent to some first order sentence.*

It is easy to see that all strongly first order dependencies are first order in the sense of Definition 5, since by the rules of Team Semantics $(M, R) \in \mathbf{D}$ if and only if $(M, R) \models \forall \vec{x} (\neg R\vec{x} \vee (R\vec{x} \wedge \mathbf{D}\vec{x}))$ and if $\mathbf{D}$ is strongly first order this is equivalent to some first order sentence; but this is certainly no sufficient condition, since as we saw there exist first order dependencies that are not strongly first order.

On the other hand, being first order and upwards closed is a sufficient condition for a dependency to be strongly first order, but it is not necessary: for example, as shown already in Sect. 3.2 of [4], *constancy atoms*

TS- $=(\cdot): \mathfrak{M} \models_X =(\vec{x})$ if and only if for all $s, s' \in X$, $s(\vec{x}) = s'(\vec{x})$,

that can be seen as degenerate cases of functional dependencies stating that $\vec{x}$ depends on *nothing* (i.e. its value is determined by the empty tuple of variables), are strongly first order despite not being upwards closed.¹¹ We can say more:

¹¹ We could consider only *unary* constancy atoms, as $=(\vec{x}) \equiv \bigwedge_{x \in \vec{x}} =(\bar{x})$; but, similarly to the NE case, for simplicity and uniformity we define constancy atoms for all arities.

Proposition 2. *Every sentence of $\mathbf{FO}(=(\cdot), \mathcal{D}^\uparrow, \sqcup)$ – i.e. of First Order Logic augmented with constancy atoms, upwards closed first order dependencies, and global disjunctions – is equivalent to some first order sentence.*

Proof. By Theorem 21 of [5], $\mathcal{D}^\uparrow \cup =(\cdot)$ is strongly first order. By Proposition 14 of [8], if $\mathcal{D}$ is a strongly first order family of dependencies, every sentence of $\mathbf{FO}(\mathcal{D}, \sqcup)$ is equivalent to some first order sentence. The result follows at once.

These observations led to the following

Problem: Provide necessary and sufficient conditions for a dependency or a family of dependencies to be strongly first order.
and to the related

Conjecture: A family of dependencies $\mathcal{D}$ is strongly first order if and only if all $\mathbf{D} \in \mathcal{D}$ are strongly first order.

A partial answer was found in [7]:

Theorem 2 ([7], Theorem 4.5). *Let $\mathbf{D}$ be a downwards closed, domain independent¹² dependency with the empty team property. Then $\mathbf{D}$ is strongly first order if and only if it is definable in Constancy Logic $\mathbf{FO}(=(\cdot))$*

In [8], a characterization was found for the strongly first order dependencies that are domain independent¹³ and have the (somewhat technical) property of being *non-jumping*, which in particular is true of strongly first order downwards closed dependencies (even of those without the empty team property):

Theorem 3 ([8], Corollary 31). *Let $\mathbf{D}$ be a non-jumping (or, in particular, downwards closed), domain independent¹⁴, strongly first order dependency. Then $\mathbf{D}$ is definable in $\mathbf{FO}(=(\cdot), \mathcal{D}^\uparrow, \sqcup)$.*

Finally, in [10], a characterization was found for dependencies that are domain-independent¹⁵ and *doubly* strongly first order, in the sense that they and their negation can be added to First Order Logic (separately or jointly) without increasing its expressive power.

In the next section we will find a necessary and sufficient condition for *union-closed*, domain-independent dependencies to be relativizable, thus completing the characterization of “strong first-orderness” with respect to dependencies satisfying the main closure properties discussed in the literature, modulo the condition of domain independence (which, as mentioned, is a natural condition that could be argued to apply to any sensible dependency notion).¹⁶

¹² The result of [7] actually requires a weaker condition than domain independence called *relativizability*. This property is implied by domain independence, and in this work we will focus on domain independent dependencies. Totality **All** is an example of a dependency that is relativizable but not domain independent.

¹³ Again, relativizability suffices for this result.

¹⁴ Or relativizable.

¹⁵ Again, or just relativizable.

¹⁶ Nonetheless, there are dependencies of interest, such as independence atoms $\perp$, which satisfy none of these closure properties; and thus, the problem of fully characterizing strongly first order dependencies remains not entirely solved yet.

3 Characterizing Union-Closed Dependencies

A key insight for our characterization is the observation that if $\mathbf{D}$ is domain-independent and strongly first order then there cannot exist a chain of models satisfying and not satisfying $\mathbf{D}$ as per Fig. 1:

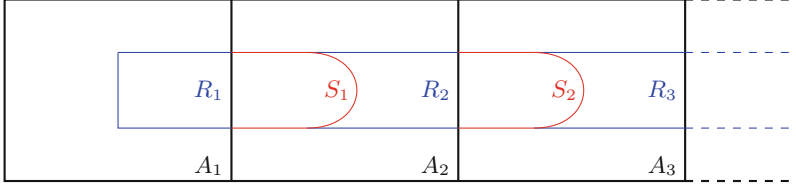


Fig. 1. A forbidden elementary chain. As per Lemma 1, if $\mathbf{D}$ is domain independent and strongly first order, it cannot be the case that, for all $i \in \mathbb{N}$, $(A_i, R_i) \preceq (A_{i+1}, R_{i+1})$, $(A_i, R_i) \in \mathbf{D}$, and $(A_{i+1}, S_i) \notin \mathbf{D}$ for $R_i \subseteq S_i \subseteq R_{i+1}$.

Lemma 1. *Let $\mathbf{D}$ be a k -ary domain-independent dependency and suppose that there exists an elementary chain $(\mathfrak{A}_i)_i \in \mathbb{N}$, where each $\mathfrak{A}_i$ is of the form (A_i, R_i) for some domain A_i and some k -ary relation R_i over A_i , such that*

- $\mathfrak{A}_i = (A_i, R_i) \in \mathbf{D}$ for all $i \in \mathbb{N}$;
- $\mathfrak{A}_i$ is an elementary substructure of $\mathfrak{A}_{i+1}$ ($\mathfrak{A}_i \preceq \mathfrak{A}_{i+1}$) for all $i \in \mathbb{N}$;
- For all $i \in \mathbb{N}$ there exists some k -ary relation S_i over A_{i+1} such that $R_i \subseteq S_i \subseteq R_{i+1}$ and $(A_{i+1}, S_i) \notin \mathbf{D}$.

Then $\mathbf{D}$ is not strongly first order.

Proof. By the Löwenheim-Skolem Theorem, we can assume that all (A_i, R_i) are countable. Then let $\mathfrak{A} = (A, R) = (\bigcup_i A_i, \bigcup_i R_i)$ be the union of the elementary chain. By the Elementary Chain Theorem we have that $\mathfrak{A} \succeq \mathfrak{A}_i$ for all $i \in \mathbb{N}$, and therefore in particular $(A, R) \models \mathbf{D}(R)$; and furthermore since A is the countable union of countable sets it is countable and we can identify it up to isomorphism with the set of natural numbers $\mathbb{N}$.

Then let $\mathfrak{B}$ be a model, with the same domain $\mathbb{N}$, whose signature Σ contains the binary predicate $<$, the unary function $\mathfrak{s}$, and the $(k+1)$ -ary predicate $\hat{S}$, such that $<$ is interpreted as the usual ordering over $\mathbb{N}$, $\mathfrak{s}$ is interpreted as the corresponding successor function $n \mapsto n+1$, and $\hat{S}$ is interpreted as the relation $\{(i, \vec{m}) : \vec{m} \in S_i\}$.

I state that if $\mathbf{D}$ is strongly first order and domain independent then $\mathfrak{B}$ has no uncountable elementary extensions. This is impossible by the Löwenheim-Skolem Theorem; therefore, if I can prove this I can conclude that $\mathbf{D}$ cannot be strongly first order.

If the dependency $\mathbf{D}$ is strongly first order and domain-independent, the complete first order theory of $\mathfrak{B}$ contains the sentence $\forall v \neg \phi(v)$, where $\phi(d)$ is

the first order sentence over the signature $\Sigma \cup \{d\}$ that is equivalent to the **FO**(**D**) sentence

$$\exists t(t < d \wedge \forall \vec{w}(\neg \hat{S}t\vec{w} \vee (\hat{S}t\vec{w} \wedge \mathbf{D}(\vec{w}))))$$

that, as we will later verify, states that there exists some nonempty set of indices $I \subseteq \{1 \dots d\}$ such that $(\mathbb{N}, \bigcup_{i \in I} S_i) \in \mathbf{D}$. Indeed, for any such set we have that $\bigcup_{i \in I} S_i = S_q$ for $q = \max(I)$, and by domain independence $(\mathbb{N}, S_q) \notin \mathbf{D}$ since $(A_{q+1}, S_q) \notin \mathbf{D}$.

Now let $\mathfrak{B}'$ be an uncountable elementary extension of $\mathfrak{B}$ with domain $B' \supsetneq \mathbb{N}$, let $d \in B' \setminus \mathbb{N}$ be a non-standard element of it, and choose $I = \mathbb{N}$. Then $i < d$ for all $i \in \mathbb{N}$, and furthermore since $\bigcup_{i \in \mathbb{N}} S_i = \bigcup_{i \in \mathbb{N}} R_i$ and since as we already observed $(\bigcup_i A_i, \bigcup_i R_i) \in \mathbf{D}$ we have that $(\bigcup_i A_i, \bigcup_i S_i) \in \mathbf{D}$. But then, as $\mathbf{D}$ is domain independent, $(B', \bigcup_i S_i) \in \mathbf{D}$. Therefore $\mathfrak{B}' \models \phi(d)$, which contradicts the claim that $\mathfrak{B}'$ is an elementary extension of $\mathfrak{B}$.

It remains to check that the formula $\phi(d)$ indeed specifies the required property. Suppose that $\mathfrak{M} \models \phi(d)$. Then, by the rules of Team Semantics, there exists some function $H : \{\epsilon\} \rightarrow \mathcal{P}(M) \setminus \{\emptyset\}$ such that $\mathfrak{M} \models_{\{\epsilon\}[H/t]} t < d \wedge \forall \vec{w}(\neg \hat{S}t\vec{w} \vee (\hat{S}t\vec{w} \wedge \mathbf{D}(\vec{w})))$. Now let $X = \{\epsilon\}[H/t]$ and let $I = X(t)$.

Then $I \neq \emptyset$; and since $\mathfrak{M} \models_X t < d$, $i < d$ for all $i \in I$, as required.

For $Y = X[M/\vec{w}]$,¹⁷ we furthermore have that $\mathfrak{M} \models_Y \neg \hat{S}t\vec{w} \vee (\hat{S}t\vec{w} \wedge \mathbf{D}(\vec{w}))$. Therefore, $Y = Y_1 \cup Y_2$, where

- For all $s \in Y_1$, $s(\vec{w}) \notin S_{s(t)}$;
- For all $s \in Y_2$, $s(\vec{w}) \in S_{s(t)}$;
- $(M, Y_2(\vec{w})) \in \mathbf{D}$.

Then necessarily $Y_1 = \{s \in Y : s(\vec{w}) \notin S_{s(t)}\}$ and $Y_2 = \{s \in Y : s(\vec{w}) \in S_{s(t)}\}$; but by construction, in Y the tuple $\vec{w}$ takes all possible values in M^k for all possible values $i \in I$ of t , and therefore $Y_2(\vec{w}) = \{\vec{m} : \exists i \in I \text{ s.t. } \vec{m} \in S_i\} = \bigcup_{i \in I} S_i$. Therefore $(M, \bigcup_{i \in I} S_i) \in \mathbf{D}$ as required.

Conversely, suppose that there exists a nonempty set I of elements less than d such that $(M, \bigcup_{i \in I} S_i) \in \mathbf{D}$. Then let $H : \{\epsilon\} \rightarrow \mathcal{P}(M) \setminus \{\emptyset\}$ be such that $H(\epsilon) = I$, and let $X = \{\epsilon\}[H/t]$ so that $X(t) = I$.

Clearly $\mathfrak{M} \models_X t < d$, and I only need to prove that – for $Y = X[M/\vec{w}]$ – $\mathfrak{M} \models_Y \neg \hat{S}t\vec{w} \vee (\hat{S}t\vec{w} \wedge \mathbf{D}(\vec{w}))$. Now let $Y_1 = \{s \in Y : s(\vec{w}) \notin S_{s(t)}\}$ and $Y_2 = \{s \in Y : s(\vec{w}) \in S_{s(t)}\}$: then $Y = Y_1 \cup Y_2$, $\mathfrak{M} \models_{Y_1} \neg \hat{S}t\vec{w}$ and $\mathfrak{M} \models_{Y_2} \hat{S}t\vec{w}$. Moreover, $Y_2(\vec{w}) = \{s(\vec{w}) \in X[M/\vec{w}] : s(\vec{w}) \in S_{s(t)}\} = \bigcup_{i \in X(t)} S_i = \bigcup_{i \in I} S_i$, and therefore $(M, Y_2(\vec{w})) \in \mathbf{D}$ and $\mathfrak{M} \models_{Y_2} \mathbf{D}\vec{w}$ as required.

In fact, if $\mathbf{D}$ is strongly first order and domain independent even one single “step” (as depicted in Fig. 2) of the chain of Fig. 1 cannot occur:

¹⁷ This is a shorthand for $X[M/\vec{w}_1][M/\vec{w}_2] \dots [M/\vec{w}_k]$.

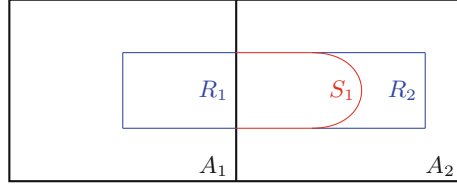


Fig. 2. A forbidden elementary “step”. As per Proposition 3, if $\mathbf{D}$ is domain independent and strongly first order, it cannot be the case that $(A_1, R_1) \preceq (A_2, R_2)$, $(A_1, R_1) \in \mathbf{D}$, $(A_2, R_2) \in \mathbf{D}$, and $(A_2, S_1) \notin \mathbf{D}$ for some S_1 such that $R_1 \subseteq S_1 \subseteq R_2$.

Proposition 3. *Let $\mathbf{D}$ be a domain independent, strongly first order dependency and suppose that $(A_1, R_1) \in \mathbf{D}$ and that $(A_1, R_1) \preceq (A_2, R_2)$. Then for all relations S_1 with $R_1 \subseteq S_1 \subseteq R_2$, $(A_2, S_1) \in \mathbf{D}$ as well.*

Proof. Suppose that this is not the case, i.e. that – as per Fig. 2 – $(A_1, R_1) \preceq (A_2, R_2)$, $R_1 \subseteq S_1 \subseteq R_2$ and $(A_2, S_1) \notin \mathbf{D}$.

Then we can build a chain as per Fig. 1 as follows: start by with the given A_1, R_1, A_2, R_2 , and S_1 . Then consider the theory

$$T := \{\phi(R_3, \vec{a}) : \vec{a} \subseteq A_2, (A_2, R_2) \models \phi(R_2, \vec{a}), \phi \in \mathbf{FO}\} \cup \{S_2 \vec{b} : \vec{b} \in R_2\} \cup \{\forall \vec{x}(S_2 \vec{x} \rightarrow R_3 \vec{x}), \neg \mathbf{D}(S_2)\}.$$

By compactness, this theory is satisfiable. Indeed, any finite subset T_0 of T is entailed by some first order sentence of the form $\phi(R_3, \vec{a}) \wedge \bigwedge_{i=1}^n S_2 \vec{b}_i \wedge \forall \vec{x}(S_2 \vec{x} \rightarrow R_3 \vec{x}) \wedge \neg \mathbf{D}(S_2)$, where $(A_2, R_2) \models \phi(R_2, \vec{a}) \wedge \bigwedge_{i=1}^n R_2 \vec{b}_i$.

But then $(A_2, R_2) \models \exists \vec{z} \vec{w}_1 \dots \vec{w}_n (\phi(R_2, \vec{z}) \wedge \bigwedge_{i=1}^n R_2 \vec{w}_i \wedge \tau(\vec{z} \vec{w}_1 \dots \vec{w}_n))$, where $\tau(\vec{z} \vec{w}_1 \dots \vec{w}_n)$ describes the identity type of $\vec{a} \vec{b}_1 \dots \vec{b}_n$, and so since $(A_1, R_1) \preceq (A_2, R_2)$ there exists a tuple $\vec{c}$ in A_1 and tuples $\vec{d}_1 \dots \vec{d}_n$ in R_1 such that $\vec{c} \vec{d}_1 \dots \vec{d}_n$ has the same identity type as $\vec{a} \vec{b}_1 \dots \vec{b}_n$ and $(A_1, R_1) \models \phi(R_1, \vec{c})$. But then $(A_2, R_2) \models \phi(R_2, \vec{c})$ as well, and moreover $\vec{d}_1 \dots \vec{d}_n \in R_1 \subseteq S_1 \subseteq R_2$, and so T_0 can be satisfied by choosing a model with domain A_2 in which $\vec{a} \vec{b}_1 \dots \vec{b}_n$ are interpreted as $\vec{c} \vec{d}_1 \dots \vec{d}_n$, S_2 is interpreted as S_1 , and R_3 is interpreted as R_2 .

Now, any model of T is an elementary extension (A_3, R_3) of (A_2, R_2) with some S_2 with $R_2 \subseteq S_2 \subseteq R_3$ and $(A_3, S_2) \notin \mathbf{D}$.¹⁸ Iterating this procedure, we can find an infinite chain as per Fig. 1; but by Lemma 1 this is impossible.

Now we can find a “local characterization” of strongly first order, domain independent, union closed dependencies:

Definition 8 (U-Sentences). *Let R be a k -ary relation symbol and let $\vec{a}$ be a tuple of constant symbols. Then a first order sentence over the signature $\{R, \vec{a}\}$ is a U-sentence if and only if it is of the form*

$$\exists \vec{x}(\eta(\vec{x}) \wedge \forall \vec{y}(R\vec{y} \rightarrow \theta(\vec{x}, \vec{y})))$$

¹⁸ Instead $(A_3, R_3) \in \mathbf{D}$, because $(A_3, R_3) \succeq (A_2, R_2) \in \mathbf{D}$ and $\mathbf{D}$ is first order.

where $\vec{x}$ and $\vec{y}$ are disjoint tuples of variables, $\eta(x)$ is a conjunction of first order literals over the signature $\{R, \vec{a}\}$ in which R occurs only positively, and $\theta(\vec{x}, \vec{y})$ is a first order formula over the signature $\{\vec{a}\}$ (i.e. in which R does not appear).

Given two models $\mathfrak{A}$ and $\mathfrak{B}$ with the same signature, we will write $\mathfrak{A} \Rightarrow_U \mathfrak{B}$ if, for every U -sentence ϕ , $\mathfrak{A} \models \phi \Rightarrow \mathfrak{B} \models \phi$.

Proposition 4. *The conjunction of two U -sentences is equivalent to some U -sentence; and if a is a constant symbol occurring on some U -sentence χ and v is a new variable not occurring in χ , $\exists v\chi[v/a]$ is also a U -sentence.¹⁹*

Proof. Let $\chi = \exists \vec{x}(\eta(\vec{x}) \wedge \forall \vec{y}(R\vec{y} \rightarrow \theta(\vec{x}, \vec{y})))$ and $\chi' = \exists \vec{z}(\eta'(\vec{z}) \wedge \forall \vec{w}(R\vec{w} \rightarrow \theta'(\vec{z}, \vec{w})))$ be two U -sentences over the same signature $\{R, \vec{a}\}$. Renaming variables as necessary, we can assume that $\vec{x}$ is disjoint from $\vec{z}$ and that $\vec{y} = \vec{w}$.

Then $\chi \wedge \chi'$ is equivalent to $\exists \vec{x}\vec{z}((\eta(\vec{x}) \wedge \eta'(\vec{z})) \wedge \forall \vec{y}(R\vec{y} \rightarrow (\theta(\vec{x}, \vec{y}) \wedge \theta'(\vec{z}, \vec{y}))))$, which is also a U -sentence.

Moreover, if a be any constant symbol occurring in χ and v is a variable that does not occur in χ , $\exists v(\chi[v/a])$ is the U -sentence

$$\exists v\exists \vec{x}(\eta(\vec{x})[v/a] \wedge \forall \vec{y}(R\vec{y} \rightarrow \theta(\vec{x}, \vec{y})[v/a]))$$

Proposition 5. *Let $\mathbf{D}$ be domain independent, strongly first order and union closed, and suppose that $(A, R) \in \mathbf{D}$.*

Then there exists some U -sentence $\chi(R)$ over the signature $\{R\}$ such that

1. $(A, R) \models \chi(R)$;
2. $\chi(R) \models \mathbf{D}(R)$.

Proof. Suppose that this is not the case. Then R (and, therefore, A) must be infinite, because otherwise there would exist a U -sentence $\exists \vec{x}(\bigwedge_{i=1}^n R\vec{t}_i \wedge \forall \vec{y}(R\vec{y} \rightarrow \bigvee_i \vec{y} = \vec{t}_i))$ that fixes R and thus - by domain independence - entails $\mathbf{D}$.

Note that the theory $\{\chi(S) : \chi \text{ } U\text{-sentence}, (A, R) \models \chi(R)\} \cup \{\neg \mathbf{D}(S)\}$ is finitely satisfiable, since otherwise some finite conjunction of U -sentences that are true of (A, R) would entail $\mathbf{D}(R)$ and by Proposition 4 any such finite conjunction is equivalent to some U -sentence. Therefore, by compactness, there exists a model $(B, S) \notin \mathbf{D}$ such that $(A, R) \Rightarrow_U (B, S)$. We can furthermore assume that (B, S) is $|A|$ -saturated.

Now let $(a_\alpha)_{\alpha < |A|}$ enumerate all elements of A and let us build a mapping $\mathfrak{h} : A \rightarrow B$ such that $(A, R, (a_\alpha)_{\alpha < |A|}) \Rightarrow_U (B, S, (\mathfrak{h}(a_\alpha))_{\alpha < |A|})$.

We can do so via transfinite induction on $\lambda < |A|$, by defining functions $\mathfrak{h}_\lambda$ each of which maps $a_{<\lambda} = (a_\alpha)_{\alpha < \lambda}$ into some $b_{<\lambda} = (b_\alpha)_{\alpha < \lambda}$ such that

- $(A, R, a_{<\lambda}) \Rightarrow_U (B, S, b_{<\lambda})$;
- If $\lambda < \lambda'$ then $\mathfrak{h}_\lambda \subseteq \mathfrak{h}_{\lambda'}$.

Then by taking $\mathfrak{h} = \bigcup_{\lambda < |A|} \mathfrak{h}_\lambda$ we will obtain the required mapping.

¹⁹ Here $\chi[v/a_\lambda]$ represents the formula obtained from χ by replacing the constant a_λ with the variable v .

1. For $\lambda = 0$, we already know that $(A, R) \Rightarrow_U (B, S)$.
2. Suppose that $\lambda + 1$ is a successor ordinal and that, for all $\gamma < \lambda + 1$, mappings $\mathfrak{h}_\gamma$ exist with the required properties. Then in particular $\mathfrak{h}_\lambda$ exists and maps $\vec{a}_{<\lambda}$ into some $\vec{b}_{<\lambda}$ such that $(A, R, \vec{a}_{<\lambda}) \Rightarrow_U (B, S, \vec{b}_{<\lambda})$. Now consider the element a_λ and the theory

$$\Theta(R, \vec{a}_{<\lambda}, a_\lambda) = \{\chi : \chi \text{ is a } U\text{-sentence, } (A, R, \vec{a}_{<\lambda}, a_\lambda) \models \chi\}$$

Now, if Θ_0 is any finite subset of Θ , by Proposition 4 the expression $\xi_0 := \exists v \bigwedge_{\chi \in \Theta_0} \chi[v/a_\lambda]$ is equivalent to some U -sentence over $\{R, \vec{a}_{<\lambda}\}$ and

$(A, R, \vec{a}_{<\lambda}) \models \xi_0$; and therefore, by induction hypothesis, $(B, S, \vec{b}_{<\lambda}) \models \xi_0$ and there exists some $b' \in B$ such that $(B, S, \vec{b}_{<\lambda}, b') \models \chi$ for all $\chi \in \Theta_0$. Since (B, S) is $|A|$ -saturated and $\lambda < |A|$, this means that there exists some $b_\lambda \in B$ such that $(B, S, \vec{b}_{<\lambda}, b_\lambda) \models \Theta$. But then $(A, R, \vec{a}_{<\lambda+1}) \Rightarrow_U (B, S, \vec{b}_{<\lambda+1})$, and we can define $\mathfrak{h}_{\lambda+1} = \mathfrak{h} \cup \{(a_\lambda, b_\lambda)\}$.

3. Suppose that λ is a limit ordinal and that, for all $\gamma < \lambda$, mappings $\mathfrak{h}_\gamma$ exist with the required properties. Then take $\mathfrak{h}_\lambda = \bigcup_{\gamma < \lambda} \mathfrak{h}_\gamma$. The mapping $\mathfrak{h}$ thus built is in particular an embedding of (A, R) into (B, S) . Indeed, it is injective, because expressions of the form $a_i \neq a_j$ are U -sentences; and furthermore, for all k -tuples $\vec{c}$ of elements in A , $\vec{c} \in R \Leftrightarrow (\mathfrak{h}(c_i))_{i=1\dots k} \in S$ since the sentences $R\vec{c}$ and $\forall \vec{y}(R\vec{y} \rightarrow \vec{y} \neq \vec{c})$ (which is equivalent to $\neg R\vec{c}$) are also U -sentences.

Now, let (A_1, R_1) be the image of (A, R) over $\mathfrak{h}$. Then (A_1, R_1) is isomorphic to (A, R) , and so in particular $(A_1, R_1) \in \mathbf{D}$, and furthermore $R_1 = S \cap (A_1)^k \subseteq S$. Note additionally that, for every U -sentence ϕ with constants $\vec{a}$ in A_1 , $(A_1, R_1, \vec{a}) \models \phi \Leftrightarrow (A, R, \mathfrak{h}^{-1}(\vec{a})) \models \phi \Rightarrow (B, S, \vec{a}) \models \phi$ and so every U -sentence with parameters in A_1 that holds of (A_1, R_1) also holds in (B, S) .

Now consider any tuple $\vec{b} \in S \setminus R_1$. I state that there exists some $(A_2, R_2) \succeq (A_1, R_1)$ such that $\vec{b} \in R_2$, as per Fig. 3. Indeed, consider the elementary diagram $\{\phi(R_2, \vec{a}) : \vec{a} \subseteq A_1, \phi \in \mathbf{FO}, (A_1, R_1, \vec{a}) \models \phi(R_1, \vec{a})\}$ of (A_1, R_1) together with the sentence $R_2 \vec{b}$. If this theory was not satisfiable, by compactness there would exist some tuple $\vec{a}$ of elements of A_1 and some sentence $\phi(R_1, \vec{a})$ such that $(A_1, R_1, \vec{a}) \models \phi(R_1, \vec{a})$ and $\phi(R, \vec{a}) \models \neg R \vec{b}$. But then, if $\vec{t}$ is the tuple of terms obtained from $\vec{b}$ by replacing every constant symbol not in $\vec{a}$ with some new variable, we have that $\phi(R, \vec{a}) \models \forall \vec{z}(\neg R \vec{t})$, where $\vec{z}$ lists all newly introduced variables,²⁰ and so $(A_1, R_1, \vec{a}) \models \forall \vec{z}(\neg R_1 \vec{t})$ and therefore $(A_1, R_1, \vec{a}) \models \forall \vec{y}(R_1 \vec{y} \rightarrow \forall \vec{z}(\vec{y} \neq \vec{t}))$. But since all U -sentences with parameters in A_1 that are true of (A_1, R_1) are also true of (B, S) we would then have in particular that $(B, S, \vec{a}) \models \forall \vec{y}(S \vec{y} \rightarrow \forall \vec{z}(\vec{y} \neq \vec{t}))$, which is impossible because $\vec{b} \in S$. Therefore the theory is satisfiable, and any model (A_2, R_2) of it is an elementary extension of (A_1, R_1) that contains $\vec{b}$, as required.

²⁰ This follows from the so-called “Lemma on Constants” (e.g. Lemma 2.3.2. of [17]): if $T \models \phi(\vec{c})$, where $\vec{c}$ is a tuple of constants not occurring in T , then $T \models \forall \vec{x} \phi(\vec{x})$.

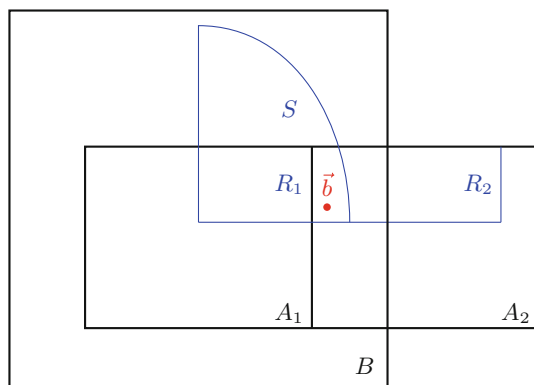


Fig. 3. (A_1, R_1) is embedded in (B, S) , and $(A_1, R_1) \in \mathbf{D}$ while $(B, S) \notin \mathbf{D}$, and $\vec{b} \in S \setminus R_1$. Then we can build some $(A_2, R_2) \succeq (A_1, R_1)$ such that $\vec{b} \in R_2$; and then, via Proposition 3, we can show that $(A_2, R_1 \cup \{\vec{b}\}) \in \mathbf{D}$. But then by domain independence $(B, R_1 \cup \{\vec{b}\}) \in \mathbf{D}$ for any such $\vec{b}$.

So we have that $(A_1, R_1) \preceq (A_2, R_2)$ and that $\vec{b} \in R_2 \setminus R_1$. Then $(A_2, R_2) \in \mathbf{D}$, because $(A_1, R_1) \in \mathbf{D}$ and (A_2, R_2) is elementarily equivalent to it. Therefore, $(A_2, R_1 \cup \{\vec{b}\}) \in \mathbf{D}$: indeed, otherwise we would have that $(A_1, R_1) \in \mathbf{D}$, $(A_2, R_1 \cup \{\vec{b}\}) \notin \mathbf{D}$, $R_1 \cup \{\vec{b}\} \subseteq R_2$ and $(A_2, R_2) \succeq (A_1, R_1)$, which (taking $S_1 = R_1 \cup \{\vec{b}\}$) is the configuration of Fig. 2 that cannot occur by Proposition 3. Then, by domain independence, $(B, R_1 \cup \{\vec{b}\}) \in \mathbf{D}$ as well.

But if this is the case for all $\vec{b} \in S \setminus R_1$ and $\mathbf{D}$ is union closed,

$(B, \bigcup_{\vec{b} \in S \setminus R_1} R_1 \cup \{\vec{b}\}) = (B, S) \in \mathbf{D}$ which contradicts our assumption that $(B, S) \notin \mathbf{D}$.

We can now find a global characterization of domain independent, union-closed, strongly first order dependencies via a simple application of compactness:

Theorem 4. *Let $\mathbf{D}$ be a domain independent, union-closed, strongly first order dependency. Then $\mathbf{D}(R)$ is equivalent to some finite disjunction of U -sentences over the signature $\{R\}$.*

Proof. For each countable $(A, R) \in \mathbf{D}$, let $\chi_{A,R}(R)$ be a U-sentence over the signature $\{R\}$ such that $(A, R) \models \chi_{A,R}(R)$ and $\chi_{A,R}(R) \models \mathbf{D}(R)$.

Then the theory $\{\neg\chi_{A,R}(R) : (A, R) \in \mathbf{D}, A \text{ countable}\} \cup \{\mathbf{D}(R)\}$ has no countable models, and therefore it has no models at all; and so, by compactness, there exists a finite number of U-sentences $\chi_1 \dots \chi_n$ such that $\mathbf{D}(R) \models \bigvee_{i=1}^n \chi_i(R)$. But every $\chi_i(R)$ entails $\mathbf{D}(R)$, and so $\mathbf{D}(R)$ is equivalent to $\bigvee_{i=1}^n \chi_i(R)$.

Corollary 1. *Let $\mathbf{D}$ be a domain independent, union-closed dependency. Then $\mathbf{D}$ is strongly first order if and only if it is definable in terms of constancy atoms, nonemptiness atoms and global disjunctions.*

Proof. Suppose that $\mathbf{D}$ is strongly first order. Then by Theorem 4, $\mathbf{D}(R)$ is equivalent to some $\bigvee_{i=1}^n \chi_i(R)$ where each $\chi_i(R)$ is a U -sentence over the signature $\{R\}$. Now let $\vec{w}$ be a k -tuple of distinct variables not occurring in any χ_i ; and for each $\chi_i = \exists \vec{x}(\eta(\vec{x}) \wedge \forall \vec{y}(R\vec{y} \rightarrow \theta(\vec{x}, \vec{y})))$, let

$$\chi'_i(\vec{w}) := \exists \vec{x}(\eta(\vec{x}) \wedge \eta'(\vec{x}, \vec{w}) \wedge \theta(\vec{x}, \vec{w}))$$

where $\eta'(\vec{x}, \vec{w})$ is obtained from $\eta(\vec{x})$ by replacing each conjunct of the form $R\vec{t}$ with $(\top \vee (\text{NE}(\vec{w}) \wedge \vec{t} = \vec{w}))$.²¹

I state that, for all models $\mathfrak{M}$ and teams X over it with domain containing $\vec{w}$, $\mathfrak{M} \models_X \mathbf{D}\vec{w}$ if and only if $\mathfrak{M} \models_X \bigsqcup_{i=1}^n \chi'_i(\vec{w})$. If I can prove this then I am done: indeed, by Proposition 2 and by the fact that $\text{NE} \subseteq \mathcal{D}^\dagger$ every dependency that is definable in $\mathbf{FO}(=\cdot, \text{NE}, \sqcup)$ is strongly first order.

It suffices to prove that, for all models $\mathfrak{M}$,²² and all teams X over $\mathfrak{M}$ whose domain contains $\vec{w}$, $\mathfrak{M} \models_X \chi'_i(\vec{w})$ if and only if $(M, X(\vec{w})) \models \chi_i(R)$.

Suppose that $\mathfrak{M} \models_X \chi'_i(\vec{w})$: then there exists a tuple $\vec{a}$ of elements such that, for $Y = X[\vec{a}/\vec{x}]$,²³ $\mathfrak{M} \models_Y \eta'(\vec{x}, \vec{w}) \wedge \theta(\vec{x}, \vec{w})$.

Then $(M, X(\vec{w})) \models \eta(\vec{a}) \wedge \forall \vec{y}(R\vec{y} \rightarrow \theta(\vec{a}, \vec{y}))$. Indeed,

1. If some literal of the form $x_j = x_{j'}$ [$x_j \neq x_{j'}$] occurs among the conjuncts of η , it occurs unchanged also in η' , and since $\mathfrak{M} \models_Y \eta$ we have that $a_j = a_{j'}$ [$a_j \neq a_{j'}$];
2. If some literal of the form $R\vec{t}$ occurs among the conjuncts of η , where $\vec{t}$ is a tuple of variables in $\vec{x}$, the expression $\top \vee (\text{NE}(\vec{w}) \wedge \vec{t} = \vec{w})$ occurs among the conjuncts of η' . Therefore, $Y = Y_1 \cup Y_2$, where $Y_2 \neq \emptyset$ and $s(\vec{t}) = s(\vec{w})$ for all $s \in Y_2$. But $s(\vec{t}) = t[\vec{a}/\vec{x}]$ for all $s \in Y$. Therefore, $t[\vec{a}/\vec{x}] \in Y_2(\vec{w}) \subseteq X(\vec{w})$.
3. Since $\mathfrak{M} \models_{X[\vec{a}/\vec{x}]} \theta(\vec{x}, \vec{w})$ and θ is first order, by Proposition 1 for all $s \in X$ we have that $\mathfrak{M} \models_s \theta(\vec{a}, \vec{w})$ in the usual Tarskian sense. This means that, for all values $\vec{m}$ that $\vec{w}$ takes for some assignment in X , $\mathfrak{M} \models \theta(\vec{a}, \vec{m})$; and therefore $(M, X(\vec{w})) \models \forall \vec{y}(R\vec{y} \rightarrow \theta(\vec{a}, \vec{y}))$.

Therefore $(M, X(\vec{w})) \models \eta(\vec{a}) \wedge \forall \vec{y}(R\vec{y} \rightarrow \theta(\vec{a}, \vec{y}))$ and hence $(M, X(\vec{w})) \models \exists \vec{x}(\eta(\vec{x}) \wedge \forall \vec{y}(R\vec{y} \rightarrow \theta(\vec{x}, \vec{y})))$ as required.

Conversely, suppose that $(M, X(\vec{w})) \models \chi_i(R)$. Then there exists a tuple $\vec{a}$ of elements of M such that $(M, X(\vec{w})) \models \eta(\vec{a}) \wedge \forall \vec{y}(R\vec{y} \rightarrow \theta(\vec{a}, \vec{y}))$. Now let $Y = X[\vec{a}/\vec{x}]$: clearly, $\mathfrak{M} \models_Y =(\vec{x})$. Moreover, $\mathfrak{M} \models_Y \eta'(\vec{x}, \vec{w}) \wedge \theta(\vec{x}, \vec{w})$: indeed,

1. If some literal of the form $x_j = x_{j'}$ [$x_j \neq x_{j'}$] occurs among the conjuncts of η' , the same conjunct also occurs in η . Therefore $a_j = a_{j'}$ [$a_j \neq a_{j'}$] and $\mathfrak{M} \models_Y x_j = x_{j'}$ [$\mathfrak{M} \models_Y x_j \neq x_{j'}$].

²¹ Here $\top$ can be any literal that is always true, such as $w = w$ for some $w \in \vec{w}$.

²² Note that the signature of $\mathfrak{M}$ does not necessarily have to include the symbol R , which is instead interpreted as $X(\vec{w})$ when evaluating $\chi_i(R)$; and accordingly, $\chi'_i(\vec{w})$ is a $\mathbf{FO}(=\cdot, \text{NE})$ formula over the *empty* signature.

²³ More formally: $Y = X[H_1/x_1] \dots [H_k/x_k]$, where each H_j picks the same singleton $\{a_j\}$ for all assignments in its domain – indeed, only choice functions of this type can ensure that $\mathfrak{M} \models_Y =(\vec{x})$.

2. If some expression of the form $\top \vee (\text{NE}(\vec{w}) \wedge \vec{t} = \vec{w})$ is among the conjuncts of η' , $R\vec{t}$ is among the conjuncts of η . Therefore, $\vec{t}[\vec{a}/\vec{x}] \in X(\vec{w})$, and so there exists some $s \in Y = X[\vec{a}/\vec{x}]$ such that $s(\vec{w}) = \vec{t}[\vec{a}/\vec{x}] = s(\vec{t})$. But then $Y = Y \cup \{s\}$, $\mathfrak{M} \models_Y \top$, and $\mathfrak{M} \models_{\{s\}} \text{NE}(\vec{w}) \wedge \vec{t} = \vec{w}$; and therefore $\mathfrak{M} \models_Y \top \vee (\text{NE}(\vec{w}) \wedge \vec{t} = \vec{w})$, as required.
3. Since $(M, X(\vec{w})) \models \forall \vec{y} (R\vec{y} \rightarrow \theta(\vec{a}, \vec{m}))$, for all $\vec{m} \in X(\vec{w})$ we have that $\mathfrak{M} \models \theta(\vec{a}, \vec{m})$. Therefore, for all $s \in X$, $\mathfrak{M} \models_{s[\vec{a}/\vec{x}]} \theta(\vec{x}, \vec{w})$ in the usual Tarskian sense; and since $Y = X[\vec{a}/\vec{x}]$, by Proposition 1 we have that $\mathfrak{M} \models_Y \theta(\vec{x}, \vec{w})$.

Therefore, $\mathfrak{M} \models_{X[\vec{a}/\vec{x}]} \eta'(\vec{x}, \vec{w}) \wedge \theta(\vec{x}, \vec{w})$ and $\mathfrak{M} \models_X \chi'_i(\vec{w})$.

Thus, for every $\chi_i(R)$ we can find a $\mathbf{FO}(=\cdot, \text{NE})$ formula $\chi'_i(\vec{w})$ such that $\mathfrak{M} \models_X \chi'_i(\vec{w})$ if and only if $(M, X(\vec{w})) \models \chi_i(R)$. But $\mathbf{D}(R)$ is equivalent to $\bigvee_{i=1}^t \chi_i(R)$. Therefore $(M, X(\vec{w})) \models \mathbf{D}(R)$ if and only if there exists some $i \in 1 \dots t$ such that $\mathfrak{M} \models_X \chi'_i(\vec{w})$, that is, if and only if $\mathfrak{M} \models_X \bigsqcup_{i=1}^t \chi'_i(\vec{w})$; and thus $\mathbf{D}$ is definable in $\mathbf{FO}(=\cdot, \text{NE}, \sqcup)$.

Corollary 2. *Let $\mathcal{D}$ be a family of domain-independent dependencies each of which is union closed, upwards closed, or downwards closed²⁴. Then $\mathcal{D}$ is strongly first order if and only if all $\mathbf{D} \in \mathcal{D}$ are individually strongly first order.*

Proof. The left to right direction is obvious.

For the right to left direction, suppose that all $\mathbf{D} \in \mathcal{D}$ are individually strongly first order. By Theorem 3, every downwards closed (or non-jumping) $\mathbf{D} \in \mathcal{D}$ is definable in $\mathbf{FO}(=\cdot, \mathcal{D}^\uparrow, \sqcup)$, and by Corollary 1 every union closed $\mathbf{D} \in \mathcal{D}$ is definable in $\mathbf{FO}(=\cdot, \text{NE}, \sqcup)$, and all upwards closed first order dependencies (including all NE dependencies) are in $\mathcal{D}^\uparrow$. Therefore, every sentence of $\mathbf{FO}(\mathcal{D})$ is equivalent to some sentence of $\mathbf{FO}(=\cdot, \mathcal{D}^\uparrow, \sqcup)$; but by Proposition 2 every such sentence is equivalent to some first order sentence, and therefore $\mathcal{D}$ is strongly first order.

4 Conclusion

In this work we found a full characterization of strongly first order dependencies for the case of dependencies that are also domain-independent and union-closed. This essentially completes the characterization of strongly first order dependencies for all the main classes of dependencies studied in the literature.

A general characterization, even modulo domain independence, of strongly first order dependencies is however still missing; and it is the hope of the author that the ideas and techniques developed in this work may be of some use for proving such a characterization. Additionally, it would be interesting to explore the same question with respect to richer languages than First Order Logic, as it was done in [10] for the case of First Order Logic (with Team Semantics) plus a restricted form of contradictory negation. For example, which dependencies can be added to Inclusion Logic without increasing its expressive power? This is not

²⁴ Or non-jumping.

fully known at the moment, although it is known that no first order union-closed dependency increases the expressive power of Inclusion Logic [11].

Finally, there exist quantitative variants of Team Semantics such as *Probabilistic Team Semantics* [2] and *Multiteam Semantics* [14]. The same question about which dependencies increase the expressive power of First Order Logic and which ones do not can be also asked for these interesting generalizations of the basic framework of Team Semantics, and its answer would also in their case provide a good starting point for the classification of the corresponding extensions of First Order Logic.

References

1. Armstrong, W.W.: Dependency structures of data base relationships. In: Proceedings of IFIP World Computer Congress, pp. 580–583 (1974)
2. Durand, A., Hannula, M., Kontinen, J., Meier, A., Vinter, J.: Probabilistic team semantics. In: Ferrarotti, F., Woltran, S. (eds.) FoIKS 2018. LNCS, vol. 10833, pp. 186–206. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-90050-6_11
3. Galliani, P.: The dynamics of imperfect information. Ph.D. thesis, University of Amsterdam, September 2012. <https://dare.uva.nl/record/425951>
4. Galliani, P.: Inclusion and exclusion dependencies in team semantics: on some logics of imperfect information. *Ann. Pure Appl. Logic* **163**(1), 68–84 (2012). <https://doi.org/10.1016/j.apal.2011.08.005>
5. Galliani, P.: Upwards closed dependencies in team semantics. *Inf. Comput.* **245**, 124–135 (2015). <https://doi.org/10.1016/j.ic.2015.06.008>
6. Galliani, P.: On strongly first-order dependencies. In: Abramsky, S., Kontinen, J., Väänänen, J., Vollmer, H. (eds.) *Dependence Logic*, pp. 53–71. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-31803-5_4
7. Galliani, P.: Characterizing downwards closed, strongly first-order, relativizable dependencies. *J. Symbol. Logic* **84**(3), 1136–1167 (2019). <https://doi.org/10.1017/jsl.2019.12>
8. Galliani, P.: Characterizing strongly first order dependencies: the non-jumping relativizable case. *Electron. Proc. Theor. Comput. Sci.* **305**, 66–82 (2019). <https://doi.org/10.4204/eptcs.305.5>
9. Galliani, P.: Safe dependency atoms and possibility operators in team semantics. *Inf. Comput.* **278**, 104593 (2020)
10. Galliani, P.: Doubly strongly first order dependencies. In: Silva, A., Wassermann, R., de Queiroz, R. (eds.) *WoLLIC 2021*. LNCS, vol. 13038, pp. 47–63. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-88853-4_4
11. Galliani, P., Hella, L.: Inclusion logic and fixed point logic. In: Rocca, S.R.D. (ed.) *Computer Science Logic 2013 (CSL 2013)*. Leibniz International Proceedings in Informatics (LIPIcs), vol. 23, pp. 281–295. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2013). <https://doi.org/10.4230/LIPIcs.CSL.2013.281>
12. Galliani, P., Väänänen, J.: On dependence logic (2013). [arXiv:1305.5948](https://arxiv.org/abs/1305.5948)
13. Grädel, E., Väänänen, J.: Dependence and independence. *Stud. Logica* **101**(2), 399–410 (2013). <https://doi.org/10.1007/s11225-013-9479-2>
14. Grädel, E., Wilke, R.: Logics with multiteam semantics. *ACM Trans. Comput. Logic (TOCL)* **23**(2), 1–30 (2022)

15. Henkin, L.: Some remarks on infinitely long formulas. In: *Infinitistic Methods, Proceedings of the Symposium on Foundations of Mathematics*, pp. 167–183. Pergamon Press (1961)
16. Hintikka, J., Sandu, G.: Informational independence as a semantic phenomenon. In: Fenstad, J., Frolov, I., Hilpinen, R. (eds.) *Logic, Methodology and Philosophy of Science*, pp. 571–589. Elsevier (1989). [https://doi.org/10.1016/S0049-237X\(08\)70066-1](https://doi.org/10.1016/S0049-237X(08)70066-1)
17. Hodges, W.: *A Shorter Model Theory*. Cambridge University Press, Cambridge (1997)
18. Hodges, W.: Compositional semantics for a language of imperfect information. *J. Interest Group Pure Appl. Logics* **5**(4), 539–563 (1997). <https://doi.org/10.1093/jigpal/5.4.539>
19. Kontinen, J., Kuusisto, A., Virtema, J.: Decidability of predicate logics with team semantics. In: *41st International Symposium on Mathematical Foundations of Computer Science (MFCS 2016)*. *Leibniz International Proceedings in Informatics (LIPIcs)*, vol. 58, pp. 60:1–60:14 (2016). <https://doi.org/10.4230/LIPIcs.MFCS.2016.60>
20. Kontinen, J., Yang, F.: Logics for first-order team properties. In: Iemhoff, R., Moortgat, M., de Queiroz, R. (eds.) *WoLLIC 2019*. *LNCS*, vol. 11541, pp. 392–414. Springer, Heidelberg (2019). https://doi.org/10.1007/978-3-662-59533-6_24
21. Kuusisto, A.: A double team semantics for generalized quantifiers. *J. Logic Lang. Inform.* **24**(2), 149–191 (2015)
22. Rönholm, R.: Arity fragments of logics with team semantics (2018)
23. Väänänen, J.: An atom’s worth of anonymity (to appear)
24. Väänänen, J.: *Dependence Logic*. Cambridge University Press, Cambridge (2007). <https://doi.org/10.1017/CBO9780511611193>



Towards an Intuitionistic Deontic Logic Tolerating Conflicting Obligations

Tiziano Dalmonte¹(✉) , Charles Grellois² , and Nicola Olivetti²

¹ Free University of Bozen-Bolzano, Bolzano, Italy
tiziano.dalmonte@unibz.it

² Aix Marseille Univ, Université de Toulon, CNRS, LIS, Marseille, France

Abstract. We propose a minimal deontic logic, called MIND, based on intuitionistic logic. This logic gives a very simple solution to handling conflicting obligations: the presence of two conflicting obligations does not entail the triviality of the set of norms. Moreover, the logic supports the claim that there may be no obligations at all, so that a logical truth is not obligatory. Like in intuitionistic/constructive modal logic, in this logic, the two deontic modalities Obligation and Permission are not dual, however as a difference with so-called constructive modal logic, it supports distribution of permission over disjunction, but it does not satisfy aggregation of obligations. The logic MIND is a non-normal modal logic based on intuitionistic logic and it is semantically characterised by a suitable neighbourhood semantics. We further present a simple cut-free sequent calculus for this logic. By means of this calculus we show that logic MIND is decidable and that it satisfies the disjunction property.

Keywords: Deontic logic · Intuitionistic modal logic · Non-normal modal logic · Neighbourhood semantics · Sequent calculus

1 Introduction

Deontic logic (DL) has been studied for a long time since the seminal work by Von Wright [22]. The standard system of DL is normal modal logic KD. In this logic the two modal operators $\mathcal{O}$ and $\mathcal{P}$, an alternative notation to $\Box$ and $\Diamond$, have a deontic meaning: respectively “it is obligatory” and “it is permitted”. The two operators are interdefinable, and the axiom (D) $\mathcal{O}A \supset \mathcal{P}A$ expresses the fundamental assumption that something is obligatory only if it is permitted.

Nonetheless, the logic KD has been considered controversial for several reasons. One of the earliest objections concerns the monotonicity principle (M)

$$\frac{A \supset B}{\mathcal{O}A \supset \mathcal{O}B}$$

This work was originated by the internship of Pauline Bonnet in spring 2021 funded by ANR-FWF TICAMORE project. This work was partially funded by the project ComprAS at the Free University of Bozen-Bolzano.

which is the source of well-known paradoxes, such as the Letter paradox by Ross [18] and the Gentle Murder paradox by Forrester [7]. However, another important and controversial aspect, widely discussed in the literature [8], is that KD does not tolerate conflicting obligations, in the sense that if a set of norms contains two incompatible obligations, it collapses becoming trivial. This is an important aspect since conflicting obligations are ubiquitous, a trivial example: you should keep your mobile switched on, say $\mathcal{O}A$ (as you are waiting for an urgent call), at the same time you should keep your mobile switched off, $\mathcal{O}\neg A$ (as you are attending a concert). We believe that a reasonable system of deontic logic should tolerate conflicting obligations.

Problems with Monotonicity. One of the earliest objections to KD is Ross' paradox [18] caused by the monotonicity principle M validated by KD, from which we get in particular $\mathcal{O}A \supset \mathcal{O}(A \vee B)$, that in turn justifies paradoxical conclusions: take for instance A = “you send the letter” and B = “you destroy the letter”. Moreover, it is easily seen that principle M is essentially involved in the notorious *Gentle Murder* paradox by Forrester [7]. Another controversial consequence of M is so-called *deontic explosion*:

$$(\text{exp}) \quad \mathcal{O}\perp \supset \mathcal{O}B$$

Thus, if the legal code contains an inconsistent obligation, e.g. $\mathcal{O}\perp$, then it becomes trivial justifying any obligation. One possible way-out is to give up the principle M , which leads to Non-Normal modal logics.

Conflicting Obligations and Axiom D . It is immediately seen that the presence of axiom D makes it impossible to tolerate conflicting obligations: since $\mathcal{P}A \equiv \neg\mathcal{O}\neg A$, by D we get $\mathcal{O}A \supset \neg\mathcal{O}\neg A$, that is

$$\neg(\mathcal{O}A \wedge \mathcal{O}\neg A)$$

which expresses that directly conflicting obligations cannot coexist. Observe that we get this conclusion even without monotonicity M .

Although the axiom D seems an essential feature of deontic logic, the literature has considered systems of non-normal modal logics in which D is replaced by the weaker principle $(P) \neg\mathcal{O}\perp$ (notice however that the two principles are equivalent in normal modal logic).

Conflicting Obligations and Aggregation. The principle of Aggregation $(C_{\mathcal{O}})$ for $\mathcal{O}$ is:

$$(\mathcal{O}A \wedge \mathcal{O}B) \supset \mathcal{O}(A \wedge B)$$

This principle is problematic when dealing with conflicting obligations, that is whenever obligations A and B cannot be jointly fulfilled, as we have:

$$(\mathcal{O}A \wedge \mathcal{O}\neg A) \supset \mathcal{O}(A \wedge \neg A)$$

or equivalently $(\mathcal{O}A \wedge \mathcal{O}\neg A) \supset \mathcal{O}\perp$. Thus, two conflicting obligations produce an inconsistent obligation. As a consequence, even if we accept only the weak principle P , we get again

$$\neg(\mathcal{O}A \wedge \mathcal{O}\neg A)$$

One possible way-out is to get rid of aggregation. Non-normal subsystems of KD, such as logic MD (minimal deontic logic) rejecting agglomeration have been considered in the literature.

However, notice that aggregation $C_{\mathcal{O}}$ is equivalent by duality to

$$(C_{\mathcal{P}}) \quad \mathcal{P}(A \vee B) \supset \mathcal{P}A \vee \mathcal{P}B$$

In contrast, the principle $C_{\mathcal{P}}$ is hardly questionable in itself¹, and there are no reasons to reject it.

In conclusion, even on the base of *classical* non-normal modal logic, with dual deontic operators and the uncontroversial $\neg\mathcal{P}\perp$, there is no system with all the following axiom/properties:

1. $\neg\mathcal{O}\perp$ or (D) $\mathcal{O}A \supset \mathcal{P}A$
2. $(C_{\mathcal{P}}) \mathcal{P}(A \vee B) \supset \mathcal{P}A \vee \mathcal{P}B$
3. $\not\vdash \neg(\mathcal{O}A \wedge \mathcal{O}\neg A)$

We consider a further requirement: following a remark going back to von Wright [22] (see also [9, 23]), we consider that there might be no obligations at all, and that a logical truth cannot be an object of a deontic statement, so that neither $\mathcal{O}\top$, nor $\mathcal{P}\top$ are taken as valid:

4. $\not\vdash \mathcal{O}\top$ and $\not\vdash \mathcal{P}\top$

We can also consider a stronger requirement, similarly to an assumption of Ability Logic [6], that the negation of the above formulas are actually explicitly assumed (cf. e.g. [13]), which gives

- 4'. $\vdash \neg\mathcal{O}\top$ and $\vdash \neg\mathcal{P}\top$

The goal of this paper is to show that it is possible to define a logical system satisfying the above properties if we take as a base intuitionistic logic rather than classical logic. However, the aim of grounding deontic logic on intuitionistic logic can also be independently motivated: we believe that it makes sense to consider a constructive meaning of the deontic modalities. It may be significant to distinguish “ A is obligatory” as it can be established on the base of a legal code from “it is not the case that $\neg A$ is permitted”, likewise to distinguish “ A is permitted” from “it is not the case that $\neg A$ is obligatory”, this latter can be read as “ A is forbidden”. In this regard, some authors distinguish between a *weak* and a *strong* notion of permission: the former is just the negation of the

¹ To the extent that the literature generally takes it *tacitly* for granted and even consider the stronger principle of Free-Choice $\mathcal{P}(A \vee B) \supset \mathcal{P}A \wedge \mathcal{P}B$ (see e.g. [11]), that we do not assume.

opposite obligation, whereas the latter must be explicitly supported by a legal code [10, 23]. Moreover, observe that in the classical setting (modal logic KD) we have that $\mathcal{P}A \vee \mathcal{P}B$ may be valid, for specific A and B even if we are unable to prove either $\mathcal{P}A$ or $\mathcal{P}B$, in particular we have that

$$\mathcal{P}A \vee \mathcal{P}\neg A \text{ is valid for any sentence } A.$$

But as we remarked, there is a stronger sense of permission for which we may not be willing to accept $\mathcal{P}A \vee \mathcal{P}\neg A$, unless we are able to establish which of the two holds. For instance, let A be the assertion “being 16 years old I get Covid vaccine booster”, according to Country regulation, by the way subject to change over time, it might be $\mathcal{P}A$ or $\mathcal{P}\neg A$, or even both, so that if I ask “Am I allowed to take the vaccine booster?”, I would be hardly satisfied to get as an answer $\mathcal{P}A \vee \mathcal{P}\neg A$, because of logical validity, without knowing which one of the two holds. On the basis of these considerations, we believe that it is meaningful to define a deontic logic based on intuitionistic logic. We note that to the best of our knowledge, deontic logics with an intuitionistic basis have not been considered in the literature, with the possible exceptions of [1, 15, 17] but only for special purposes and contexts, and not as a general topic.

We propose a logic called MIND (Minimal Intuitionistic Deontic Logic) which is intended to be a minimal system of deontic logic based on intuitionistic logic. This logic satisfies the properties mentioned above (notably axiom D), it does not satisfy monotonicity, and tolerates conflicting obligations. We consider also a stronger version MIND^+ of MIND: while MIND satisfies the requirement 4, MIND^+ satisfies the stronger requirement 4', as a consequence it cannot be a subsystem of any monotonic (intuitionistic) modal logic. The logic MIND is an extension of intuitionistic logic with non-normal modalities in the sense of [4]. However, it *does not* belong to the family logics studied in [4] and it is not a subsystem of any of them, even more: since it satisfies $\mathcal{C}_\mathcal{P}$ (but not $\mathcal{C}_\mathcal{O}$), it is not a subsystem of constructive modal logic CK [2], nor of Wijesekera’s system CCDL [21]. We present first the logic MIND axiomatically, we then provide a matching neighbourhood semantics. Furthermore, we present a simple cut-free sequent calculus for it. Notice in passim that the fact that the simple structure of sequent calculi suffices is a difference with intuitionistic modal logic satisfying the axiom $C_\Diamond$ such as Simpson’s IK [19], which requires complex structured calculi. By means of this calculus we obtain that the logic MIND is decidable and that it satisfies the disjunction property, so that $\mathcal{P}A \vee \mathcal{P}\neg A$ is not valid. The same results apply to the extension MIND^+ . We believe that this logic is a good starting point for developing deontic logics on a constructive base tolerating conflicting obligations.

2 The Logics MIND and MIND^+

The language $\mathcal{L}$ of MIND is based on a set $Atm = \{p_1, p_2, p_3, \dots\}$ of countably many propositional variables, it extends the language of intuitionistic propositional logic with the modalities $\mathcal{O}$ and $\mathcal{P}$. The formulas of $\mathcal{L}$ are defined by the

following grammar, where p is any element of Atm :

$$A := p \mid \perp \mid A \wedge A \mid A \vee A \mid A \supset A \mid \mathcal{O}A \mid \mathcal{P}A.$$

We consider as usual some defined connectives: $\neg A := A \supset \perp$ and $\top := \neg \perp$.

Definition 1. *The logic MIND is axiomatised by the following:*

- Any axiomatisation of intuitionistic propositional logic (IPL) formulated in the language $\mathcal{L}$
- (D) $\mathcal{O}A \supset \mathcal{P}A$
- ($P_{\mathcal{P}}$) $\neg \mathcal{P}\perp$
- ($C_{\mathcal{P}}$) $\mathcal{P}(A \vee B) \supset (\mathcal{P}A \vee \mathcal{P}B)$
- Congruence rules

$$(CG_{\mathcal{P}}) \frac{A \supset B \quad B \supset A}{\mathcal{P}A \supset \mathcal{P}B} \quad (CG_{\mathcal{O}}) \frac{A \supset B \quad B \supset A}{\mathcal{O}A \supset \mathcal{O}B}$$

We also consider a stronger system, called $MIND^+$, defined extending MIND with the following axiom:

- ($Q_{\mathcal{P}}$) $\neg \mathcal{P}\top$

Provability $\vdash_{MIND}$ and $\vdash_{MIND^+}$ in MIND and $MIND^+$ is defined as usual (the index will be often omitted by readability). We extend provability to sets of formulas $X \subseteq \mathcal{L}$, by defining $X \vdash A$ if there exist $B_1, \dots, B_n \in X$ with $n \geq 0$ such that $\vdash (B_1 \wedge \dots \wedge B_n) \supset A$. Moreover we use $MIND^{(+)}$ to denote any of the two systems.

The semantics of the logics MIND and $MIND^+$ is defined in terms of Intuitionistic Neighbourhood models, which are equipped by two independent neighbourhood functions for $\mathcal{O}$ and $\mathcal{P}$.

Definition 2 (Semantics). *A model for MIND is a tuple $M = (W, \leq, N_{\mathcal{O}}, N_{\mathcal{P}}, V)$, where W is a non-empty set, whose elements are called worlds, $\leq$ is a preorder on W (i.e., a reflexive and transitive relation), $V : W \longrightarrow 2^{Atm}$ is a hereditary valuation function, that is*

- $x \leq y$ implies $V(x) \subseteq V(y)$,

and $N_{\mathcal{O}}, N_{\mathcal{P}}$ are two neighbourhood functions $W \longrightarrow 2^{2^W}$ satisfying the following conditions:

- $x \leq y$ implies $N_{\mathcal{O}}(x) \subseteq N_{\mathcal{O}}(y)$,
- $x \leq y$ implies $N_{\mathcal{P}}(x) \subseteq N_{\mathcal{P}}(y)$,
- $N_{\mathcal{O}}(x) \subseteq N_{\mathcal{P}}(y)$,
- $\emptyset \notin N_{\mathcal{P}}(x)$,
- for $\alpha, \beta \subseteq W$, if $\alpha \cup \beta \in N_{\mathcal{P}}(x)$ then $\alpha \in N_{\mathcal{P}}(x)$ or $\beta \in N_{\mathcal{P}}(x)$.

Moreover, a model for $MIND^+$ is any model for MIND that also satisfies the following condition:

- $W \notin N_{\mathcal{P}}(x)$.

Truth conditions are defined as follows:

- $M, x \models p$ iff $p \in V(x)$.
- $M, x \not\models \perp$.
- $M, x \models A \wedge B$ iff $M, x \models A$ and $M, x \models B$.
- $M, x \models A \vee B$ iff $M, x \models A$ or $M, x \models B$.
- $M, x \models A \supset B$ iff for all $y \in W$, if $x \leq y$ and $M, y \models A$, then $M, y \models B$.
- $M, x \models \mathcal{O}A$ iff $[A]^M \in N_{\mathcal{O}}(x)$.
- $M, x \models \mathcal{P}A$ iff $[A]^M \in N_{\mathcal{P}}(x)$.

where $[A]^M = \{x \mid M, x \models A\}$. When the intended model M is clear, we simply write $x \models A$ and $[A]$ instead of $M, x \models A$ and $[A]^M$. A formula A is valid in M if $[A]^M = W$; finally a formula is valid if it is valid in every model M .

The type of models considered here is the same as in [4], but the model conditions are different².

It is easy to see that the Hereditary Property holds in neighbourhood models.

Proposition 1. *Given a model $M = (W, \leq, N_{\mathcal{O}}, N_{\mathcal{P}}, V)$, for any $x, y \in W$, and any formula A : if $x \leq y$ and $M, x \models A$, then also $M, y \models A$.*

Theorem 1 (Soundness and Completeness). *A formula is valid in all models for MIND (respectively, for MIND⁺) if and only if it is derivable in MIND (respectively, in MIND⁺).*

Proof. Soundness is left to the reader.

To prove completeness we provide a canonical model construction M_{can} . We make use of the usual notion of *prime set*, meaning a set $X \subseteq \mathcal{L}$ satisfying the following conditions (i) $X \not\models \perp$, (ii) if $\vdash A$ then $A \in X$, (iii) if $A \vee B \in X$ then $A \in X$ or $B \in X$. We make use of the following claims whose proof is standard:

- *Claim 1.* For any set $X \subseteq \mathcal{L}$ and formulas B, C , if $X \cup \{B\} \not\models C$, then there exists Y prime such that $X \cup \{B\} \subseteq Y$ and $C \notin Y$.
- *Claim 2.* For any formula A , $\vdash A$ if and only if $A \in X$, for all prime sets X .

Given a formula A we define:

$$\widehat{A} = \{X \mid X \text{ is prime and } A \in X\}.$$

We now define the canonical model $M_{can} = (W, \leq, N_{\mathcal{O}}, N_{\mathcal{P}}, V)$ as follows:

- $W = \{X \subseteq \mathcal{L} \mid X \text{ is prime}\}$
- $X \leq Y$ iff $X \subseteq Y$

² A Neighbourhood semantics for intuitionistic modal logic has been defined by Kojima [14], both the type of models and the truth conditions are different from the models considered here; in particular, in Kojima's models, the neighbourhood function is *antimonotonic* with respect to the order.

- $V(X) = \{p \mid p \in X\}$
- $N_{\mathcal{O}}(X) = \{\widehat{B} \mid \mathcal{O}B \in X\}$
- $N_{\mathcal{P}}(X) = \{\widehat{B} \mid \mathcal{P}B \in X\} \cup \{S \subseteq W \mid \text{for all } C \in \mathcal{L}, S \neq \widehat{C}\}^3.$

We first prove the following claims:

- *Claim 3.* $\widehat{A} \in N_{\mathcal{O}}(X)$ if and only if $\mathcal{O}A \in X$.
- *Claim 4.* $\widehat{A} \in N_{\mathcal{P}}(X)$ if and only if $\mathcal{P}A \in X$.

If $\mathcal{O}A \in X$ (respectively $\mathcal{P}A \in X$), then by definition $\widehat{A} \in N_{\mathcal{O}}(X)$ (respectively $\widehat{A} \in N_{\mathcal{P}}(X)$). If $\widehat{A} \in N_{\mathcal{O}}(X)$, then by definition there is $\mathcal{O}B \in X$ such that $\widehat{B} = \widehat{A}$. This means that $\vdash A \supset B$ and $\vdash B \supset A$, then by $CG_{\mathcal{O}}$, $\vdash \mathcal{O}B \supset \mathcal{O}A$, therefore $\mathcal{O}A \in X$. Similarly, if $\widehat{A} \in N_{\mathcal{P}}(X)$, then there is $\mathcal{P}B \in X$ such that $\widehat{B} = \widehat{A}$. Then $\mathcal{P}A \in X$ follows using $CG_{\mathcal{P}}$. Note that in the proof for $\mathcal{P}B$ only the first half of the definition of $N_{\mathcal{P}}(X)$ comes into play, since the second set is composed of sets of formulas which do not coincide with any $\widehat{C}$.

Now we can prove that M_{can} has the required properties.

- $X \leq Y$ implies $V(X) \subseteq V(Y)$: This is obvious since $X \leq Y$ means $X \subseteq Y$.
- $X \leq Y$ implies $N_{\mathcal{O}}(X) \subseteq N_{\mathcal{O}}(Y)$: Obvious for the same reason.
- $X \leq Y$ implies $N_{\mathcal{P}}(X) \subseteq N_{\mathcal{P}}(Y)$: Obvious by $X \subseteq Y$ and definition of $N_{\mathcal{P}}$.
- $N_{\mathcal{O}}(X) \subseteq N_{\mathcal{P}}(X)$: Let $\alpha \in N_{\mathcal{O}}(X)$. Then for some formula C , $\alpha = \widehat{C}$ and $\mathcal{O}C \in X$, thus since $\vdash \mathcal{O}C \supset \mathcal{P}C$, we have $\mathcal{P}C \in X$, whence $\alpha \in N_{\mathcal{P}}(X)$.
- $\emptyset \notin N_{\mathcal{P}}(x)$: Observe that $\emptyset = \widehat{\perp}$. Thus if $\emptyset \in N_{\mathcal{P}}(x)$, then $\widehat{\perp} \in N_{\mathcal{P}}(X)$, this means that $\mathcal{P}\perp \in X$ (as shown above), and since $\neg\mathcal{P}\perp \in X$ we have a contradiction.
- $\alpha \cup \beta \in N_{\mathcal{P}}(X)$ implies $\alpha \in N_{\mathcal{P}}(X)$ or $\beta \in N_{\mathcal{P}}(X)$. We prove the converse: suppose that for some $\alpha, \beta \subseteq W$ we have $\alpha \notin N_{\mathcal{P}}(X)$ and $\beta \notin N_{\mathcal{P}}(X)$, we show that $\alpha \cup \beta \notin N_{\mathcal{P}}(X)$. Since $\alpha \notin N_{\mathcal{P}}(X)$ then there is a formula $C \in \mathcal{L}$ such that $\alpha = \widehat{C}$ but $\mathcal{P}C \notin X$; similarly since $\beta \notin N_{\mathcal{P}}(X)$, there is a formula $D \in \mathcal{L}$ such that $\beta = \widehat{D}$ but $\mathcal{P}D \notin X$. By the properties of prime sets it follows that $\mathcal{P}C \vee \mathcal{P}D \notin X$. Since $\vdash \mathcal{P}(C \vee D) \supset (\mathcal{P}C \vee \mathcal{P}D)$, we get $\mathcal{P}(C \vee D) \notin X$, whence $\widehat{C \vee D} \notin N_{\mathcal{P}}(X)$. But observe that, by the properties of prime sets it holds $\widehat{C \vee D} = \widehat{C} \cup \widehat{D}$. Thus $\alpha \cup \beta = \widehat{C} \cup \widehat{D} = \widehat{C \vee D}$, then we can conclude $\alpha \cup \beta \notin N_{\mathcal{P}}(X)$.

The canonical model for MIND^+ also satisfies the following:

- $W \notin N_{\mathcal{P}}(x)$: Observe that $W = \widehat{\top}$, thus if $W \in N_{\mathcal{P}}(X)$, then $\mathcal{P}\top \in X$ against the axiom $\neg\mathcal{P}\top$.

³ The second half of the definition of $N_{\mathcal{P}}$ is needed in order to enforce the last model condition of Definition 2 and it is inspired by *Non-Proof Sets* used by Chellas [3] in his canonical model construction for some classical non-normal modal logics.

We now prove the truth Lemma: for any formula A , $\hat{A} = [A]$ or more explicitly, for any $X \in W$:

$$A \in X \text{ iff } X \models A.$$

We prove the two directions by mutual induction on the complexity of A . The cases $A = p, \perp, B \wedge C, B \vee C$ are standard, here we show the cases $A = B \supset C, \mathcal{O}B$ (the case $A = \mathcal{P}B$ is analogous to $A = \mathcal{O}B$):

- Let $A = B \supset C$. Suppose that $B \supset C \notin X$, then $X \not\models B \supset C$, thus $X \cup \{B\} \not\models C$. By Claim 1, there is Y prime such that $X \cup \{B\} \subseteq Y$ and $C \notin Y$. By definition, $Y \in W$ and $X \leq Y$, moreover by inductive hypothesis $Y \models B$ and $Y \not\models C$, therefore $X \not\models B \supset C$. For the opposite direction, suppose that $B \supset C \in X$, $X \leq Y$, and $Y \models B$. Then $B \supset C \in Y$ (because $X \subseteq Y$), moreover by inductive hypothesis $B \in Y$. It follows $C \in Y$, that by inductive hypothesis implies $Y \models C$. Therefore $X \models B \supset C$.
- Let $A = \mathcal{O}B$. By inductive hypothesis we have $\hat{B} = [B]$. Thus $\mathcal{O}B \in X$ iff $\hat{B} \in N_{\mathcal{O}}(X)$ iff $[B] \in N_{\mathcal{O}}(X)$ iff $X \models \mathcal{O}B$.

Completeness of $\text{MIND}^{(+)}$ follows from these results in the usual way: suppose that $\not\models A$. Then there is a prime set X such that $A \notin X$. By definition, X belongs to the canonical model for $\text{MIND}^{(+)}$, that was shown to be a model for $\text{MIND}^{(+)}$, moreover $X \not\models A$. Therefore A is not valid in the class of all models for $\text{MIND}^{(+)}$.

3 Sequent Calculi

In this section, we define sequent calculi $\mathcal{S}_{\text{MIND}}$ and $\mathcal{S}_{\text{MIND}^+}$ for the logics MIND and MIND^+ . We show that the calculi are equivalent with the axiomatic systems, and that they enjoy the admissibility of the standard structural rules and cut. Then, basing on the calculi, we prove that MIND and MIND^+ satisfy the disjunction property. Moreover, we discuss how the calculi can be used to establish a decision procedure for derivability in the logics.

As usual, *sequents* have the form $\Gamma \Rightarrow \Delta$, where Γ and Δ are finite, possibly empty multisets of formulas of $\mathcal{L}$. A sequent $\Gamma \Rightarrow \Delta$ is interpreted as a formula of $\mathcal{L}$ as $\bigwedge \Gamma \supset \bigvee \Delta$ if Γ is non-empty, and it is interpreted as $\bigvee \Delta$ if Γ is empty, where $\bigvee \emptyset$ is interpreted as $\perp$.

The calculi $\mathcal{S}_{\text{MIND}}$ and $\mathcal{S}_{\text{MIND}^+}$ are defined as modal extensions of a calculus for intuitionistic propositional logic IPL. We adopt as propositional base the multi-succedent sequent calculus m-G3i from [20, Ch. 3] (Fig. 1). Then $\mathcal{S}_{\text{MIND}}$ and $\mathcal{S}_{\text{MIND}^+}$ are defined extending m-G3i with the following modal rules from Fig. 1:

$$\mathcal{S}_{\text{MIND}} := \text{m-G3i} \cup \{\text{CG}_{\mathcal{O}}, \text{D}, \text{C}_{\mathcal{P}}, \text{P}_{\mathcal{O}}, \text{P}_{\mathcal{P}}\}, \quad \mathcal{S}_{\text{MIND}^+} := \mathcal{S}_{\text{MIND}} \cup \{\text{Q}_{\mathcal{O}}, \text{Q}_{\mathcal{P}}\}.$$

While both single-succedent and multi-succedent calculi are available for IPL, the multi-succedent formulation is required for $\text{MIND}^{(+)}$ in order to express a rule for $\text{C}_{\mathcal{P}}$. Examples of derivations of the axioms of $\text{MIND}^{(+)}$ in $\mathcal{S}_{\text{MIND}^{(+)}}$ are displayed in Fig. 2.

$$\begin{array}{c}
\text{init} \frac{}{\Gamma, p \Rightarrow p, \Delta} \quad \perp_L \frac{}{\Gamma, \perp \Rightarrow \Delta} \quad \wedge_L \frac{\Gamma, A, B \Rightarrow \Delta}{\Gamma, A \wedge B \Rightarrow \Delta} \quad \vee_R \frac{\Gamma \Rightarrow A, B, \Delta}{\Gamma \Rightarrow A \vee B, \Delta} \\
\wedge_R \frac{\Gamma \Rightarrow A, \Delta \quad \Gamma \Rightarrow B, \Delta}{\Gamma \Rightarrow A \wedge B, \Delta} \quad \vee_L \frac{\Gamma, A \Rightarrow \Delta \quad \Gamma, B \Rightarrow \Delta}{\Gamma, A \vee B \Rightarrow \Delta} \quad \supset_R \frac{\Gamma, A \Rightarrow B}{\Gamma \Rightarrow A \supset B, \Delta} \\
\supset_L \frac{\Gamma, A \supset B \Rightarrow A, \Delta \quad \Gamma, B \Rightarrow \Delta}{\Gamma, A \supset B \Rightarrow \Delta} \quad P_O \frac{A \Rightarrow}{\Gamma, \mathcal{O}A \Rightarrow \Delta} \quad P_P \frac{A \Rightarrow}{\Gamma, \mathcal{P}A \Rightarrow \Delta} \\
CG_O \frac{A \Rightarrow B \quad B \Rightarrow A}{\Gamma, \mathcal{O}A \Rightarrow \mathcal{O}B, \Delta} \quad C_P \frac{A \Rightarrow B_1, \dots, B_n \quad B_1 \Rightarrow A \quad \dots \quad B_n \Rightarrow A}{\Gamma, \mathcal{P}A \Rightarrow \mathcal{P}B_1, \dots, \mathcal{P}B_n, \Delta} \\
D \frac{A \Rightarrow B_1, \dots, B_n \quad B_1 \Rightarrow A \quad \dots \quad B_n \Rightarrow A}{\Gamma, \mathcal{O}A \Rightarrow \mathcal{P}B_1, \dots, \mathcal{P}B_n, \Delta} \quad Q_O \frac{\Rightarrow A}{\Gamma, \mathcal{O}A \Rightarrow \Delta} \quad Q_P \frac{\Rightarrow A}{\Gamma, \mathcal{P}A \Rightarrow \Delta}
\end{array}$$

Fig. 1. Rules of sequent calculi.

$$\begin{array}{c}
\text{init} \frac{}{p \Rightarrow p, q} \quad \text{init} \frac{}{q \Rightarrow p, q} \quad \text{init} \frac{}{p \Rightarrow p, q} \quad \text{init} \frac{}{q \Rightarrow p, q} \quad \text{init} \frac{}{q \Rightarrow p, q} \\
\vee_L \frac{}{p \vee q \Rightarrow p, q} \quad \vee_R \frac{}{p \Rightarrow p \vee q} \quad \vee_R \frac{}{q \Rightarrow p \vee q} \quad C_P \frac{}{p \vee q \Rightarrow \mathcal{P}p, \mathcal{P}q} \\
\frac{\mathcal{P}(p \vee q) \Rightarrow \mathcal{P}p, \mathcal{P}q}{\mathcal{P}(p \vee q) \Rightarrow \mathcal{P}p \vee \mathcal{P}q} \vee_R \\
\frac{}{\Rightarrow \mathcal{P}(p \vee q) \supset \mathcal{P}p \vee \mathcal{P}q} \supset_R \\
\\
\frac{}{p \Rightarrow p} \text{init} \quad \frac{}{\mathcal{O}p \Rightarrow \mathcal{P}p} D \quad \frac{}{\mathcal{O}p \Rightarrow \mathcal{P}p} \supset_R \quad \frac{}{\perp \Rightarrow} \perp_L \quad \frac{}{\perp \Rightarrow} P_P \quad \frac{}{\perp \Rightarrow} \supset_R \\
\frac{}{\mathcal{O}p \Rightarrow \mathcal{P}p} \supset_R \quad \frac{}{\mathcal{P}\perp \Rightarrow \perp} \supset_R \quad \frac{}{\mathcal{P}(\perp \supset \perp) \Rightarrow \perp} Q_P \\
\frac{}{\Rightarrow \mathcal{O}p \supset \mathcal{P}p} \supset_R \quad \frac{}{\Rightarrow \mathcal{P}\perp \supset \perp} \supset_R \quad \frac{}{\Rightarrow \mathcal{P}(\perp \supset \perp) \supset \perp} \supset_R
\end{array}$$

Fig. 2. Derivations of the axioms of MIND and MIND⁺ in the sequent calculi $\mathcal{S}_{\text{MIND}}$ and $\mathcal{S}_{\text{MIND}^+}$.

We are now going to prove that the calculi $\mathcal{S}_{\text{MIND}^{(+)}}$ are sound and complete with respect to the corresponding logic $\text{MIND}^{(+)}$. Both soundness and completeness are proved in a purely syntactical way. The proof of soundness is as follows.

Theorem 2 (Soundness). *If $\Gamma \Rightarrow \Delta$ is derivable in $\mathcal{S}_{\text{MIND}}$ (respectively in $\mathcal{S}_{\text{MIND}^+}$), then $\bigwedge \Gamma \supset \bigvee \Delta$ is derivable in MIND (respectively in MIND⁺).*

Proof. From left to right, we show that for each rule of $\mathcal{S}_{\text{MIND}^{(+)}}$ with premisses $\Gamma_1 \Rightarrow \Delta_1, \dots, \Gamma_n \Rightarrow \Delta_n$, and conclusion $\Gamma' \Rightarrow \Delta'$, the corresponding Hilbert-style rule with premisses $\bigwedge \Gamma_1 \supset \bigvee \Delta_1, \dots, \bigwedge \Gamma_n \supset \bigvee \Delta_n$ and conclusion $\bigwedge \Gamma' \supset \bigvee \Delta'$ is derivable in MIND⁽⁺⁾.

(CG_O) Suppose that $\vdash A \supset B$ and $\vdash B \supset A$. Then by CG_O , $\vdash \mathcal{O}A \supset \mathcal{O}B$, therefore $\vdash \bigwedge \Gamma \wedge \mathcal{O}A \supset \mathcal{O}B \vee \bigvee \Delta$.

(C_P) Suppose that $\vdash A \supset B_1 \vee \dots \vee B_k$ and $\vdash B_i \supset A$ for each $i \leq k$. Then $\vdash B_1 \vee \dots \vee B_k \supset A$, thus by CG_P , $\vdash \mathcal{P}A \supset \mathcal{P}(B_1 \vee \dots \vee B_k)$. Moreover by C_P , $\vdash \mathcal{P}(B_1 \vee \dots \vee B_k) \supset \mathcal{P}B_1 \vee \dots \vee \mathcal{P}B_k$, therefore $\vdash \bigwedge \Gamma \wedge \mathcal{P}A \supset \mathcal{P}B_1 \vee \dots \vee \mathcal{P}B_k \vee \bigvee \Delta$.

(D) Suppose that $\vdash A \supset B_1 \vee \dots \vee B_k$ and $\vdash B_i \supset A$ for each $i \leq k$. Then $\vdash B_1 \vee \dots \vee B_k \supset A$, thus by CG_O , $\vdash \mathcal{O}A \supset \mathcal{O}(B_1 \vee \dots \vee B_k)$. Moreover by

$D, \vdash \mathcal{O}(B_1 \vee \dots \vee B_k) \supset \mathcal{P}(B_1 \vee \dots \vee B_k)$, and by $C_P, \vdash \mathcal{P}(B_1 \vee \dots \vee B_k) \supset \mathcal{P}B_1 \vee \dots \vee \mathcal{P}B_k$, therefore $\vdash \bigwedge \Gamma \wedge \mathcal{O}A \supset \mathcal{P}B_1 \vee \dots \vee \mathcal{P}B_k \vee \bigvee \Delta$.

($P_{\mathcal{O}}$) Suppose that $\vdash A \supset \perp$. Since $\vdash \perp \supset A$, by $CG_{\mathcal{O}}, \vdash \mathcal{O}A \supset \mathcal{O}\perp$. However from D and P_P , it follows $\vdash \neg \mathcal{O}\perp$, thus $\vdash \neg \mathcal{O}A$. (P_P) is analogous.

($Q_{\mathcal{O}}$) Suppose that $\vdash A$. Then $\vdash \top \supset A$, moreover $\vdash A \supset \top$, then by $CG_{\mathcal{O}}, \vdash \mathcal{O}A \supset \mathcal{O}\top$. However from D and Q_P , it follows $\vdash \neg \mathcal{O}\top$, thus $\vdash \neg \mathcal{O}A$. (Q_P) is analogous.

As usual, the syntactic proof of completeness is a bit more complicated as it requires some auxiliary results about the properties of the calculi $\mathcal{S}_{\text{MIND}(+)}$. We first prove that the following structural rules of weakening and contraction are height-preserving admissible in the calculi:

$$\text{wk}_L \frac{\Gamma \Rightarrow \Delta}{\Gamma, A \Rightarrow \Delta} \quad \text{wk}_R \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow A, \Delta} \quad \text{ctr}_L \frac{\Gamma, A, A \Rightarrow \Delta}{\Gamma, A \Rightarrow \Delta} \quad \text{ctr}_R \frac{\Gamma \Rightarrow A, A, \Delta}{\Gamma \Rightarrow A, \Delta}$$

In turn, the admissibility of these rules is needed in order to prove that the following cut rule is also admissible (where A is called the cut formula):

$$\text{cut} \frac{\Gamma \Rightarrow A, \Delta \quad \Gamma', A \Rightarrow \Delta'}{\Gamma, \Gamma' \Rightarrow \Delta, \Delta'}$$

To make this precise, we recall that a rule R is *admissible* in $\mathcal{S}_{\text{MIND}(+)}$ if whenever the premisses of R are derivable, the conclusion of R is also derivable, and that a single-premiss rule R is *height-preserving admissible* if whenever the premiss of R is derivable, the conclusion of R is derivable with a derivation of at most the same height. We start considering the following proposition.

Proposition 2 (Admissibility of structural rules). *The rules wk_L , wk_R , ctr_L , and ctr_R above are height-preserving admissible in $\mathcal{S}_{\text{MIND}(+)}$.*

Proof. By mutual induction on the height of the derivation of the premiss of wk_L , wk_R , ctr_L , or ctr_R , taking into account the last rule applied in the derivation. The proof extends the one of [20] for m-G3i with the analysis of the modal rules. For wk_L and wk_R the proof is straightforward, here we consider as an example the proof for ctr_R , distinguishing among the following cases.

(i) Let R be the last rule applied in the derivation of $\Gamma \Rightarrow A, A, \Delta$, and suppose that none of the occurrences of A is principal in the application of R . Then both occurrences of A are introduced by R as side context. It is easy to see that with a different application of R one obtains $\Gamma \Rightarrow A, \Delta$.

(ii) Let R be the last rule applied in the derivation of $\Gamma \Rightarrow A, A, \Delta$, and suppose that only one occurrence of A is principal in the application of R . Then the second occurrence of A are introduced by R as side context. It is easy to see that with a different application of R one obtains $\Gamma \Rightarrow A, \Delta$.

(iii) Let R be the last rule applied in the derivation of $\Gamma \Rightarrow A, A, \Delta$, and suppose that both occurrences of A are principal in the application of R . In this case one can apply contraction to the premisses of R , which is admissible by inductive hypothesis. Consider for instance the following derivation:

$$\text{D} \frac{A \Rightarrow B_1, \dots, B_j, B_j, \dots, B_n \quad B_1 \Rightarrow A \quad \dots \quad B_j \Rightarrow A \quad \dots \quad B_n \Rightarrow A}{\Gamma, \mathcal{O}A \Rightarrow \mathcal{P}B_1, \dots, \mathcal{P}B_j, \mathcal{P}B_j, \dots, \mathcal{P}B_n, \Delta}$$

The derivation is converted as follows, with a height-preserving admissible application of ctr_R , thus obtaining a derivation of $\Gamma, \mathcal{O}A \Rightarrow \mathcal{P}B_1, \dots, \mathcal{P}B_j, \dots, \mathcal{P}B_n, \Delta$ with the same height:

$$\text{ctr}_R \frac{A \Rightarrow B_1, \dots, B_j, B_j, \dots, B_n}{A \Rightarrow B_1, \dots, B_j, \dots, B_n} \quad \text{D} \frac{B_1 \Rightarrow A \quad \dots \quad B_j \Rightarrow A \quad \dots \quad B_n \Rightarrow A}{\Gamma, \mathcal{O}A \Rightarrow \mathcal{P}B_1, \dots, \mathcal{P}B_j, \dots, \mathcal{P}B_n, \Delta}$$

Given the above proposition we can prove the following result.

Theorem 3 (Cut admissibility). *The rule cut is admissible in $\mathcal{S}_{\text{MIND}+}$.*

Proof. By induction on lexicographically ordered pairs (c, h) , where c is the complexity of the cut formula (i.e., the number of binary connectives or modalities occurring in it), and $h = h_1 + h_2$, called cut height, is the sum of the heights of the derivations of the premisses of cut. As usual, we distinguish some cases according to whether the cut formula is or not principal in the last rules applied in the derivation of the premisses of cut. For the cases where the last rules applied in the derivation of the premisses of cut are propositional we refer to [20, Ch. 4]. Here we only show a few most relevant cases involving modal rules, the other cases are similar.

(i) The cut formula is not principal in the derivation of the left premiss of cut. We consider the following example.

$$\text{D} \frac{B \Rightarrow C_1, \dots, C_n \quad C_1 \Rightarrow B \quad \dots \quad C_n \Rightarrow B}{\Gamma, \mathcal{O}B \Rightarrow \mathcal{P}C_1, \dots, \mathcal{P}C_n, A, \Delta} \quad \frac{\Gamma', A \Rightarrow \Delta'}{\Gamma, \Gamma', \mathcal{O}B \Rightarrow \mathcal{P}C_1, \dots, \mathcal{P}C_n, \Delta, \Delta'} \text{ cut}$$

The derivation can be transformed considering the following alternative application of D:

$$\frac{B \Rightarrow C_1, \dots, C_n \quad C_1 \Rightarrow B \quad \dots \quad C_n \Rightarrow B}{\Gamma, \Gamma', \mathcal{O}B \Rightarrow \mathcal{P}C_1, \dots, \mathcal{P}C_n, \Delta, \Delta'} \text{D}$$

(ii) The cut formula is not principal in the derivation of the left premiss of cut.

$$\frac{\Gamma \Rightarrow A, \Delta \quad \frac{B \Rightarrow C_1, \dots, C_n \quad C_1 \Rightarrow B \quad \dots \quad C_n \Rightarrow B}{\Gamma', A, \mathcal{O}B \Rightarrow \mathcal{P}C_1, \dots, \mathcal{P}C_n, \Delta'} \text{D}}{\Gamma, \Gamma', \mathcal{O}B \Rightarrow \mathcal{P}C_1, \dots, \mathcal{P}C_n, \Delta, \Delta'} \text{ cut}$$

The derivation can be transformed considering the following alternative application of D:

$$\frac{B \Rightarrow C_1, \dots, C_n \quad C_1 \Rightarrow B \quad \dots \quad C_n \Rightarrow B}{\Gamma, \Gamma', \mathcal{O}B \Rightarrow \mathcal{P}C_1, \dots, \mathcal{P}C_n, \Delta, \Delta'} \text{D}$$

(iii) The cut formula is principal in the derivations of both premisses of cut. We consider as an example the following application of cut:

$$\frac{\Gamma, \mathcal{O}A \Rightarrow \mathcal{P}B_1, \dots, \mathcal{P}B_n, \mathcal{P}C, \Delta \quad \Gamma', \mathcal{P}C \Rightarrow \mathcal{P}D_1, \dots, \mathcal{P}D_m, \Delta'}{\Gamma, \Gamma', \mathcal{O}A \Rightarrow \mathcal{P}B_1, \dots, \mathcal{P}B_n, \mathcal{P}D_1, \dots, \mathcal{P}D_m, \Delta, \Delta'} \text{ cut}$$

where the two premisses of **cut** are respectively derived as follows:

$$\frac{A \Rightarrow B_1, \dots, B_n, C \quad \{B_i \Rightarrow A\}_{i \leq n} \quad C \Rightarrow A}{\Gamma, \mathcal{O}A \Rightarrow \mathcal{P}B_1, \dots, \mathcal{P}B_n, \mathcal{P}C, \Delta} \text{D}$$

$$\frac{C \Rightarrow D_1, \dots, D_m \quad \{D_j \Rightarrow C\}_{j \leq m}}{\Gamma', \mathcal{P}C \Rightarrow \mathcal{P}D_1, \dots, \mathcal{P}D_m, \Delta'} \text{C}_{\mathcal{P}}$$

The derivation is converted as follows, with several application of **cut** on a cut formula of lower complexity, whence admissible by inductive hypothesis: first we derive

$$\frac{A \Rightarrow B_1, \dots, B_n, C \quad C \Rightarrow D_1, \dots, D_m}{A \Rightarrow B_1, \dots, B_n, D_1, \dots, D_m} \text{cut}$$

Then for each $j \leq m$ we derive

$$\frac{D_j \Rightarrow C \quad C \Rightarrow A}{D_j \Rightarrow A} \text{cut}$$

Finally we obtain

$$\frac{A \Rightarrow B_1, \dots, B_n, D_1, \dots, D_m \quad \{B_i \Rightarrow A\}_{i \leq n} \quad \{D_j \Rightarrow A\}_{j \leq m}}{\Gamma, \Gamma', \mathcal{O}A \Rightarrow \mathcal{P}B_1, \dots, \mathcal{P}B_n, \mathcal{P}D_1, \dots, \mathcal{P}D_m, \Delta, \Delta'} \text{D}$$

Basing on the structural properties of the calculi we can now prove that $\mathcal{S}_{\text{MIND}}$ and $\mathcal{S}_{\text{MIND}^+}$ are complete with respect to **MIND** and **MIND**⁺.

Theorem 4 (Completeness). *If $\bigwedge \Gamma \supset \bigvee \Delta$ is derivable in **MIND** (respectively in **MIND**⁺), then $\Gamma \Rightarrow \Delta$ is derivable in $\mathcal{S}_{\text{MIND}}$ (respectively in $\mathcal{S}_{\text{MIND}^+}$).*

Proof. We have shown in Fig. 2 that the modal axioms of **MIND**⁽⁺⁾ are derivable in $\mathcal{S}_{\text{MIND}^{(+)}}$ (as usual, the derivations can be generalised to arbitrary formulas A and B), moreover the derivations of the intuitionistic axioms are standard. Concerning the rules of **MIND**⁽⁺⁾, Modus Ponens is simulated by **cut**, which has been proved admissible, in the usual way. It remains to show that $CG_{\mathcal{O}}$ and $CG_{\mathcal{P}}$ are admissible: Assume that $\Rightarrow A \supset B$ and $\Rightarrow B \supset A$ are derivable. Considering the derivable sequents $A, A \supset B \Rightarrow B$ and $B, B \supset A \Rightarrow A$, by **cut** we obtain $A \Rightarrow B$ and $B \Rightarrow A$, then by $C_{\mathcal{P}}$ we derive $\mathcal{P}A \Rightarrow \mathcal{P}B$, finally by $\supset_R$ we obtain $\Rightarrow \mathcal{P}A \supset \mathcal{P}B$. The proof for $CG_{\mathcal{O}}$ is analogous using the rule $CG_{\mathcal{O}}$.

We now show that, basing on the calculi $\mathcal{S}_{\text{MIND}^{(+)}}$, one can prove two fundamental properties of the logics **MIND**⁽⁺⁾. First, we can show that the calculi $\mathcal{S}_{\text{MIND}^{(+)}}$ provide a decision procedure for the logics. This is the case whenever the calculi support terminating proof search, which in turn requires that every branch of every proof in the calculi is finite. To this purpose, observe that the premisses of each rule of $\mathcal{S}_{\text{MIND}^{(+)}}$ (including also the modal rules) have a strictly smaller complexity than the conclusion, with the exception of the rule $\supset_L$ where the principal implication $A \supset B$ is copied into the left premiss. It follows that a loop-checking mechanism analogous to the one in [20] for m-G3i is sufficient to ensure that proof search in $\mathcal{S}_{\text{MIND}^{(+)}}$ terminates. As a consequence we obtain:

Theorem 5. *The logics **MIND** and **MIND**⁺ are decidable.*

Concerning the complexity of the logics $\text{MIND}^{(+)}$, we strongly conjecture that the logics are PSPACE (as intuitionistic logic), a property that we aim to check in detail in future work.

Finally, we show that the logics are acceptable from an intuitionistic, or constructive, point of view as they satisfy the fundamental disjunction property. This property can be obtained as a consequence of the following lemma.

Lemma 1. *If $\Rightarrow A_1, \dots, A_n$ is derivable in $\mathcal{S}_{\text{MIND}^{(+)}}$, then $\Rightarrow A_i$ is derivable in $\mathcal{S}_{\text{MIND}^{(+)}}$ for some $1 \leq i \leq n$.*

Proof. By induction on the height of the derivation of $\Rightarrow A_1, \dots, A_n$. The last rule applied in the derivation is necessarily a propositional right rule, since all other rules (including the modal ones) introduce a formula in the left-hand side of the conclusion. We have the following cases.

(i) The last rule applied is $\wedge_R$:

$$\frac{\Rightarrow B, A_2, \dots, A_n \quad \Rightarrow C, A_2, \dots, A_n}{\Rightarrow B \wedge C, A_2, \dots, A_n} \wedge_R$$

Then by i.h., $\Rightarrow A_i$ is derivable in for some $2 \leq i \leq n$, or both $\Rightarrow B$ and $\Rightarrow C$ are derivable. In the first case the claim holds, in the second case by $\wedge_R$ we obtain $\Rightarrow B \wedge C$.

(ii) The last rule applied is $\vee_R$:

$$\frac{\Rightarrow B, C, A_2, \dots, A_n}{\Rightarrow B \vee C, A_2, \dots, A_n} \vee_R$$

Then by i.h., $\Rightarrow B$, or $\Rightarrow C$, or $\Rightarrow A_i$ is derivable in for some $2 \leq i \leq n$. If $\Rightarrow B$ or $\Rightarrow C$ is derivable, then by wk_R we obtain $\Rightarrow B, C$, then by $\vee_L$ we derive $\Rightarrow B \vee C$.

(iii) The last rule applied is $\supset_R$:

$$\frac{B \Rightarrow C}{\Rightarrow B \supset C, A_2, \dots, A_n} \supset_R$$

From $B \Rightarrow C$, by $\supset_R$ we obtain $\Rightarrow B \supset C$.

Theorem 6 (Disjunction property). *If $A \vee B$ is derivable in MIND (resp. MIND^+), then A is derivable or B is derivable in MIND (resp. MIND^+).*

Proof. Suppose that $\vdash A \vee B$. By the completeness of the calculi, $\Rightarrow A \vee B$ is derivable in $\mathcal{S}_{\text{MIND}^{(+)}}$. The last rule applied in the derivation is necessarily $\vee_R$, with premiss $\Rightarrow A, B$, which is in turn derivable. Then by Lemma 1, $\Rightarrow A$ is derivable or $\Rightarrow B$ is derivable. Finally, from the soundness of the calculi it follows that $\vdash A$ or $\vdash B$.

4 Conclusions

We have shown that intuitionistic logic may provide a meaningful base for deontic logic, by proposing the minimal system MIND and its extension MIND^+ . The

logic MIND has several features: conflicting obligations can be expressed without incurring into triviality; at the same time it satisfies the basic principle that “obligation entails permission” (axiom D). Observe that these two features are incompatible in the classical setting. Moreover the logic MIND rejects monotonicity, which is the source of well-known paradoxes. It equally rejects aggregation of obligations, while retaining (half of) distribution of permission over disjunction; observe that this distinction is not possible on a classical base where Obligation and Permission are defined by duality. Finally, MIND supports the claim that a logical truth is not a valid deontic statement, MIND^+ strengthens this claim by assuming that $\neg\mathcal{O}\top$ and $\neg\mathcal{P}\top$ are valid.

The rejection of the duality between Obligation and Permission is not only technically motivated by the above considerations: it is also grounded on a constructive interpretation of these modalities as it is provided by their intuitionistic base: neither of the two can be defined in terms of the other by negation, the idea is that there must be a positive evidence (a proof) for establishing a permission or an obligation.

We have undertaken some analysis of MIND and MIND^+ : both are characterised by a simple neighbourhood semantics and, as a difference with intuitionistic modal logics (à la Simpson), they enjoy a simple sequent calculus, which provides a decision procedure for them. It should be possible to define strictly-terminating calculi à la Dickhoff for both provability and unprovability for MIND as it is done in [5] for constructive modal logics CK and WK (see also [12]), in particular the unprovability calculus might provide a direct countermodel construction for MIND. We intend to investigate this issue in future work.

If the logic MIND is a minimal system there are several extensions that may be worth studying. In the wait of a deeper study, we mention a few directions: first of all, other deontic modalities can be defined by duality from the basic Permission and Obligation, a kind of weak version of them, and one could investigate what would be their interrelation. On the other hand, an independent Forbidden modality with a constructive meaning makes sense in order to distinguish what is not allowed (for instance by lack of right/authorisation) and what it is explicitly forbidden by legal or moral reasons: compare the two examples (1) “You do not have the permission to post to the mailing-list if you have not subscribed to it.” (2) “You are forbidden to post to the mailing list, if you sent offensive messages.” However we leave this issue open to future discussion.

The “square of opposition” has been considered in both deontic logic [10] and recently in intuitionistic modal logic [16], it could be the right setting to put some order on the set of possible modalities. Furthermore, in a previous work [4] we have proposed a framework for intuitionistic non-normal modal logics, where we have considered three levels of interaction between the two modal operators: it turns out that the logics MIND and MIND^+ satisfy only the lowest level of interaction, as the further two do not tolerate conflicting obligations, but if we are willing to give up this requirement, there are other logical systems which may be worth to investigate in a deontic setting. All of this will be the object of future research.

References

1. Bellin, G., Carrara, M., Chiffi, D.: On an intuitionistic logic for pragmatics. *J. Log. Comput.* **28**(5), 935–966 (2018)
2. Bellin, G., de Paiva, V., Ritter, E.: Extended Curry-Howard correspondence for a basic constructive modal logic. In: *Proceedings of M4M-2* (2001)
3. Chellas, B.F.: *Modal Logic - An Introduction*. Cambridge University Press, Cambridge (1980)
4. Dalmonte, T., Grellois, C., Olivetti, N.: Intuitionistic non-normal modal logics: a general framework. *J. Philos. Log.* **49**(5), 833–882 (2020)
5. Dalmonte, T., Grellois, C., Olivetti, N.: Terminating calculi and countermodels for constructive modal logics. In: Das, A., Negri, S. (eds.) *TABLEAUX 2021. LNCS (LNAI)*, vol. 12842, pp. 391–408. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86059-2_23
6. Elgesem, D.: The modal logic of agency. *Nordic J. Philos. Logic* (1997)
7. Forrester, J.W.: Gentle murder, or the adverbial samaritan. *J. Philos.* **81**(4), 193–197 (1984)
8. Goble, L.: Prima facie norms, normative conflicts, and dilemmas. *Handb. Deontic Logic Normative Syst.* **1**, 241–351 (2013)
9. Hansson, S.O.: Alternative semantics for deontic logic. *Handb. Deontic Logic Normative Syst.* **1**, 445–497 (2013)
10. Hansson, S.O.: The varieties of permission. *Handb. Deontic Logic Normative Syst.* **1**, 195–240 (2013)
11. Hilpinen, R., McNamara, P.: Deontic logic: a historical survey and introduction. *Handb. Deontic Logic Normative Syst.* **1**, 3–136 (2013)
12. Iemhoff, R.: Terminating sequent calculi for two intuitionistic modal logics. *J. Log. Comput.* **28**(7), 1701–1712 (2018)
13. Jones, A.J., Pörn, I.: Ideality, sub-ideality and deontic logic. *Synthese*, 275–290 (1985)
14. Kojima, K.: Relational and neighborhood semantics for intuitionistic modal logic. *Reports Math. Log.* **47**, 87–113 (2012)
15. Lokhorst, G.J.C.: An intuitionistic reformulation of Mally’s deontic logic. *J. Philos. Log.* **42**(4), 635–641 (2013)
16. Mendler, M., Scheele, S., Burke, L.: The došen square under construction: a tale of four modalities. In: Das, A., Negri, S. (eds.) *TABLEAUX 2021. LNCS (LNAI)*, vol. 12842, pp. 446–465. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86059-2_26
17. Parent, X., Gabbay, D., Torre, L.: Intuitionistic basis for input/output logic. In: Hansson, S.O. (ed.) *David Makinson on Classical Methods for Non-Classical Problems*. OCL, vol. 3, pp. 263–286. Springer, Dordrecht (2014). https://doi.org/10.1007/978-94-007-7759-0_13
18. Ross, A.: Imperatives and logic. *Philos. Sci.* **11**(1), 30–46 (1944)
19. Simpson, A.K.: The proof theory and semantics of intuitionistic modal logic. Ph.D. thesis, University of Edinburgh, UK (1994)
20. Troelstra, A.S., Schwichtenberg, H.: *Basic Proof Theory*, 2nd edn. Cambridge tracts in theoretical computer science, vol. 43. Cambridge University Press (2000)
21. Wijesekera, D.: Constructive modal logics I. *Ann. Pure Appl. Log.* **50**(3), 271–301 (1990)
22. von Wright, G.H.: Deontic logic. *Mind* **60**(237), 1–15 (1951)
23. Von Wright, G.H.: New studies in deontic logic. In: Hilpinen, R. (ed.) *On the Logic of Norms and Actions*, vol. 152, pp. 3–35. Springer, Dordrecht (1981). https://doi.org/10.1007/978-94-009-8484-4_1



Presburger Büchi Tree Automata with Applications to Logics with Expressive Counting

Bartosz Bednarczyk^{1,2}  and Oskar Fiuk²

¹ Computational Logic Group, Technische Universität Dresden, Dresden, Germany
bartosz.bednarczyk@cs.uni.wroc.pl

² Institute of Computer Science, University of Wrocław, Wrocław, Poland
307023@uw.edu.pl

Abstract. We introduce two versions of Presburger Automata with the Büchi acceptance condition, working over infinite, finite-branching trees. These automata, in addition to the classical ones, allow nodes for checking linear inequalities over labels of their children. We establish tight NP and EXPTIME bounds on the complexity of the non-emptiness problem for the presented machines. We demonstrate the usefulness of our automata models by polynomially encoding the two-variable guarded fragment extended with Presburger constraints, improving the existing triply-exponential upper bound to a single exponential.

1 Introduction

Description Logics (DLs) [3] are a robust family of first-order logic (FO) fragments designed for knowledge representation and reasoning. They serve as the core of the Web Ontology Languages (OWL) and are successfully used in the bio-medical domain, *e.g.* in Snomed CT [17]. Unfortunately, the expressiveness of DLs is quite limited: as description logics are fragments of FO, they are not able to express even simple statistical properties. Such information is crucial to reason about real-life scenarios as witnessed by the following example:

Example 1. Consider a knowledge base of Citizens voting for their new mayor. During the election, they must decide for each candidate whether they reject him or not. Moreover, each citizen can give at most two accepting votes. This situation can be modelled, in a DL-like language as follows with:

$$\text{Citizen} \sqsubseteq (\leq 2.\text{votedForAcc}).\top, \text{Winner} \sqsubseteq (> 50\%)\text{votedForAcc}^-. \text{Citizen},$$

so the elected winner received more than 50% of accepting votes from Citizens.

The above-mentioned properties can be formalised in extensions of the standard description logic $\mathcal{ALC}$ with Presburger arithmetic, as was done *e.g.* in [1, 15, 19]. However, to model Example 1 one additionally requires the use of inverses of roles, making the logics from the cited papers not directly applicable in our

setting. The decidability status of $\mathcal{ALC}$ with inverses and arithmetics was established only a few months ago in [5], but with a non-optimal 3NEXPTIME complexity. The aim of this short paper is to improve this complexity to a single exponential.

Our Results. We follow a well-established automata-theoretic approach to decidability of description logics, as treated in several papers, see e.g. [2, 11, 12]. We present two models of tree automata, working over infinite finite-branching trees, with Büchi accepting conditions. The first of them, called SPBTA, assumes the powerset alphabet and has an EXPTIME-complete non-emptiness problem. That makes SPBTA an attractive automata model for encoding modal and description logics with Presburger constraints. The other one, called PBTA, works over the usual alphabet and turns out to be NP-complete, complementing already existing results on finite tree automata with expressive counting from [22, 23].

As an application of our results, we provide a translation from an extension of $\mathcal{ALC}$ with expressive cardinality constraints, namely the two-variable guarded fragment with Presburger arithmetics [5], to our automata. This yields EXPTIME-completeness of the mentioned logic. Omitted proofs are available in the extended version published [on the ICCL portal of TU Dresden](#).

Related Automata Models. We briefly discuss what distinguishes our automata models in comparison to the existing models. First of all, our automata work over unordered and unranked trees, hence all different automata models do not apply to our case. Second, our trees are infinite, hence the framework of [22, 23] cannot simply be transferred here. Third, several tree automata models with arithmetics were proposed quite recently [6, 7, 9] but in their case the complexity of the non emptiness problem is either higher EXPTIME or is nondeterministic, disallowing a reasonable translation of description logics into it. Finally, standard automata models used for description logics, e.g. the ones from [8] are not applicable to our case as they cannot impose Presburger constraints on successors.

2 Preliminaries

We assume that the reader is familiar with basics on formal languages, computability [24], and tree automata [14].

Let $\underline{n}$ denote the set $\{k \in \mathbb{N} \mid k < n\}$. By a *tree*, we mean a finite-branching tree, in which every branch is infinite. More formally, $\mathbf{t}$ is a prefix-closed subset of $\mathbb{N}^\omega$, such that for every $x \in \mathbf{t}$ we have $x0 \in \mathbf{t}$ and there is $n \in \mathbb{N}$ so that no xn belongs to $\mathbf{t}$. The elements of $\mathbf{t}$ are called *nodes*. W.l.o.g. we assume that trees are *children-closed*, i.e. if $x \in \mathbf{t}$ and $n \in \mathbb{N}$ then $xn \in \mathbf{t}$ implies xk for all $k \in \underline{n}$. Given $x \in \mathbf{t}$ we define the subtree of x (resp. children of x), denoted $\text{Subt}(x)$ (resp. $\text{Chld}(x)$), as the set $\{z \in \mathbf{t} \mid z = xy\}$ (resp. $\{xn \in \mathbf{t} \mid n \in \mathbb{N}\}$). The *degree* $d(x)$ (resp. *height* $h(x)$) of x is $|\text{Chld}(x)|$ (resp. $|x|$). A *branch* of $\mathbf{t}$ is any infinite sequence $v_0, v_1, v_2, \dots$ of nodes of $\mathbf{t}$, such that $v_0 = \varepsilon$ and for all i the node v_{i+1} is a child of v_i . A Σ -labelled tree is a pair $(\ell, \mathbf{t})$, composed of a tree $\mathbf{t}$ and a

mapping $\ell : \mathbf{t} \rightarrow \Sigma$. Sometimes we will also think about *finite* trees, defined in a natural way, with all the definitions adopted to them as the reader may expect.

By a *linear constraint* we mean an expression of the form

$$C + \sum_{i=0}^n a_i \cdot x_i \bowtie D + \sum_{i=0}^m b_i \cdot y_i,$$

where x_i, y_i are variables from some countably-infinite set $\mathcal{V}$, $\bowtie$ is one of $\leq, <, \equiv_k$ (the last one denotes equality modulo¹ k) and $a_i, b_i, k > 0, C, D$ are integers encoded in *binary*. We define *systems of constraints* and their *solutions over $\mathbb{N}$* in the expected way. Given a solution $S : \mathcal{V} \rightarrow \mathbb{N}$ to a system E , the *support* $\text{supp}(S)$ of S is defined as the set $\{v \in \mathcal{V} \mid S(v) \neq 0\}$.

Testing *solvability* (over $\mathbb{N}$) of systems of constraints is NP-complete [10]. Moreover, the following less-known lemma provides the existence of a “sparse and small” solution; consult: [16, Theorem 1], [1, Lemma 3] and [21].

Lemma 2. *Let $\mathcal{I}$ be a system of E linear constraints with integer coefficients absolutely bounded by C . Assume a solution S to $\mathcal{I}$ (over $\mathbb{N}$). Then there is a solution S' to $\mathcal{I}$ (over $\mathbb{N}$) with $|\text{supp}(S')| \leq 2E \cdot \log(4EC)$ and $\text{supp}(S') \subseteq \text{supp}(S)$. Moreover, the maximal absolute value assigned to variables by S' is bounded by $V(E \cdot C)^{2E+1}$, where V is the total number of variables.*

3 Suitable Tree Automata Models

Henceforth, we consider two tree automata models, starting from the more succinct and expressive one:

Definition 3. *A Succinct Presburger Büchi Tree Automata (SPBTA) $\mathcal{A}$ is a tuple $\mathcal{A} = (\Sigma, Q, F, Q_0, I)$, where all of its components are finite, Σ is an alphabet, Q is a set of states, $F \subseteq Q$ is a set of final states, $Q_0 \subseteq Q$ is an initial configuration and I is a finite set of instructions of the form:*

- $\pm q \rightarrow \pm q', q \wedge q' \rightarrow q''$ for $q, q', q'' \in Q$,
- $q \rightarrow \pm a, \pm a \rightarrow q$ for $q \in Q$ and $a \in \Sigma$, and
- $q \rightarrow \mathcal{C}, \mathcal{C} \rightarrow q$, where $\mathcal{C}$ is a linear constraint over $\mathcal{V} = \{x_S \mid S \subseteq Q\}$. For succinctness, only the variables mentioned in $\mathcal{V}$ are required to be explicitly written in $\mathcal{C}$.

Our automata work over 2^Σ -labelled trees and labels each node with a subset of Q . Such a labelling is required to be compatible with Q_0 , i.e. ε is labelled with Q_0 . Moreover, it should be compatible with I , i.e. that the presence of a state in a node v implies the absence/presence of a state/letter in v , and that linear constraints over the labellings of v 's children, where the value of the variable x_S is the total number of v 's children labelled by some superset of S . Formally:

Definition 4. *A run of an SPBTA $\mathcal{A} = (\Sigma, Q, F, Q_0, I)$ over a 2^Σ -labelled tree $(\ell_{\mathbf{t}}, \mathbf{t})$ is a 2^Q -labelled tree $(\ell_{\mathbf{t}}^{\mathcal{A}}, \mathbf{t})$ s.t. $\ell_{\mathbf{t}}^{\mathcal{A}}(\varepsilon) = Q_0$, and for all $v \in \mathbf{t}$, $q \in \ell_{\mathbf{t}}^{\mathcal{A}}(v)$:*

¹ Constraints with $\equiv_k$ can be eliminated [4].

- if $(q \rightarrow q') \in I$ (resp. $(q \rightarrow -q') \in I$) with $q' \in Q$, then $q' \in \ell_t^A(v)$ (resp. $q' \notin \ell_t^A(v)$), and if $(q \wedge q') \rightarrow q'' \in I$ with $q', q'' \in Q$ then $q, q' \in \ell_t^A(v)$ implies $q'' \in \ell_t^A(v)$,
- if $(q \rightarrow \pm a) \in I$ (resp. $(q \rightarrow -a) \in I$) with $a \in \Sigma$, then $a \in \ell_t(v)$ (resp. $a \notin \ell_t(v)$),
- if $(q \rightarrow C) \in I$ with C being a linear constraint, then $x_S \mapsto |\{u \in \text{Chld}(v) : S \subseteq \ell_t^A(u)\}|$ is a solution to C .

Omitted cases are symmetric. A run is successful, if every branch of $\mathbf{t}$ has inf. many elements v fulfilling the condition: “there is an accepting state $q_f \in F$ such that $q_f \in \ell_A(v)$ ”. An SPBTA $\mathcal{A}$ accepts a 2^Σ -labelled tree $(\ell_t, \mathbf{t})$ if there is a successful run over $(\ell_t, \mathbf{t})$. A language of $\mathcal{A}$, denoted $\mathcal{L}(\mathcal{A})$, is the set of all 2^Σ -labelled trees accepted by $\mathcal{A}$.

The succinct representation of the alphabet and constraints in SPBTA will turn out to be especially useful for encoding models of modal and description logics with Presburger constraints (due to their tree-like model property). This is the biggest advantage of SPBTA in contrast to other automata models mentioned in the introduction. In order to translate logics into automata in the usual setting, the user must first provide a bound on the number of children that a node may have². This essentially forces the user to prove some kind of a “small-branching tree model lemma” before you proceed with tree-encoding. Our automata model will do it automatically Lemma 16, so you can encode tree-models without the need of finding an appropriate bound (which sometimes can be difficult).

In what follows we present yet another model of tree automata, which is simpler due to the use of a standard alphabet. Within a run, nodes are labelled with exactly one state, and therefore, many of the presented instructions can be removed. While PBTA does not have any direct applications to logic, we find the drop of complexity interesting.

Definition 5. A Presburger Büchi Tree Automata (PBTA) $\mathcal{A}$ is a tuple $\mathcal{A} = (\Sigma, Q, F, q_0, I)$, where all its components are finite, Σ is an alphabet, Q is a set of states, $F \subseteq Q$ is a set of final states, $q_0 \in Q$ is an initial configuration and I is a finite set of instructions of the form:

- $(q, a) \rightarrow C$, where $q \in Q$, $a \in \Sigma$ and C is a linear constraint with variables $\mathcal{V} = \{x_q \mid q \in Q\}$.

We define runs of PBTA analogously to the case of SPBTAs.

Definition 6. A run of a PBTA $\mathcal{A} = (\Sigma, Q, F, q_0, I)$ over a Σ -labelled tree $(\ell_t, \mathbf{t})$ is a Q -labelled tree $(\ell_t^A, \mathbf{t})$ such that $\ell_t^A(\varepsilon) = q_0$, and for all $v \in \mathbf{t}$ with $q_v = \ell_t^A(v)$ and $a_v = \ell_t(v)$:

- if $((q_v, a_v) \rightarrow C) \in I$, then $x_q \mapsto |\{u \in \text{Chld}(v) \mid q = \ell_t^A(u)\}|$ is a solution to C .

² Of course we may always restrict ourselves to binary trees by employing standard tricks, but then the verification of Presburger constraints becomes a challenge.

A run is successful, if every branch of $\mathbf{t}$ has infinitely many elements v fulfilling the condition: “there is an accepting state $q_f \in F$ such that $q_f = \ell_{\mathcal{A}}(v)$ ”.

The notion of accepted trees and the *language* of PBTA are defined analogously to the case of SPBTA. Note that the languages accepted by (S)PBTA are no longer first-order definable. Consider:

Example 7. A language $\{\mathbf{t} \mid \forall v \in \mathbf{t} \quad d(v) \equiv_2 0\}$ can be recognized by the PBTA $\mathcal{A} = (\{a\}, \{q\}, \{q\}, q, \{(q, a) \rightarrow x_q \equiv_2 0\})$.

The *non-emptiness* problem asks whether the language of a given (S)PBTA is non-empty. An immediate reduction from the solvability (over $\mathbb{N}$) of systems of linear equations yields NP-hardness for testing non-emptiness of PBTA.

Lemma 8. *The non-emptiness for PBTA is NP-hard.*

Proof. Take a system of linear equations $\mathcal{I}$ and let $\Sigma = \{a_0\}, Q = F = \{q_v \mid v \in \mathcal{V}(\mathcal{I})\}, I = \{(q_v, a_0) \rightarrow \mathcal{C} \mid v \in \mathcal{V}(\mathcal{I}), \mathcal{C} \in \mathcal{I}\}$. Choose v_0 to be an arbitrary variable in $\mathcal{V}(\mathcal{I})$. Then the language recognized by automaton $\mathcal{A} = (\Sigma, Q, F, q_{v_0}, I)$ is non-empty iff $\mathcal{I}$ has a solution.

In the sequel, we provide a matching upper bound.

4 Non-emptiness Problem for PBTA is in NP

Henceforth, we fix a PBTA $\mathcal{A} = (\Sigma, Q, F, q_0, I)$. By its size, denoted with $|\mathcal{A}|$, we mean the size of some reasonable succinct representation of $\mathcal{A}$, where the numbers in constraints are encoded in *binary*. We start with auxiliary definitions.

Definition 9. Let $(\ell_{\mathcal{A}}, \mathbf{t})$ be an accepting run of $\mathcal{A}$ on a tree $(\ell_{\mathbf{t}}, \mathbf{t})$. Then a state $q \in Q$ has an occurrence in a node $v \in \mathbf{t}$ iff $\ell_{\mathcal{A}}(v) = q$. If there is $v \in \mathbf{t}$, such that q occurs in v , then q has an occurrence in an automaton run $(\ell_{\mathcal{A}}, \mathbf{t})$. An occurrence of $q \in Q$ in $v \in \mathbf{t}$ is safe iff either $q \in F$ or on each branch $v_1, v_2, v_3, \dots$ starting in v there exists an occurrence of some state from F before the next occurrence of q (if any), i.e. $\inf\{i \mid \ell_{\mathcal{A}}(v_i) \in F\} < \inf\{i \mid \ell_{\mathcal{A}}(v_i) = q\}$. A safe subtree of a safe occurrence $q \in Q$ in $v \in \mathbf{t}$, denoted as $\text{SSubt}(q, v)$, is a pair $(v, \mathbf{t}')$ such that $\mathbf{t}'$ is a singleton $\{\varepsilon\}$ whenever $q \in F$ or otherwise is a finite tree $\{x \mid vx \in \text{Subt}(v), \forall st = x, t \neq \varepsilon, \ell_{\mathcal{A}}(vs) \notin F\}$, i.e. accepting states occur only in leaves.

Lemma 10. *If $q \in Q$ has an occurrence in an accepting run $(\ell_{\mathcal{A}}, \mathbf{t})$ of $\mathcal{A}$, then it has a safe one.*

Proof. Suppose that there is $q \in \ell_{\mathcal{A}}(\mathbf{t})$, such that no occurrence is safe. Clearly, $q \notin F$. Then one can construct an infinite branch without any state from F , yielding a contradiction with the Büchi acceptance condition. Employ the following process. Start with any occurrence of q in some $v \in \mathbf{t}$. Then by assumption it is not safe. Therefore, it contains a branch that reaches another occurrence of q in some other v' before reaching any state from F . Repeat from v' .

To get an NP-upper bound, we construct a *regular tree*, generated by a pair of functions and a state, that will serve as a witness for the non-emptiness test of a given automaton.

Definition 11. For functions $\alpha : Q \rightarrow \Sigma$, $\gamma : Q \rightarrow Q^+$, and an element $q_0 \in Q$, define inductively a family of sets:

$$- T_0 = \{(\varepsilon, q_0)\}, \text{ and } T_{n+1} = \{(xk, \gamma(q)_k) \mid (x, q) \in T_n, k = 0, \dots, |\gamma(q)| - 1\}.$$

Let $\mathcal{T} = \bigcup_{n \geq 0} T_n$. Then

- $\mathbf{t} = \{x \mid (x, q) \in \mathcal{T}\}$ is a tree,
- ℓ_Q is taken as any Q -labelling of $\mathbf{t}$ such that $(x, \ell_Q(x)) \in \mathcal{T}$,
- ℓ_Σ is taken as any Σ -labelling of $\mathbf{t}$ such that $\ell_\Sigma(x) = \alpha(\ell_Q(x))$.

The role of α is to name elements with letters and the role of γ is to create fresh state-labelled children of a given node. Note that we are working with unordered trees, thus w.l.o.g. we assume that γ is “sorted”. Thus γ can be represented as a function $\Gamma : Q \rightarrow \mathbb{N}^Q$ defined as:

$$\gamma(q) = \underbrace{q_1 \dots q_1}_{\Gamma(q)(q_1) \text{ times}} \dots \underbrace{q_{|Q|} \dots q_{|Q|}}_{\Gamma(q)(q_{|Q|}) \text{ times}}.$$

The role of Γ is to encode potential solutions to a system of inequalities from the I component of $\mathcal{A}$. Finally, we say that $(\ell_Q, \mathbf{t})$ (resp. $(\ell_\Sigma, \mathbf{t})$) is *Q -generated* (resp. *Σ -generated*) tree by a triple (q_0, α, Γ) .

Lemma 12. Let $u = (q_0, \alpha, \Gamma)$ generate a tree $\mathbf{t}$ accepted by $\mathcal{A}$. Then there is a triple $u' = (q_0, \alpha, \Gamma')$ generating another tree $\mathbf{t}'$ that is still accepted by $\mathcal{A}$, and which has small values in Γ , i.e. $\log \max\{\Gamma(q)(q') \mid q, q' \in Q^2\} = O(\text{poly}(|\mathcal{A}|))$.

Proof. Since u generates $\mathbf{t}$ accepted by $\mathcal{A}$, the values of $\Gamma(q)$ (after a renaming) are solutions to constraints $q \rightarrow \mathcal{C}$ given by $\mathcal{A}$. By Lemma 2 we get smaller solutions that can be encoded as Γ' . Then the tree $\mathbf{t}'$ generated by (q_0, α, Γ') is clearly accepted by $\mathcal{A}$. Indeed, suppose that there is a branch with only finitely many accepting states. Then, by the inclusion of supports for the solutions and due to the way the tree $\mathbf{t}'$ is generated, there will be a branch in the original tree $\mathbf{t}$ with the same Q -labelling, yielding a contradiction.

The next crucial lemma shows how to generate a small regular tree, serving as a witness for our non-emptiness test.

Lemma 13. If $\mathcal{L}(\mathcal{A})$ is nonempty, then there is a tree $\mathbf{t}$ and a triple $u = (q_0, \alpha, \Gamma)$ satisfying: (1) $(\ell_\Sigma, \mathbf{t}) \in \mathcal{L}(\mathcal{A})$, and is Σ -generated by u , (2) $(\ell_Q, \mathbf{t})$ is an accepting run of $\mathcal{A}$ on the tree $(\ell_\Sigma, \mathbf{t})$, and is Q -generated by u , and (3) the triple u has size polynomial in $|\mathcal{A}|$.

Proof. Fix a tree $(\ell_{\mathbf{t}'}, \mathbf{t}') \in \mathcal{L}(\mathcal{A})$ and a corresponding accepting run $(\ell_{\mathbf{t}'}^A, \mathbf{t}')$ of $\mathcal{A}$. Let $Q_0 = \{q_1, \dots, q_k\}$ be the set of states appearing in $\ell_{\mathbf{t}'}^A(\mathbf{t}')$. For each $q_i \in Q_0$ fix a safe occurrence in some $v_i \in \mathbf{t}'$ and let $(u_i, \mathbf{t}_i) = \text{SSubt}(q_i, v_i)$. We also fix some post-order enumeration of nodes from $\mathbf{t}_i$ and let $\text{PO}(\mathbf{t}_i, v)$ denotes the position of $v \in \mathbf{t}_i$ in such order.

A *solution* in a node v is a function $S_v : Q \rightarrow \mathbb{N}$ such that

$$S_v(q) = |\{u \in \text{Chld}(v) : \ell_{\mathbf{t}'}^A(u) = q\}|.$$

We define a *witness* injection $j : Q_0 \rightarrow \mathbb{N}^2 \times \mathbb{N}^*$, that will be used to define α and Γ components of u , as follows:

$$j(q_i) = \min\{(r, s, x) \mid x \in \mathbf{t}_r, q_i = \ell_{\mathbf{t}'}^A(u_r x), \text{PO}(\mathbf{t}_r, x) = s\},$$

where the minimum is taken with respect to the lexicographic order. The intuition behind the function j is that it assigns, for each state, the deepest leftmost occurrence of this state in the safe subtree with the lowest possible index. Therefore, if the node is not a leaf, any outgoing edge leads to a child with a strictly smaller value of j . Since all leaves are labelled with accepting states, any branch in the resulting tree that visits the same node twice must pass through the accepting state.

Suppose that $j(q_i) = (r_i, s_i, x_i)$. Then we simply put $\alpha(q_i) = \ell_{\mathbf{t}'}(u_{r_i} x_i)$ and $\Gamma(q_i) = S_{u_{r_i} x_i}$ for each $q_i \in Q_0$, and let $\alpha(q) = \perp$ and $\Gamma(q) = \perp$ for each $q \in Q \setminus Q_0$. One can show (see Appendix A) that $(\ell_Q, \mathbf{t})$ is indeed an accepting run of $\mathcal{A}$ on the tree $(\ell_{\Sigma}, \mathbf{t})$ with respect to the given definition of (q_0, α, Γ) . Thus, by applying Lemma 12 we conclude that u has a polynomial size description.

Lemma 14. *Given $\mathcal{A}$ and a tuple $u = (q_0, \alpha, \Gamma)$, it is in PTIME to test if u generates a tree from $\mathcal{L}(\mathcal{A})$.*

Proof. We can clearly verify all constraints from the I component of $\mathcal{A}$, since Γ encodes all the required solutions. For the Büchi acceptance condition build a directed graph on the vertices Q and edges $E = \{(q, q') \in Q^2 \mid q \notin F, \Gamma(q)(q') > 0\}$. Then if the graph (Q, E) is acyclic, output **Yes**, and **No** otherwise.

Thus as a corollary of Lemmas 8, 13, 14 we conclude:

Theorem 15. *The non-emptiness problem for Presburger Büchi Tree Automata is NP-complete.*

5 Non-emptiness for SPBTA

Fix an SPBTA $\mathcal{A} = (\Sigma, Q, F, Q_0, I)$. W.l.o.g. we can assume that Σ is empty; if not, then we simply introduce fresh states q_a per each letter $a \in \Sigma$ and replace all occurrences of a in I by q_a . Take E be the number of constraints in I and let C be the minimal absolute bound on the coefficients from I .

We start with a “small accepted tree lemma”:

Lemma 16. *If there is an accepting run $(\ell_t^A, \mathbf{t})$ of $\mathcal{A}$ on some tree $(\ell_t, \mathbf{t})$, then there is a tree $(\ell_{t'}, \mathbf{t}')$ with an accepting run $(\ell_{t'}^A, \mathbf{t}')$ of $\mathcal{A}$ satisfying the following conditions:*

- for all nodes $v \in \mathbf{t}'$ we have $|\text{Chld}(v)| \leq 2^{|Q|}(E \cdot C)^{2E+1}$ and $|\{\ell_{t'}^A(u) \mid u \in \text{Chld}(v)\}| \leq 2E \cdot \log(4CE)$.
- On any path $v_1, v_2, \dots, v_n$ of length $n > 2^{|Q|}$ in $\mathbf{t}'$ there is a node v such that $\ell_{t'}^A(v)$ contains a final state.

Proof. We construct $(\ell_{t'}^A, \ell_{t'}, \mathbf{t}')$ recursively, starting from $(\ell_t^A, \ell_t, \mathbf{t})$. The second condition can be established simply by taking any finite path $v_1, v_2, \dots, v_n$ of length $n > 2^{|Q|}$ violating it and by observing, by the pigeonhole principle, that two nodes v_i, v_j ($i < j$) have the same value of $\ell_{t'}^A$. Hence, we obtain a new tree by replacing the subtree of v_i by a subtree of v_j . We repeat the process indefinitely. For the first condition, assume that there is a node v violating the condition. Per each subset $X \subseteq Q$ let v_X be any fixed child of v with $\ell_{t'}^A(v_X) = X$ (if it exists). Let $\mathcal{I} = \{\mathcal{C} \mid q \in \ell_{t'}^A(v), (q \rightarrow \mathcal{C}) \in I\}$ be a system of inequalities. Note that in $\mathcal{I}$ we have unknowns of the form x_X per each $X \subseteq Q$, which by the semantics of $\mathcal{A}$ counts nodes labelled by some superset of X . But $\mathcal{I}$ can be rewritten into a system $\mathcal{I}'$ (by applying the inclusion-exclusion principle) so that the unknowns x_X indicates the total number of children labelled by *precisely the set* X . Thus, the assignment $S : X \mapsto |\{u \in \text{Chld}(v) \mid \ell_{t'}^A(u) = X\}|$ is the solution to $\mathcal{I}'$. By employing Lemma 2 to $\mathcal{I}'$ we infer the existence of a “sparse and small” solution S' to $\mathcal{I}'$ with $\text{supp}(S') \subseteq \text{supp}(S)$. We conclude by modifying $\mathbf{t}'$ as follows: we remove all descendants of v and for any subset $X \subseteq Q$ with $S'(x_X) \neq 0$ we assign as children of v precisely $S'(x_X)$ copies of the previously selected elements v_X . The labellings $\ell_{t'}^A, \ell_{t'}$ are as indicated by the “copies”. The resulting tree is accepted by $\mathcal{A}$: systems of constraints are satisfied (again by applying the inclusion-exclusion principle on S') and the Büchi acceptance condition still holds, since the “copied” subtrees were present in the original run.

-
-
- 1 **Guess** the root node $v = (q, \ell)$ and set **FCnt** = 0.
 - 2 **If** **FCnt** > $2^{|Q|}$ **then** *Accept* **else** set **QCnt** = 0.
 - 3 **Guess** $\leq 2E \cdot \log(4CE)$ nodes (q_i, ℓ_i) that will be repeated **guessed**
 $n_i \leq 2^{|Q|}(E \cdot C)^{2E+1}$ times.
 - 4 Check if the guessed children are consistent with the transition function of $\mathcal{A}$.
 - 5 **Universally choose** (q_i, ℓ_i) as $v = (q, \ell)$.
 - 6 **If** q contains a final state **then** **FCnt** ++ and **goto** 2.
 - 7 **QCnt** ++. **If** **QCnt** > $2^{|Q|}$ **then** *Reject* **else goto** 3.
-

We next describe APSPACE (= EXPTIME by [13]) procedure for testing non-emptiness of $\mathcal{L}$. It will be tableaux-like: we start by guessing a node, then we guess its children, verify the consistency of the guess with the transition function

of $\mathcal{A}$ and repeat the process from a universally selected child. In the pseudocode below we identify a node with a pair $(q, \ell) \in 2^Q \times 2^\Sigma$. We employ two counters, stored in binary. The counter **FCnt** is responsible for counting how many times we encountered an accepting state (if **FCnt** $> 2^{|Q|}$ we know that we ended up in the same accepting configuration twice and hence, we can build the tree by making precisely the same choices as we did in the past). The counter **QCnt** is responsible for counting how many configurations we have visited without entering into an accepting state. If **QCnt** $> 2^{|Q|}$ then some non-final state was surely repeated twice, violating the second condition of Lemma 16.

Speaking informally, the role of counters is to prevent us from mixing-up the numbers of already visited non-accepting and accepting elements on a branch: we want to make sure that there is a repetition of an accepting state but the path between two accepting nodes can be long (more than 2^Q), which potentially spoils the counting. The second counter ensures us that we count correctly.

A proof of correctness can be found in Appendix B. We conclude:

Theorem 17. *The non-emptiness problem for SPBTAs is decidable in EXPTIME.*

6 Applications in Logic

We assume familiarity the basics on (description) logics and model theory [3, 20]. The two-variable guarded fragment of first-order logic, GF^2 , is a relevant fragment of FO that captures many standard description logics up to $\text{ALCITH}^{\text{self}}$ [3, 18]. Here we consider its expressive extension with *local Presburger quantifiers*, GP^2 , introduced in [5]. For brevity we introduce it already in a suitable, polynomially computable, normal form [5, p. 10]. A formula φ is in GP^2 if it has the shape:

$$\forall x \gamma(x) \wedge \bigwedge_{i=1}^n \left(\forall x \forall y E_i(x, y) \rightarrow \alpha_i(x, y) \right) \wedge \bigwedge_{i=1}^m \forall x A_i(x) \rightarrow \left(\sum_{j=1}^{m_i} \lambda_{i,j} \cdot \#_y^{R_{i,j}}[\top] \otimes_i \delta_i \right),$$

where $\gamma(x)$, $\alpha_i(x, y)$ are quantifier-free formulae in NNF, each $e_i, A_i, r_{i,j}$ are relational symbols, all $\lambda_{i,j}$'s, δ_i 's are integers, and $\otimes_i$ is one of $=, \neq, \leq, \geq, \equiv_d$ or $\not\equiv_d$, where $d \in \mathbb{N}_+$. Here $\equiv_d$ denotes the congruence modulo d . W.l.o.g. assume that there is a symbol D so that any element is required to a D -successor.

We restrict ourselves to finite-branching structures (the logic trivialises if we allow for infinite branching). The semantics of GP^2 is like in FO, except for the expressions $\#_y^{R_i}[\top]$, which evaluate to the total number of R_i -successors y of x .

$$\mathfrak{A}, x/a \models \sum_{i=1}^n \lambda_i \cdot \#_y^{R_i}[\top] \otimes \delta \quad \text{iff} \quad \sum_{i=1}^n \lambda_i \cdot |\{b \mid \mathfrak{A}, x/a, y/b \models R_i(x, y)\}| \otimes \delta$$

A GP^2 -knowledge-base (kb for brevity) is a pair $\mathcal{K} = (\mathbb{D}, \mathcal{T})$, where $\mathbb{D}$ is a database, i.e. the set of ground facts employing indiv. names and unary/binary predicates, and $\mathcal{T}$ is a set of GP^2 formulae. A structure $\mathfrak{A}$ satisfies $\mathcal{K}$ (written

$\mathfrak{A} \models \mathcal{K}$) if $\mathfrak{A} \models \bigwedge \mathcal{T} \wedge \bigwedge \mathbb{D}$. In the kb-satisfiability problem we ask whether an input GP^2 -kb has a (finite branching, possibly infinite) model. Note that this generalises the usual formula satisfiability. Fix a GP^2 -kb $\mathcal{K} = (\mathbb{D}, \mathcal{T})$. The main goal is to construct an SPBTA $\mathcal{A}$ from $\mathcal{K}$ such that $\mathcal{L}(\mathcal{A})$ is non-empty iff $\mathcal{K}$ has a model.

We first argue that we can restrict ourselves to tree-like models (*i.e.* models whose underlying structure is a tree). Let $\mathfrak{A}$ be a (finite branching) model of $\mathcal{K}$. Let $G = (V, E)$ be a Gaifman graph of $\mathfrak{A}$, *i.e.* we have domain elements of $\mathfrak{A}$ as nodes of G and undirected edges in G between any two elements linked by some binary relation. Let $\text{names}(\mathcal{K})$ be the set of all individual names appearing in $\mathcal{K}$. By a $\mathcal{K}$ -named element in $\mathfrak{A}$ we mean any element $a^{\mathfrak{A}}$ for $a \in \text{names}(\mathcal{K})$. By a $\mathcal{K}$ -path in $\mathfrak{A}$ we mean any word $v_1, v_2, \dots, v_n$ from A^+ such that: (i) v_1 is $\mathcal{K}$ -named in $\mathfrak{A}$, (ii) v_2 is not $\mathcal{K}$ -named, (iii) any two v_i, v_{i+1} are connected by an edge in G , and (iv) any three consecutive v_i, v_{i+1}, v_{i+2} are pairwise different.

Definition 18. *The $\mathcal{K}$ -unravelling of $\mathfrak{A}$ is a structure $\mathfrak{B}$, which domain is composed of all $\mathcal{K}$ -paths in $\mathfrak{A}$ and the relational symbols appearing in $\mathcal{K}$ are interpreted as follows for all words $\text{wd} \in A^*A$ and $\text{wde} \in A^*AA$:*

- $a^{\mathfrak{B}} = a^{\mathfrak{A}}$ for all $a \in \text{names}(\mathcal{K})$, and $\text{wd} \in P^{\mathfrak{B}}$ iff $d \in P^{\mathfrak{A}}$ for all unary P ,
- For any $x, y \in B$ we put $(x, y) \in R^{\mathfrak{B}}$ if and only if:
 - (i) $x = a^{\mathfrak{B}}, y = b^{\mathfrak{B}}$ and $(a^{\mathfrak{A}}, b^{\mathfrak{A}}) \in R^{\mathfrak{A}}$, (ii) $x = y = \text{wd}$ and $(d, d) \in R^{\mathfrak{A}}$, (iii) $x = \text{wd}, y = \text{wde}$ and $(d, e) \in R^{\mathfrak{A}}$, or (iv) $x = \text{wde}, y = \text{wd}$ and $(e, d) \in R^{\mathfrak{A}}$.

The following fact justifies the use of unravellings and follows directly from the semantics of GP^2 (after a trivial renaming of structures).

Fact 19. $\mathfrak{B} \models \mathcal{K}$ and for any $a \in \text{names}(\mathcal{K})$ the substructure of $\mathfrak{B}$ restricted to all $\mathcal{K}$ -paths starting from $a^{\mathfrak{A}}$ constitute a tree.

Call a database *simple*, if it has the form $\{U(a)\}$ for some indiv. name a and a unary predicate symbol U . It is a standard trick in DLs that one can reduce the kb-satisfiability problem to polynomially many subtasks of kb-satisfiability for instances having simple databases only. Indeed, observe first that the unravelled model $\mathfrak{B}$ is just a collection of trees rooted at named elements, were the roots can be interlinked by some binary relations. As the number of individual names appearing in $\text{names}(\mathcal{K})$ is clearly linear in $|\mathcal{K}|$, one can simply guess the substructure $\mathfrak{C} := \mathfrak{B} \upharpoonright \{a^{\mathfrak{B}} \mid a \in \text{names}(\mathcal{K})\} \models \mathbb{D}$ and after verifying its consistency (doable in polynomial time), encode connection between a given root and the other roots as follows. Let us introduce fresh unary predicates $\text{DB}_{\pm R(a,b)}, \text{DB}_{\pm U(a)}, \text{Root}_a$ per each binary relation symbol R and names $a, b \in \text{names}(\mathcal{K})$, and a binary predicate symbol insideDB . Fix a name $a \in \text{names}(\mathcal{K})$; we shall write a reduction for an $a^{\mathfrak{B}}$ point of view. The reduction for other individual names is analogous. We append to the $\mathcal{T}$ -component of $\mathcal{K}$ the following formulae: (i) whenever Root_a is satisfied at x then for each $b \neq a \in \text{names}(\mathcal{K})$ we have that x has precisely one insideDB -successor y labelled with Root_b ; moreover the pair (x, y) should satisfy precisely the same unary and binary relations as indicated in $\mathfrak{C}$. (ii) Whenever x satisfies Root_a , it encodes the

full structure $\mathfrak{C}$ with the help of unary predicate symbols $\text{DB}_{\pm R(a,b)}$, $\text{DB}_{\pm U(a)}$, e.g. if $(c^{\mathfrak{B}}, d^{\mathfrak{B}}) \notin S^{\mathfrak{B}}$ then the satisfaction of Root_a implies the satisfaction of $\text{DB}_{-S(c,d)}$. Note that we do not impose any constraints on successors of the elements labelled with Root_b for $b \neq a$. This will be taken into account in the reduction for other individual names (here the role of such elements is only to make the counting constraints of $\mathcal{K}$ work). This yields the desired reduction. Take $\mathcal{K}_a^{\mathfrak{C}} = (\{\text{Root}_a(a)\}, \varphi_a^{\mathfrak{C}} := \bigwedge_{\varphi \in \mathcal{T}} \varphi \wedge \psi_a^{\mathfrak{C}})$, where $\psi_a^{\mathfrak{C}}$ simply contains the extra formulae that we discussed. It is easy to show that:

Lemma 20. *A GP^2 -kb $\mathcal{K}$ is satisfiable iff there is a structure $\mathfrak{C} \models \mathbb{D}$ (with at most $|\text{names}(\mathcal{K})|$ elements) such that all $\mathcal{K}_a^{\mathfrak{C}}$ (defined above) are satisfiable.*

Proof. ($\Rightarrow$). Take the unravelled model $\mathfrak{B}$ of $\mathcal{K} = (\mathbb{D}, \mathcal{T})$, and let $\mathfrak{C}$ to be the restriction of $\mathfrak{B}$ to the roots of $\mathfrak{B}$. Make $a^{\mathfrak{B}}$ to be the only elements satisfying the symbol Root_a and interpret predicate symbols $\text{DB}_{\pm R(a,b)}$, $\text{DB}_{\pm U(a)}$ in $\mathfrak{C}$ in a above-described way. Finally let $\text{insideDB}^{\mathfrak{B}}$ be the full (non-reflexive) binary relation on $\mathfrak{C}$. Take $\mathfrak{B}_a$ as the substructure of $\mathfrak{B}$ obtained by removing all pairs having the form $(b^{\mathfrak{B}}, c^{\mathfrak{B}})$ for $b \neq a$ from all binary relations. It follows that $\mathfrak{B}_a \models \mathcal{K}_a$. ($\Leftarrow$). Let $\mathfrak{B}_a$ be tree-models of $\mathcal{K}_a^{\mathfrak{C}}$. Thus, by definition of $\text{DB}_{\pm R(a,b)}$, $\text{DB}_{\pm U(a)}$ predicates, there is unique $\mathfrak{C} \models \mathbb{D}$ described inside $\mathcal{K}_a^{\mathfrak{C}}$. Let $\mathfrak{C}_a$ be a substructure of $\mathfrak{B}_a$ obtained by removing elements that are descendants of insideDB -successors of $a^{\mathfrak{B}_a}$ labelled with Root_b for $b \in \text{names}(\mathcal{K})$. Take $\mathfrak{D}$ to be the disjoint union of all $\mathfrak{C}_a$ with $a^{\mathfrak{D}}$ interpreted as the root of $\mathfrak{C}_a$. Finally, make the restriction of $\mathfrak{D}$ to the roots of $\mathfrak{C}_a$ isomorphic to $\mathfrak{C}$. The resulting structure $\mathfrak{D}$ is a model of $\mathcal{K}$.

After the above reduction, we are going to explain how we will encode tree-like models inside trees. For the rest of the paper fix the kb $\mathcal{K}_a^{\mathfrak{C}} = (\{\text{Root}_a(a)\}, \varphi_a^{\mathfrak{C}})$. We first prepare the alphabet for our automaton. Let $\mathcal{S}_0$ be the set of all possible binary and unary atoms for predicate names appearing in $\varphi_a^{\mathfrak{C}}$, involving only the variables x and y . Take $\mathcal{S}_1$ to be composed of all subformulae of α_i and γ from $\varphi_a^{\mathfrak{C}}$ (consult the normal form if needed). Finally take³ $\mathcal{S} := \mathcal{S}_0 \cup \mathcal{S}_1$ and close it under a single negation. Note that $|\mathcal{S}|$ is of size polynomial in $|\mathcal{K}_a^{\mathfrak{C}}|$.

Let the alphabet Σ be equal to

$$\Sigma := \{\sigma_{\psi} \mid \psi \in \mathcal{S}\} \cup \{\text{Root}_a, \text{Parent}, \text{Self}, \text{Child}\}. \quad (1)$$

The intuition behind Σ is as follows. Roughly speaking, whenever a symbol σ_{ψ} will label a node in a tree-encoding of a tree-like structure $\mathfrak{A}$, this will indicate that the corresponding element in $\mathfrak{A}$ satisfies ψ . To speak about formulae satisfied at a node, we always think that the variable y is evaluated at the current node, while the variable x corresponds to its parent. In order to verify cardinality constraints, note that SPBTA can count only node's children, while the expressions $\#_y^{\text{R}_{i,j}}[\top]$ can also count the node itself as a witness as well as its parent. Thus a naive translation to automata may destroy counting constraints. To handle this issue, every node in a tree-encoding of $\mathfrak{A}$ will have two extra

³ In the literature $\mathcal{S}$ appears under the name of a syntactic closure.

children, being “copies” of itself and its parent. This, after the introduction of some simple consistency rules, will make cardinality constraints work.

The formal definition of the encoding is given next. This is going to be *very* technical but we are required such a notion in order capture all necessary properties. The first two items imply the satisfaction of $\mathcal{K}_a^c$, while the others are responsible for the correct evaluation of formulae and for propagating atoms from parent to child as well as to different copies.

Definition 21. *A tree encoding of $\mathcal{K}$ is a Σ -labelled tree $\mathbf{t}$ satisfying:*

- *The root of $\mathbf{t}$ is the unique node in $\mathbf{t}$ labelled with Root_a .*
- *If v is a Child-labelled node or the root of $\mathbf{t}$ then it is labelled with σ_γ and if v is labelled by $\sigma_{A_i(y)}$ then the total number of its children labelled by $\sigma_{R_{i,j}(x,y)}$ satisfies the cardinality constraint $\otimes_i \delta_i$ (cf. normal form).*
- *The letters Parent, Child, Self and Root_a label disjoint sets of nodes in $\mathbf{t}$. Every Child-labelled node has a unique child labelled with Parent and a unique child labelled with Self. The root of $\mathbf{t}$ has a unique Self-labelled child, but it does not have any Parent-labelled children.*
- *If the root of $\mathbf{t}$ is labelled by some σ_ψ then ψ uses the variable y only.*
- *For every unary (resp. binary) symbol U (resp. B) from $\mathcal{K}_a^c$ and every node v from $\mathbf{t}$, v is labelled by precisely one of $\sigma_{\pm U(y)}$ (resp. by one of $\sigma_{\pm B(y,y)}$). If v is neither the root of $\mathbf{t}$ nor the Self-labelled child of the root, then we also take all other atomic formulae involving x and/or y into account.*
- *Labelling of nodes is consistent, i.e. a node v cannot be labelled by both σ_λ and $\sigma_{\neg\lambda}$ and that if v is labelled by $\sigma_{\lambda\wedge\lambda'}$ (resp. $\sigma_{\lambda\vee\lambda'}$) then v is labelled by both (resp. one of) $\sigma_\lambda, \sigma_{\lambda'}$ (the converse also holds for $\sigma_{\lambda\vee\lambda'}, \sigma_{\lambda\wedge\lambda'}$ from Σ).*
- *If v is a Child-labelled node or the root of $\mathbf{t}$, then (i) if v is labelled by some letter $\sigma_{\pm\lambda(y)}$ then every Child-labelled child of v is labelled by $\sigma_{\pm\lambda(x)}$, the Self-labelled child of v is labelled with both $\sigma_{\pm\lambda(y)}$ and $\sigma_{\pm\lambda(x)}$ as well as by letters obtained by replacing binary relations $B(y,y)$ with $B(x,y)$; (ii) if v is labelled by some letter $\sigma_{\pm\lambda(x,y)}$ then its Parent-labelled child is labelled by $\sigma_{\pm\lambda'(x,y)}$, where λ' is obtained by replacing every occurrence of the variable x with y and vice-versa.*

A crucial lemma, linking tree encodings with tree models is:

Lemma 22. *There exists a tree encoding of $\mathcal{K}$ iff $\mathcal{K}_a^c$ is satisfiable.*

Proof. ($\Leftarrow$). Let $\mathbf{t}$ be the promised tree encoding. First, remove all Parent-labelled and Self-labelled children from $\mathbf{t}$ together with their subtrees. Then, we interpret unary symbols U so that its interpretation contains precisely the nodes labelled with $\sigma_{U(y)}$. For binary relations R we take the union of two cases: (i) we include all pairs (v, v) for v labelled with $\sigma_{R(y,y)}$, (ii) we include all parent-child pairs (v, w) (resp. child-parent pairs (w, v)) for which we have that v is labelled with $\sigma_{R(x,y)}$ (resp. $\sigma_{R(y,x)}$). Let $\mathfrak{A}$ be the obtained structure. It is immediate to see by construction that $\mathfrak{A}, x/v, y/w \models \pm\lambda$ whenever w is labelled with $\sigma_{\pm\lambda}$ where λ ranges over subformulae from $\mathcal{S}$ and v is a parent of w . The constraints

from $\mathcal{K}_a^c$ are fulfilled due to our trick with extra nodes (we took parents and the elements themselves into account when counting). Hence, $\mathfrak{A} \models \mathcal{K}_a^c$.

($\Rightarrow$). For the opposite side we will take a tree model $\mathfrak{A} \models \mathcal{K}_a^c$ and “convert it” into a tree representation. As the preliminary step, label all the elements of $\mathfrak{A}$ but root with a Child predicate symbol. After the first step, we select any Child-labelled element v , create a fresh copy of its parent w and insert a copy of w as a child of v . We maintain all the relations between v and the copy of w as they were between v and w . Finally, we make the Parent predicate to be satisfied at the copy of w . After the second step, for every element v in $\mathfrak{A}$ we copy v and make the copy a child of v and make the predicate Self to be satisfied at the copy. Then each of these copies is connected to their original elements with all binary R for which v has an R -loop. We repeat the process infinitely often in order to provide a child per each node. Finally, take any element v and label it with all predicates from $\{\sigma_{\pm\lambda(y)} \mid \mathfrak{A}, v \models \lambda, \lambda \in \mathcal{S}\}$, and whenever v has a Child-labelled child w then we additionally make w to be labelled with $\{\sigma_{\pm\lambda(x)}, \sigma_{\pm\lambda'(x,y)} \mid \mathfrak{A}, x/v, y/w \models \lambda', \mathfrak{A}, x/v \models \lambda, \lambda, \lambda' \in \mathcal{S}\}$. It is immediate to check that the structure $(\ell_t, t = A)$ with ℓ_t inherited from $\mathfrak{A}$ is a tree-encoding.

Relying on Definition 21, the construction of a polynomial-size SPBTA $\mathcal{A}$ accepting Σ -labelled tree encoding is rather straightforward. Thus, we delegate it to Appendix C. Since the satisfiability of GP^2 is known to be EXPTIME -hard (inherited e.g. from the global satisfiability problem for the modal logic K), we conclude the last result of the paper:

Theorem 23. *The knowledge-base satisfiability problem for GP^2 over finite-branching models is EXPTIME -complete. Hence, the non-emptiness problem for SPBTA is EXPTIME -hard.*

Acknowledgements. This work was supported by the Polish Ministry of Science and Higher Education program “Diamantowy Grant” no. DI2017 006447. We are grateful to Tim Lyon and Reijo Jaakkola for proofreading and language improvements.

Results from this paper will appear in the BSc thesis of Oskar Fiuk, written under informal supervision of Bartosz Bednarczyk at the University of Wrocław. Oskar Fiuk was supported by NCN grant No. 2021/41/B/ST6/00996.

References

1. Baader, F.: A new description logic with set constraints and cardinality constraints on role successors. In: Dixon, C., Finger, M. (eds.) FroCoS 2017. LNCS (LNAI), vol. 10483, pp. 43–59. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66167-4_3
2. Baader, F., Hladik, J., Peñaloza, R.: Automata can show PSPACE results for description logics. *Inf. Comput.* **206**, 1045–1056 (2008)
3. Baader, F., Horrocks, I., Lutz, C., Sattler, U.: An Introduction to Description Logic. Cambridge University Press, Cambridge (2017)
4. Bednarczyk, B.: One-variable logic meets Presburger arithmetic. *Theor. Comput. Sci.* **802**, 141–146 (2020)

5. Bednarczyk, B., Orlowska, M., Pacanowska, A., Tan, T.: On classical decidable logics extended with percentage quantifiers and arithmetics. In: FSTTCS 2021 (2021)
6. Boiret, A., Hugot, V., Niehren, J., Treinen, R.: Automata for unordered trees. *Inf. Comput.* **253**, 304–335 (2017)
7. Boiret, A., Hugot, V., Niehren, J., Treinen, R.: Logics for unordered trees with data constraints. *J. Comput. Syst. Sci.* **104**, 149–164 (2019)
8. Bonatti, P.A., Lutz, C., Murano, A., Vardi, M.Y.: The complexity of enriched μ -calculi. *Log. Methods Comput. Sci.* (2008)
9. Boneva, I., Talbot, J.-M.: Automata and logics for unranked and unordered trees. In: Giesl, J. (ed.) RTA 2005. LNCS, vol. 3467, pp. 500–515. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-32033-3_36
10. Borosh, I., Flahive, M., Treybig, B.: Small solutions of linear Diophantine equations. *Discret. Math* **58**, 215–220 (1986)
11. Calvanese, D., Carbotta, D., Ortiz, M.: A practical automata-based technique for reasoning in expressive description logics. In: Walsh, T. (ed.) IJCAI 2011 (2011)
12. Calvanese, D., Eiter, T., Ortiz, M.: Answering regular path queries in expressive description logics via alternating tree-automata. *Inf. Comput.* **237**, 12–55 (2014)
13. Chandra, A.K., Stockmeyer, L.J.: Alternation. In: FOCS 1976 (1976)
14. Comon, H., et al.: Tree automata techniques and applications (2008)
15. Demri, S., Lugiez, D.: Complexity of modal logics with Presburger constraints. *J. Appl. Log.* **8**, 233–252 (2010)
16. Eisenbrand, F., Shmonin, G.: Carathéodory bounds for integer cones. *Oper. Res. Lett.* **34**, 564–568 (2006)
17. El-Sappagh, S., Franda, F., Ali, F., Kwak, K.S.: SNOMED CT standard ontology based on the ontology for general medical science. *BMC Med. Inform. Decis. Making* **18**, 1–19 (2018)
18. Grädel, E.: Description logics and guarded fragments of first order logic. In: DL (1998)
19. Kupke, C., Pattinson, D.: On modal logics of linear inequalities. In: Beklemishev, L.D., Goranko, V., Shehtman, V.B. (eds.) AIML 2010 (2010)
20. Libkin, L.: Elements of Finite Model Theory. Springer, Heidelberg (2004)
21. Papadimitriou, C.H.: On the complexity of integer programming. *J. ACM (JACM)* **28**, 765–768 (1981)
22. Schwentick, T.: Trees, automata and XML. In: Beeri, C., Deutsch, A. (eds.) PODS 2004 (2004)
23. Seidl, H., Schwentick, T., Muscholl, A.: Counting in trees. In: Flum, J., Grädel, E., Wilke, T. (eds.) Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas] (2008)
24. Sipser, M.: Introduction to the theory of computation. *ACM Sigact News* **27**, 27–29 (1996)



Abstract Cyclic Proofs

Bahareh Afshari^{1,2} and Dominik Wehr²(✉)

¹ Institute for Logic, Language and Computation, University of Amsterdam,
Amsterdam, The Netherlands

² Department of Philosophy, Linguistics and Theory of Science,
University of Gothenburg, Gothenburg, Sweden
`dominik.wehr@gu.se`

Abstract. Cyclic proof systems permit derivations that are finite graphs in contrast to conventional derivation trees. The soundness of such proofs is ensured by a condition on the paths through the derivation graph, known as the *global trace condition*. To give a uniform treatment of such cyclic proof systems, Brotherston proposed an abstract notion of trace. We extend Brotherston’s approach into a category theoretical rendition of cyclic derivations, advancing the framework in two ways: First, we introduce *activation algebras* which allow for a more natural formalisation of trace conditions in extant cyclic proof systems. Second, accounting for the composition of trace information allows us to derive novel results about cyclic proofs, such as introducing the Ramsey trace condition.

Keywords: Cyclic Proof Theory · Category Theory · Mu-Calculus

1 Introduction

In a cyclic proof system, proofs are finite graphs which represent the non-wellfounded derivations obtained by unravelling them. Broadly speaking, the logics benefiting from cyclic proofs often feature notions of (co-)induction or fixed-points (e.g. [5, 10, 12, 15]). Differing from conventional proof systems, cyclic systems typically eschew explicit induction axioms which can instead be simulated by cyclic derivations. This means cut-free cyclic proof systems lend themselves well to proof theoretic investigations (see e.g. [1, 2, 14]) and proof search procedures (e.g. [6, 7, 21]).

Since cyclic proofs contain infinite paths, their soundness can not be reduced to the local soundness of derivation rules. Instead, one usually must impose further soundness conditions on these paths. The most common such condition is known as the *global trace condition*. While the concrete formulation of the trace condition differs between logics, it often adheres to a certain form: A path satisfies the trace condition if it has a suffix along which some parameter (such as a

This work was supported by the Knut and Alice Wallenberg Foundation [2020.0199, 2015.0179]; and the Swedish Research Council [2017-05111, 2016-03502].

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
A. Ciabattini et al. (Eds.): WoLLIC 2022, LNCS 13468, pp. 309–325, 2022.
https://doi.org/10.1007/978-3-031-15298-6_20

term or fixed-point quantifier) can be traced and which *progresses* (e.g. decreases or is unfolded) infinitely often.

Motivated by this structural similarity, Brotherston [5] developed an abstract framework to uniformly represent and reason about cyclic proof systems. The formalism was used to give a general proof of the decidability of proof checking for cyclic proofs and to study various transformations of cyclic derivations which maintain the global trace condition. While Brotherston’s notion of trace condition encompasses most cyclic systems in the literature, it cannot naturally capture the trace condition of the modal μ -calculus, which is central to cyclic proof theory.

This article introduces an abstract representation of cyclic proof systems which expands Brotherston’s approach. Its base is formed by an abstract, category theoretical rendition of cyclic derivations and their trace conditions. As an instance of this abstract rendition, we propose an extension of Brotherston’s abstract trace condition which replaces his notion of single progress points with a more nuanced, algebraic notion of progress. Notably, this extension can capture the trace condition of the modal μ -calculus in a natural manner. Furthermore, by employing a categorical formalism which considers the composition of trace information, we obtain two novel results: derivation compression and an alternative soundness condition. Derivation compression allows a cyclic derivation with n cyclic edges to be represented as a graph of size $O(n)$, which we believe should yield performance-gains in implementations of, for example, cyclic proof checking algorithms. Finally, we give a condition on derivations using Ramsey’s theorem that is equivalent to the global trace condition and induces a proof checking algorithm which does not rely on automata theory. While a similar condition has been known in the field of program termination checking [13], this is, to the best of our knowledge, the first time such a condition has been considered for cyclic proofs.

Overview. In Sect. 2 we outline a cyclic proof system for the modal μ -calculus which serves as an example of a cyclic proof system throughout the rest of the paper. Section 3 presents our abstract notion of trace condition which is then used in Sect. 4 to introduce an abstract notion of cyclic derivation. In Sect. 5 we give a soundness condition, equivalent to the global trace condition, inspired by a result from program termination checking. In Sect. 6 we sketch a proof of the PSPACE-completeness of checking the global trace condition of abstract cyclic proofs. We close with a discussion of related and future work in Sect. 7.

2 Cyclic Proofs for the Modal μ -Calculus

We begin by laying out a cyclic proof system for the modal μ -calculus. It will serve as an example motivating the abstract definitions of Sects. 3 and 4. This presentation of the system is taken from Afshari and Leigh [1] and is an adaptation of the tableaux of Niwiński and Walukiewicz [15]. The choice of logic for this example is secondary, the main focus being the cyclic aspects of the proof system.

$$\begin{array}{ccc}
\text{Ax} \frac{}{p, \neg p} & \text{Wk} \frac{\Gamma}{\Gamma, \varphi} & \vee \frac{\Gamma, \varphi, \psi}{\Gamma, \varphi \vee \psi} \quad \wedge \frac{\Gamma, \varphi \quad \Gamma, \psi}{\Gamma, \varphi \wedge \psi} \\
\text{Mod} \frac{\Gamma, \varphi}{\Diamond \Gamma, \Box \varphi} & \mu \frac{\Gamma, \varphi[\mu x. \varphi/x]}{\Gamma, \mu x. \varphi} & \nu \frac{\Gamma, \varphi[\nu x. \varphi/x]}{\Gamma, \nu x. \varphi}
\end{array}$$

Fig. 1. Derivation rules of the modal μ -calculus

For a set PROP of propositional letters a countable set VAR of variables, the μ -formulas are given by the following grammar:

$$\varphi \in \text{FORM} ::= p \mid \neg p \mid x \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \Box \varphi \mid \Diamond \varphi \mid \mu x. \varphi \mid \nu x. \varphi \quad p \in \text{PROP}, x \in \text{VAR}$$

If $x, y \in \text{VAR}$ occur in φ , we say x *subsumes* y , writing $x <_{\varphi} y$, if $\sigma y. \psi$ occurs as a subformula of φ for some $\sigma \in \{\mu, \nu\}$ and ψ , and furthermore x is free in $\sigma y. \psi$. If the relation $<_{\varphi}$ is a strict preorder, we call φ *well-named*. In the remainder of this article, we assume all μ -formulas are well-named. This is unproblematic because any μ -formula is α -equivalent to a well-named one. Any μ -formula is positive in all variables. The fixed-point expressions $\mu x. \varphi$ and $\nu x. \varphi$ denote, respectively, the least and greatest fixed-point of the semantic counterpart to the function $x \mapsto \varphi(x)$, which is well-defined by the previous observation. The semantics of the modalities and connectives are as in the modal logic K.

A *tree* on a set X is a finite, non-empty set of X -sequences $T \subseteq X^*$ which is prefix closed, i.e. if $tx \in T$ for $t \in X^*$ and $x \in X$ then $t \in T$ as well. We call $t \in T$ a *node* of T and any $tx \in T$ a *child* of t . A node $t \in T$ is a *leaf* of T if it has no children. A *cyclic tree* is a pair $C = (T, \beta)$ consisting of a tree T on some set X and a partial function $\beta: \text{Leaf}(T) \rightarrow T$ mapping some leaves of T to nodes of T with $\beta(s) \notin \text{dom}(\beta)$ for all $s \in \text{dom}(\beta)$. Each $s \in \text{dom}(\beta)$ is called a *bud* and $\beta(s)$ its *companion*.

A μ -*pre-proof* is a pair (C, λ, ρ) of a cyclic tree C and a function $\lambda: T \rightarrow \text{SEQ}$ labelling the nodes of C with sequents, finite sets of μ -formulas. The function $\rho: \text{Inner}(T) \rightarrow \mathcal{R}$ labels each inner node with a rule from Fig. 1 such that for any $t \in \text{Inner}(T)$ the conclusion of $\rho(t)$ is $\lambda(t)$, the children of t analogously corresponding to the premises of $\rho(t)$. We use the notations $\Gamma, \varphi := \Gamma \cup \{\varphi\}$ and $\Diamond \Gamma := \{\Diamond \varphi \mid \varphi \in \Gamma\}$. Any $t \in \text{Leaf}(T)$ of a μ -pre-proof which is not labelled by an axiomatic sequent needs to satisfy $t \in \text{dom}(\beta)$ and $\lambda(t) = \lambda(\beta(t))$.

Not all pre-proofs “prove” valid sequents. For example, $\mu x. \Box x$ is invalid but is concluded by the μ -pre-proof given in Fig. 2 (the dashed arrow representing β). It is thus necessary to impose an additional soundness condition which delineates μ -proofs from μ -pre-proofs.

A *branch* through a cyclic μ -pre-proof $((T, \beta), \lambda, \rho)$ is an infinite sequence $t \in T^{\omega}$ such that $t_0 = \varepsilon$ and for any t_i , either (a) t_{i+1} is a child of t_i or (b) $t_i \in \text{dom}(\beta)$ and $t_{i+1} = \beta(t_i)$. This induces the sequence $\Gamma \in \text{SEQ}^{\omega}$ with $\Gamma_i := \lambda(t_i)$ and the partially defined $R: \omega \rightarrow \mathcal{R}$ with $R_i := \rho(t_i)$, which we use interchangeably with $t \in T^{\omega}$ to denote a branch. Along a branch $(\Gamma_i)_{i \in \omega}$, we call $\varphi' \in \Gamma_{i+1}$ a *precursor* of $\varphi \in \Gamma_i$, writing $\varphi' \leftarrow_i \varphi$, if Γ_{i+1} was chosen according

$$\text{MOD} \frac{\mu x. \Box x}{\Box \mu x. \Box x} \quad \mu \frac{\Box \mu x. \Box x}{\mu x. \Box x} \leftarrow$$

Fig. 2. A μ -pre-proof of an invalid μ -formula

to (a) and either φ is principal in R_i , i.e. φ is “altered by R_i ”, and φ' is one of the formula occurrences resulting from φ via R_i or if φ is not principal in R_i and $\varphi = \varphi'$. If Γ_{i+1} was chosen according to (b) and $\varphi = \varphi'$ then $\varphi' \leftarrow_i \varphi$ as well. A sequence of formulas $(\varphi_i)_{i \in \omega}$ is called a *trace* along $(\Gamma_i)_{i \in \omega}$ if $\varphi_{i+1} \leftarrow_i \varphi_i$ for all $i \in \omega$. It is easily observed that the subsumption order is preserved along traces, i.e. $<_{\varphi_{i+1}} \subseteq <_{\varphi_i}$ whenever $\varphi_{i+1} \leftarrow_i \varphi_i$, which is why we from now on associate to each trace $(\varphi_i)_{i \in \omega}$ a *global subsumption order* $<_{\varphi} := <_{\varphi_0} = \bigcup_{i \in \omega} <_{\varphi_i}$.

Let $(\Gamma_i)_{i \in \omega}$ be a branch through a cyclic μ -pre-proof and $(\varphi_i)_{i \in \omega}$ be a trace along it. The trace is called a ν -trace if there exists an $x \in \text{VAR}(\varphi_0)$ such that $\varphi_i = \nu x. \psi$ and $\varphi_{i+1} = \psi[\varphi_i/x]$ for infinitely many $i \in \omega$, and furthermore for any $y \in \text{VAR}(\varphi_0)$ if there are infinitely many $\varphi_i = \mu y. \theta$ then $x <_{\varphi} y$.

Definition 1 (μ -proof). A μ -proof is a μ -pre-proof which satisfies the global trace condition, i.e. if every infinite branch through it has a ν -trace.

For example, as the unique branch through the pre-proof in Fig. 2 clearly does not have a ν -trace, it is no proof.

3 Abstracting the Trace Condition

In this section, we demonstrate our formalism for capturing the trace conditions of cyclic proof systems. It encompasses two levels of abstraction: The notion of a *trace category* which captures what it means to be a trace condition, and a family of concrete trace categories, generated by the notion of an *activation algebra*.

An abstract notion of trace condition requires an abstract notion of branches through a proof. Working in a category theoretical framework, we observe that branches are infinite sequences of rule applications, and identify them with infinite sequences of morphisms called *paths*. A trace condition is then a condition on such paths which is invariant under certain path transformations.

A *semi-category* is a category which may not have identity morphisms. That is, a semi-category $\mathcal{C}$ consists of a collection $\text{Ob}(\mathcal{C})$ of objects and collections $\text{Hom}_{\mathcal{C}}(X, Y)$ of morphisms between each pair $X, Y \in \text{Ob}(\mathcal{C})$ of objects. There is a composition $\circ : \prod X, Y, Z \in \text{Ob}(\mathcal{C}). \text{Hom}_{\mathcal{C}}(Y, Z) \times \text{Hom}_{\mathcal{C}}(X, Y) \rightarrow \text{Hom}_{\mathcal{C}}(X, Z)$ which is associative. A *semi-functor* is a semi-category homomorphism. That is, for semi-categories $\mathcal{C}, \mathcal{D}$, a semi-functor $F : \mathcal{C} \rightarrow \mathcal{D}$ consists of a map on objects $F_0 : \text{Ob}(\mathcal{C}) \rightarrow \text{Ob}(\mathcal{D})$ and furthermore a map F_2 on all morphisms of $\mathcal{C}$ such that for $f : X \rightarrow Y$ we have $F_1(f) : F_0(X) \rightarrow F_0(Y)$. F_1 distributes over the

composition operation, i.e. $F_1(g \circ f) = F_1(g) \circ F_1(f)$. As is also common for standard functors, we denote both F_0 and F_1 by F .

We write ω to denote both the preorder category on $(\omega, \leq)$ and the strict preorder semi-category on $(\omega, <)$. The choice between the two will always be clear from the context. We denote the non-identity morphisms of each variant of ω by $i < j$ where $i, j \in \omega$.

A *path* through a category $\mathcal{T}$ is a functor $P: \omega \rightarrow \mathcal{T}$. Given $P, P': \omega \rightarrow \mathcal{T}$ we call P a subpath of P' , written $P \subseteq P'$, if there is a semi-functor $S: \omega \rightarrow \omega$ such that $P = P' \circ S$. The transitive, symmetric closure of $\subseteq$ is denoted $\sim$. This means $P \subseteq P'$ holds if P' can be transformed into the P by applying a combination of two path transformations: (1) Discarding a finite prefix of path P' , for example by taking $S(m) := m + n$. (2) Composing morphisms along P' . For example, if P' is of the form

$$X_0^0 \xrightarrow{\tau_0^0} X_0^1 \xrightarrow{\tau_0^1} \dots X_0^{n_0} \xrightarrow{\tau_0^{n_0}} X_1^0 \xrightarrow{\tau_1^0} \dots X_1^{n_1} \xrightarrow{\tau_1^{n_1}} X_2^0 \xrightarrow{\tau_2^0} \dots X_2^{n_2} \xrightarrow{\tau_2^{n_2}} \dots$$

then $P \subseteq P'$ for the path P below

$$X_0^0 \xrightarrow{\tau_0^{n_0} \circ \dots \circ \tau_0^0} X_1^0 \xrightarrow{\tau_1^{n_1} \circ \dots \circ \tau_1^0} X_2^0 \xrightarrow{\tau_2^{n_2} \circ \dots \circ \tau_2^0} \dots$$

witnessed by $S(i) := \sum_{j < i} (n_j + 1)$.

Definition 2. A trace category is a category $\mathcal{T}$ equipped with a trace condition, a predicate on paths, fulfilling the property: For any two paths $P \sim P'$ the trace condition holds for P if and only if it holds for P' .

This means any trace condition is invariant under taking suffixes and composition. All concrete trace conditions in the literature are closed under taking suffixes, making closure under suffixes a natural criterion for identifying trace conditions. Composition of rules along branches is not part of cyclic proof systems and hence there is a priori no precedent for closure under compositions. However, the cumulative nature of trace conditions suggests that composition of traces should not invalidate them either. Indeed, all trace categories which we use to model concrete trace conditions from the literature possess a trace condition closed under composition. Furthermore, it is precisely this closure condition that has proven instrumental in deriving new results in our framework.

The remainder of this section is concerned with defining a family of concrete trace categories which can model the trace conditions of all cyclic proof systems we know of.

Definition 3. An activation algebra is a tuple $\mathcal{A} = (A, \leq, \vee, 0, \alpha)$ a semilattice $(A, \leq, \vee, 0)$ and an activation element $\alpha \in A$ where $0 \neq \alpha$.

Definition 4. Let $\mathcal{A}$ be an activation algebra. The $\mathcal{A}$ -activated trace category $\mathcal{T}_{\mathcal{A}}$ has as its objects the finite sets. The morphisms between sets X, Y are all relations $R \subseteq X \times \mathcal{A} \times Y$. The identities are $1_X := \{(x, 0, x) \mid x \in X\}$. We often

write xR^ay to mean $(x, a, y) \in R$. Given morphisms $R: X \rightarrow Y, R': Y \rightarrow Z$ their composition is specified by $(x, c, z) \in R' \circ R$ iff

$$\exists y \in Y \exists a, b \in A. (x, a, y) \in R \text{ and } (y, b, z) \in R' \text{ and } a \vee b = c$$

A path $P: \omega \rightarrow \mathcal{T}_{\mathcal{A}}$ satisfies the trace condition if there exists a subpath $P' \subseteq P$ and a sequence $\sigma: \Pi i \in \omega. P'(i)$ along it such that $\sigma_i P'(i < i + 1)^\alpha \sigma_{i+1}$ for all $i \in \omega$.

Proposition 1. *The trace condition in Definition 4 is invariant under $\sim$.*

Proof. It suffices to prove invariance under $\subseteq$. Let $P \subseteq Q$ with $P = Q \circ S$. Suppose P satisfies the trace condition meaning there exists $P' \subseteq P$ with $P' = P \circ S'$ and a validating sequence σ . Then $P' \subseteq Q$ via $P' = Q \circ S \circ S'$, meaning σ is a validating sequence through a subpath of Q as well.

For the converse direction, let $P = Q \circ S$ and suppose Q satisfies the trace condition as witnessed by $Q' = Q \circ S'$ and a sequence $\sigma: \Pi i \in \omega. Q'(i)$. It remains to show that there is $P' \subseteq P$ and a suitable sequence $\sigma'': \Pi i \in \omega. P'(i)$ along it. By fixing $b := S'(0)$ and analysing Definition 4, one can conclude there are two sequences $\sigma': \Pi i \in \omega. Q(b+i)$ and $a: \omega \rightarrow \mathcal{A}$ such that $\sigma'_i Q(b+i < b+i+1)^{a_i} \sigma'_{i+1}$ with $a_i \leq \alpha$. For $i_j := S'(j) - b$ we then have $\sigma'_{i_j} = \sigma_j$ and $\bigvee_{i=i_j}^{i < i_{j+1}} a_i = \alpha$. Now construct the following sequence

$$k_0 := \text{least } i \in \omega \text{ such that } S(i) \geq b$$

$$k_{n+1} := \text{least } i > k_n \text{ s.t. } S(k_n) \leq S'(j) < S'(j+1) \leq S(i) \text{ for some } j \in \omega$$

We claim setting $S''(i) := S(k_i)$ induces the desired subpath $P' := Q \circ S''P$ with $P'(i) = P(k_i)$ as witnessed by $\sigma'': \Pi i \in \omega. P'(i)$ given by $\sigma''_i := \sigma'_{S(k_i)}$. For this, we need to check that $\sigma''_{i+1} P'(i < i + 1)^\alpha \sigma''_{i+1}$. Let $j \in \omega$ be such that $S''(i) \leq S'(j) < S'(j+1) \leq S''(i+1)$ then $P'(i < i + 1) = Q(b + i_{j+1} \leq S(k_{i+1})) \circ Q(b + i_j < b + i_{j+1}) \circ Q(S(k_i) \leq b + i_j)$ meaning

$$\left(\sigma''_i, \underbrace{\bigvee_{l=b+i_{j+1}}^{l < S(k_{i+1})} a_{l-b}}_{\leq \alpha} \vee \underbrace{\bigvee_{l=i_j}^{l < i_{j+1}} a_l}_{=\alpha} \vee \underbrace{\bigvee_{l=S(k_i)}^{l < b+i_j} a_{l-b}}_{\leq \alpha}, \underbrace{\sigma'_{k_{i+1}-b}}_{=\sigma''_{i+1}} \right) \in P'(i < i + 1)$$

and thus $(\sigma''_i, \alpha, \sigma''_{i+1}) \in P'(i < i + 1)$ as desired.

We close this section with two examples of activation algebras and an illustration of their use to formalise trace conditions.

Example 1. The simplest example of an activation algebra is given by the simplest semilattice of at least two elements: The binary Boolean algebra $\mathbb{B} = \{\top, \perp\}$. Here the choice $\alpha := \top$ is forced as necessarily $\perp = 0 \neq \alpha$. This activation algebra is implicit in Brotherston's abstract notion of trace [5] and suffices to model most trace conditions in the literature, for example [5, 12, 18, 19].

Example 2. The activation algebra used to formalise the trace condition of the modal μ -calculus is the three value *failure algebra* $\mathbb{F} := (\{0, 1, 2\}, \leq, \vee, 0, 1)$. In this algebra, the value 2 can be used to represent a sort of “failure state”, as illustrated in Definition 5.

Definition 5. Let $(\Gamma_i)_{i \in \omega}$ be a branch through a μ -pre-proof, then $\hat{\Gamma}: \omega \rightarrow \mathcal{T}_{\mathbb{F}}$ is defined as below. If $t_i \in \text{dom}(\beta)$ and $t_{i+1} = \beta(t_i)$ then R_i is undefined, in which case the third case of the definition of $\hat{\Gamma}(i < i + 1)$ applies.

$$\begin{aligned}\hat{\Gamma}(i) &:= \{(\varphi, x) \mid \varphi \in \lambda(t_i) \text{ and } x \text{ is bound by } \nu \text{ in } \varphi\} \\ \hat{\Gamma}(i < i + 1) &:= \{((\varphi, x), a^*, (\varphi', x)) \mid \varphi' \leftarrow \varphi\} \\ \text{where } a^* &:= \begin{cases} 2, & \text{if } R_i = \mu, \varphi = \mu y. \theta, \varphi' = \theta[\mu y. \theta / y] \text{ and } y <_{\varphi} x, \\ 1, & \text{if } R_i = \nu, \varphi = \nu x. \theta, \varphi' = \theta[\nu x. \theta / x], \\ 0, & \text{otherwise.} \end{cases}\end{aligned}$$

As $\hat{\Gamma}(i < i + n) = \hat{\Gamma}(i + n - 1 < i + n) \circ \dots \circ \hat{\Gamma}(i < i + 1)$, this fully specifies $\hat{\Gamma}$.

Proposition 2. A branch $(\Gamma_i)_{i \in \omega}$ through a μ -pre-proof has a ν -trace if and only if $\hat{\Gamma}: \omega \rightarrow \mathcal{T}_{\mathbb{F}}$ satisfies the trace condition.

Proof. Suppose $(\Gamma_i)_{i \in \omega}$ has a ν -trace $(\varphi_i)_{i \in \omega}$. Then there exists $x \in \bigcap_{i \in \omega} \text{VAR}(\varphi_i)$ bound by ν -quantifier and an increasing sequence $(j_i)_{i \in \omega}$ such that

- (i) $\varphi_{j_i} = \nu x. \psi$ and $\varphi_{j_{i+1}} = \psi[\varphi_{j_i} / x]$, and
- (ii) no formula $\mu y. \theta$ with $y <_{\varphi} x$ is unfolded along $(\varphi_i)_{i > j_0}$.

The subpath $P := \hat{\Gamma} \circ S$ induced by $S(i) := j_i$ and the sequence $\sigma_i := (\varphi_{j_i}, x)$ witness that $\hat{\Gamma}$ satisfies the trace condition: Clearly always $(\sigma_i, a, \sigma_{i+1}) \in \hat{\Gamma}(j_i, j_{i+1})$ for some $a \in \mathbb{F}$. We know $1 \leq a$ because between Γ_{j_i} and $\Gamma_{j_{i+1}}$, $\nu x. \psi$ is unfolded, and further $a < 2$, as no $\mu y. \theta$ with $y <_{\varphi} x$ is unfolded after j_0 , meaning $a = 1$ as desired.

Conversely, suppose $\hat{\Gamma}$ satisfied the trace condition. Then there exist $S: \omega \rightarrow \omega$ and $\sigma: \Pi i \in \omega. \hat{\Gamma}(S(i))$ such that $(\sigma_i, 1, \sigma_{i+1}) \in \hat{\Gamma}(S(i) < i + 1)$ for every $i \in \omega$. It is easily observed that necessarily $\sigma_i = (\varphi_i, x)$ for some fixed ν -variable x and $\varphi_i \in \Gamma_{S(i)}$. Furthermore, because the activation algebra element along that trace is precisely 1 we can conclude that between $\Gamma_{S(i)}$ and $\Gamma_{S(i+1)}$

- (i) the formula $\nu x. \psi$ corresponding to x in φ_i is unfolded (as $1 \leq \alpha$),
- (ii) no $\mu y. \theta$ with $y <_{\varphi} x$ is unfolded (as $\alpha < 2$).

By scrutinising the derivation rules, it can be observed that $\varphi_{S(0)}$ can be “traced back” to some $\varphi \in \Gamma_0$ and subsequently completed into a trace $(\varphi'_i)_{i \in \omega}$ along $(\Gamma_i)_{i \in \omega}$ with $\varphi'_{S(i)} = \varphi_i$. By the observations above, this must be a ν -trace.

The reader may consult [22] for further examples of the trace conditions of cyclic proof systems being modeled in $\mathcal{T}_{\mathcal{A}}$, namely those of cyclic arithmetic [19], $\text{HFL}_{\mathbb{N}}$ [12] and Grzegorzcyk modal logic [18].

4 Abstract Cyclic Derivations

Having defined abstract notions of branches, traces and the trace condition, we combine them into an abstract account of cyclic derivations. Such abstract cyclic derivations are pairs (C, Tr) of cyclic trees C and maps Tr which “decorate” the edges of C with maps of a trace category. Such an abstract cyclic derivation then is considered a proof if all paths which can be generated by traversing C satisfy the trace condition imposed by the trace category. To express these ideas in a category theoretical manner, we begin by giving a categorical representation of cyclic trees, by defining the (semi-)category induced by the paths through said trees.

Fix a cyclic tree $C = (T, \beta)$. The finite paths from s to t , denoted by $\text{Path}_C(s, t) \subseteq T^+$, are defined as the smallest sets satisfying the following three conditions: (i) For any $s \in T$ we have $s \in \text{Path}_C(s, s)$, (ii) for any $t, u \in T$ with u child of t , whenever $p \in \text{Path}_C(s, t)$ we also have $pu \in \text{Path}_C(s, u)$, and (iii) for any $t \in \text{dom}(\beta)$, whenever $p \in \text{Path}_C(s, t)$ we also have $p\beta(t) \in \text{Path}_C(s, \beta(t))$. The category $\mathcal{P}_C$ of paths through $C = (T, \beta)$ has T as its objects and fixes $\text{Hom}_{\mathcal{P}_C}(s, t) = \text{Path}_C(s, t)$. The identities are $1_s = s : s \rightarrow s$ and, given morphisms $p : s \rightarrow t$ and $q : t \rightarrow u$, we define $q \circ p = pq'$ where $q = tq'$ for $q' \in T^*$. The semi-category $\mathcal{P}_C^S$ of progressing paths is the same as $\mathcal{P}_C$ except that $\text{Hom}_{\mathcal{P}_C^S}(s, t) := \{p \in \text{Path}_C(s, t) \mid |p| > 1\}$.

The informal notion of “decorating a cyclic tree with trace information” can now be expressed as a functor.

Definition 6. *An abstract cyclic derivation (ACD) in a trace category $\mathcal{T}$ is a pair (C, Tr) consisting of a cyclic tree $C = (T, \beta)$ and a functor $\text{Tr} : \mathcal{P}_C \rightarrow \mathcal{T}$ such that for any $s \in \text{dom}(\beta)$, $\text{Tr}(s) = \text{Tr}(\beta(s))$ and $\text{Tr}(s\beta(s)) = 1_{\text{Tr}(s)}$.*

We delineate the abstract cyclic proofs from mere abstract cyclic derivations via a global trace conditions, similar to the cyclic proof system of the modal μ -calculus.

Definition 7. *Let D be an ACD given by $(C, \text{Tr} : \mathcal{P}_C \rightarrow \mathcal{T})$. A path through D is a path $P : \omega \rightarrow \mathcal{T}$ such that there exists a semi-functor $P' : \omega \rightarrow \mathcal{P}_C^S$ with $P = \text{Tr} \circ P'$. D satisfies the global trace condition (GTC) if every path through D satisfies the trace condition of $\mathcal{T}$. We call an ACD satisfying the GTC an abstract cyclic proof.*

Requiring P' to be a semi-functor in the definition above rules out constant “paths” obtained via $P(i) := \text{Tr}(s)$, $P(i < i+1) := 1_{\text{Tr}(s)}$ which are not generated by traversing the ACD along any branch of the cyclic tree.

Example 3. Given a μ -pre-proof (C, λ, ρ) , the functor $\text{Tr} : \mathcal{P}_C \rightarrow \mathcal{T}_{\mathbb{F}}$ inducing its corresponding ACD (C, Tr) is defined analogously to the path $\hat{F}$ arising from a branch F through a μ -pre-proof. Consider the μ -pre-proof corresponding to the ACD pictured in Fig. 3. For the sake of readability, we have opted to write the carrier sets of the ACD as simple sets of variables x instead of

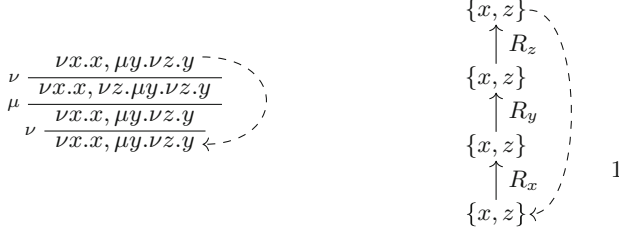


Fig. 3. A μ -proof and its corresponding ACD

sets of pairs (φ, x) because each variable occurs in only one of the formulas in the sequent. Fully specified, the first set should be $\{(\nu x.x, x), (\mu y.\nu z.y, z)\}$. The annotated morphisms are $1 := \{(x, 0, x), (z, 0, z)\}$, $R_x := \{(x, 1, x), (z, 0, z)\}$, $R_y := \{(x, 0, x), (z, 2, z)\}$ and $R_z := \{(x, 0, x), (z, 1, z)\}$.

Note that $(z, 2, z) \in R_y$ because $y <_\varphi z$ with $\varphi := \mu y.\nu z.y$. The μ -pre-proof above has only one branch Γ which has one ν -trace (that on x) and is thus a proof. Similarly, $\hat{\Gamma}$ is a path through the ACD and satisfies the trace condition with $S(i) := 5i$ and $\sigma := i \mapsto x$. In principle, verifying that the ACD is a proof would require checking infinitely many other paths P , as paths through ACDs need not start at their root and a step $P(i < i + 1)$ may correspond to a path segment $p \in \text{Path}_C(s, t)$ with $|p| > 2$. However, as all such paths are subpaths of $\hat{\Gamma}$, they satisfy the trace condition by subpath invariance. Indeed, this observation extends to arbitrary ACDs obtained by transforming concrete cyclic derivations: Any path through an ACD is always a subpath of $\hat{\Gamma}$ for some branch Γ through the corresponding concrete cyclic derivation. That means that the global trace condition in Definition 7, although seeming stricter at first, corresponds precisely to that of concrete cyclic proof systems.

We close this section by showing that, by composing trace maps, ACDs can be transformed into equivalent ACDs whose number of nodes $|T|$ is linear in their number of cyclic edges $|\beta|$. For ACDs generated from concrete cyclic derivations, as outlined in Example 3, this will usually result in a drastic reduction of $|T|$. This procedure can be viewed as a sort of “compression algorithm”. This should prove useful when implementing programs dealing with representations of cyclic proofs as many algorithms, prime among them proof checking ((Theorem 4), have $|T|$ as one of their complexity parameters. Indeed, it seems the CYCLIST automated theorem prover already relies on a similar optimisation (see [7, Section 4.2]) although we are not aware of a formal characterisation of this optimization in the literature.

Theorem 1. *Any $D = ((T, \beta), \text{Tr})$ can be transformed into $D' = ((T', \beta'), \text{Tr}')$ such that D' is a proof if and only if D is a proof and $|T'| \leq 2|\beta| + 1$.*

Proof. Fix $B := \text{im}(\beta) \cup \text{dom}(\beta) \cup \{\varepsilon \mid \nexists s \in \text{im}(\beta). \forall t \in \text{im}(\beta) \cup \text{dom}(\beta). s \leq t\}$, where $s < t$ if t is a proper prefix of s . We now construct a sequence of partial maps $(f_i : B \rightarrow \omega^*)_i$. Fix $f_0(s) := \varepsilon$ where s is the $<$ -least element of B .

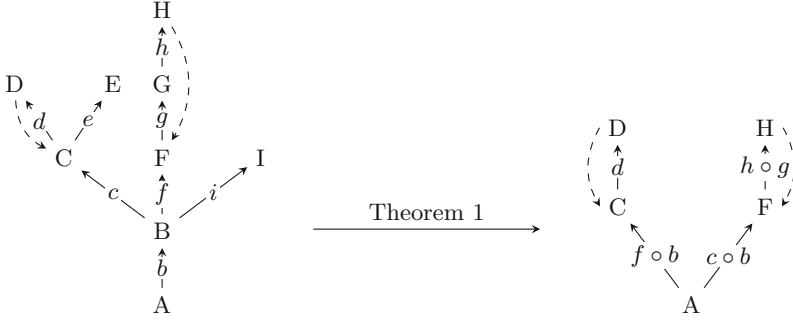


Fig. 4. Compressing an ACD via Theorem 1

To construct f_{n+1} consider each $s \in \text{dom}(f_n)$, that is, every $s \in B$ which was added by the previous construction step. For each such s , let $\{t_0, \dots, t_n\} \subseteq B$ be all $t \in B$ such that s can reach t without crossing any other node from B (formally, if there is $spt \in \text{Path}_C(s, t)$ with $p \in (T \setminus B)^*$) and set $f_{n+1}(t_i) := f_n(s)i$. Fixing $f := \bigcup_{n \leq |B|} f_n$ and $T' := \text{im}(f)$, observe that $f: B \rightarrow T'$ is a $<$ -isomorphism between B and T' . Thus $C' := (T', \beta' := f \circ \beta \circ f^{-1})$ is a cyclic tree with $\beta'(x) = y$ iff $\beta(f^{-1}(x)) = f^{-1}(y)$. Also, clearly $|T'| = |B| \leq 2|\beta| + 1$.

To extend C' to an abstract cyclic derivation D' , take $\text{Tr}'(s) := \text{Tr}(f^{-1}(s))$ for $s \in T'$. For any $s < t \in T'$, observe that per construction of T' , there exists a unique path $f^{-1}(s)p_{st}f^{-1}(t) \in \text{Path}_C(f^{-1}(s), f^{-1}(t))$ with the property that $p_{st} \in (T \setminus B)^*$. Pick $\text{Tr}'(st) := \text{Tr}(f^{-1}(s)p_{st}f^{-1}(t))$ and observe that this is both well-defined and fully specifies $\text{Tr}': \mathcal{P}_{C'} \rightarrow \mathcal{T}$. Claim 1. now follows from the following fact, which is easily verified: For any path P through D there exists a path $P' \sim P$ through D' and vice versa (Fig. 4).

5 A Ramsey-Based Soundness Condition

This section presents a soundness condition on cyclic derivations which is equivalent to the common global trace condition. It is similar to the condition put forward by Lee et al. [13] for the purpose of program termination checking based on the size change principle, a condition analogous to the trace conditions of cyclic proof systems such as cyclic arithmetic [19] and cyclic Gödel's T [9]. The correctness proof of the soundness condition relies on Ramsey's theorem [17, Theorem A], rather than the automata theoretic methods so prevalent in cyclic proof theory. We employ the notation $[A]^n := \{X \subseteq A \mid |X| = n\}$.

Theorem 2 (Ramsey's theorem). *Let A be a countable set, C finite and $n \in \omega$. For any colouring $f: [A]^n \rightarrow C$ there exists a colour $c \in C$ and a countable $B \subseteq A$ such that for any $X \in [B]^n$ we have $f(X) = c$.*

The central insight motivating the soundness condition is that the Ramsey theorem guarantees the existence of certain well-behaved subpaths of paths

through finite categories. For every $\tau: X \rightarrow X$ in a category $\mathcal{T}$, the *periodic* τ path is $\tau^\omega: \omega \rightarrow \mathcal{T}$ with by $\tau^\omega(i) := X$ and $\tau^\omega(i < i+1) := \tau$. We call $\tau: X \rightarrow X$ idempotent if $\tau = \tau \circ \tau$.

Lemma 1. *Let $P: \omega \rightarrow \mathcal{T}$ be a path and $X \in \text{Ob}(\mathcal{T})$ such that $P(i) = X$ infinitely often. If $\text{HOM}_{\mathcal{T}}(X, X)$ is finite then $\tau^\omega \subseteq P$ for some idempotent $\tau: X \rightarrow X$.*

Proof. $P(i) = X$ holding infinitely often means there exists $Q \subseteq P$ such that $Q(i) = X$ for all $i \in \omega$. Then $Q(\{i, j\}) := Q(i < j)$ induces a colouring $Q: [\omega]^2 \rightarrow \text{HOM}_{\mathcal{T}}(X, X)$ on ω . By the Ramsey theorem, there exists $\tau \in \text{HOM}_{\mathcal{T}}(X, X)$ and an infinite $M \subseteq \omega$ such that for $i < j \in M$ we have $Q(i < j) = \tau$. Then $\tau^\omega = Q \circ S$ where $S(i) :=$ the i th least $m \in M$. The idempotence of τ follows as $\tau = Q(S(0 < 2)) = Q(S(1 < 2)) \circ Q(S(0 < 1)) = \tau \circ \tau$.

Similarly to Definition 7, we limit our attention to the image of the trace functor $\hat{\text{Tr}}: \mathcal{P}_C^S \rightarrow \mathcal{T}$ restricted to the category of progressing paths. Fixing some ACD (C, Tr) we thus write $\overline{\text{HOM}}(s, t)$ for $\text{HOM}_{\text{im}(\hat{\text{Tr}})}(\hat{\text{Tr}}(s), \hat{\text{Tr}}(t))$, the set of all trace maps that can be generated by walking along some path $p \in \text{Path}_C(s, t)$ with $|p| > 1$.

Definition 8. *Let (C, Tr) be an ACD such that for every $u \in \text{dom}(\beta)$, $\overline{\text{HOM}}(u, u)$ is finite. Then it satisfies the Ramsey trace condition if for every $u \in \text{dom}(\beta)$ and every idempotent $\tau \in \overline{\text{HOM}}(u, u)$, the path $\tau^\omega: \omega \rightarrow \mathcal{T}$ satisfies the trace condition.*

Theorem 3. *Let (C, Tr) be an ACD such that for every $u \in \text{dom}(\beta)$, $\overline{\text{HOM}}(u, u)$ is finite. Then it satisfies the global trace condition if and only if it satisfies the Ramsey trace condition.*

Proof. First, suppose D satisfies the GTC. Then pick $u \in \text{dom}(\beta)$ and $\tau \in \overline{\text{HOM}}(u, u)$. As τ is in the image of $\hat{\text{Tr}}$, we know that there exists some $p \in \text{Path}_C(u, u)$ with $|p| > 1$ and $\text{Tr}(p) = \tau$. Then $\tau^\omega = \text{Tr} \circ p^\omega$, meaning τ^ω is a path through D and thus satisfies the trace condition.

Conversely, if D satisfies the Ramsey trace condition and pick some path $\text{Tr} \circ P$ for $P: \omega \rightarrow \mathcal{P}_C^S$. Clearly, there exists $P' \subseteq P$ with $|P'(i < i+1)| = 2$ for all $i \in \omega$, i.e. a representation of P which does not “skip” any nodes. As P' describes an infinite path through C , there needs to be some $u \in \text{dom}(\beta)$ with $P'(i) = u$ for infinitely many $i \in \omega$ and thus by Lemma 1 an idempotent $\tau: \text{Tr}(u) \rightarrow \text{Tr}(u)$ such that $\tau^\omega \subseteq \text{Tr} \circ P' \supseteq \text{Tr} \circ P$. By the Ramsey trace condition, τ^ω satisfies the trace condition, meaning $\text{Tr} \circ P$ does so as well.

By restricting our attention to $\text{HOM}_{\text{Tr}^s}(u, u)$, we guarantee that $1_{\text{Tr}(u)}$ can only occur in $\text{HOM}_{\text{Tr}^s}(u, u)$ if there is some $p \in \text{Path}_C(u, u)$ such that $\text{Tr}(p) = 1_{\text{Tr}(u)}$. This is important as in most trace categories—including all trace categories defined in this article—the path $1_{\text{Tr}(u)}^\omega$ does not satisfy the trace condition. Naïvely including $1_{\text{Tr}(u)}$ in the collection of idempotent morphisms to consider

when checking for GTC satisfaction would thus invalidate the condition given above.

It should be noted that this soundness condition can only be stated in category theoretical presentations of the trace condition like ours. This is because it relies on the composition of trace maps to present paths in such a way that the Ramsey theorem can be applied. The Ramsey trace condition gives rise to a novel algorithm for checking whether an ACD is a proof. This algorithm is analogous to that given for termination checking by Lee et al. [13].

Theorem 4. *Let $\mathcal{T}$ be such that (1) from $\tau: X \rightarrow Y$ and $\tau': Y \rightarrow Z$ one can compute $\tau' \circ \tau: X \rightarrow Z$ and (2) for $\tau, \tau': X \rightarrow Y$ one can decide whether $\tau = \tau'$ and (3) for idempotent $\tau: X \rightarrow X$ one can decide whether $\tau^\omega: \omega \rightarrow \mathcal{T}$ satisfies the trace condition. Let further $D = (C, \text{Tr}: \mathcal{P}_C \rightarrow \mathcal{T})$ be an abstract cyclic derivation with Tr computable and such that for every $u, v \in T$ the set $\overline{\text{HOM}}(u, v)$ is finite. Then it is decidable whether D is a proof.*

Proof. By Theorem 3 we know that it suffices to check that for each $u \in \text{dom}(\beta)$ and for all idempotent $\tau \in \overline{\text{HOM}}(u, u)$, τ^ω satisfies the trace condition. We thus simply compute all of the $\overline{\text{HOM}}$ -sets and then check each idempotent endomorphism via procedure (3). The $\overline{\text{HOM}}$ -sets are computed by an iterative procedure with base cases

$$H^0(u, v) := \{\text{Tr}(uv) \mid \text{if } v \text{ child of } u\} \cup \{1_{\text{Tr}(u)} \mid \text{if } u \in \text{dom}(\beta) \text{ and } \beta(u) = v\}$$

and the iterative steps, using the procedure from assumption (1),

$$H^{i+1}(u, v) := H^i(u, v) \cup \{\tau \circ \tau' \mid w \in T, \tau' \in H^i(u, w), \tau \in H^i(w, v)\}.$$

We carry out the procedure until $H^i(u, v) = H^{i+1}(u, v)$ for all $u, v \in T$, i.e. until a fixed point is reached, which can be detected using procedure (2). That such a fixed point will be reached is guaranteed by the finiteness of the $\overline{\text{Hom}}(u, v)$. It is easily observed that $H(u, v) = \overline{\text{HOM}}(u, v)$. Now decide whether D satisfies the RTC by using the procedure (3) on all $\tau \in H(u, u)$ for $u \in \text{dom}(\beta)$ which satisfy $\tau = \tau \circ \tau$, which can be detected by using the procedures (1) and (2).

Corollary 1. *Let $\mathcal{A}$ be a finite activation algebra. It is decidable whether an abstract cyclic derivation $(C, \text{Tr}: \mathcal{P}_C \rightarrow \mathcal{T}_{\mathcal{A}})$ is a proof.*

Proof. Simply observe that $\mathcal{T}_{\mathcal{A}}$ satisfies the criteria above. Notably, for an idempotent $R: X \rightarrow X$ in $\mathcal{T}_{\mathcal{A}}$, one can decide whether R^ω satisfies the trace condition by checking if there exists an $x \in X$ such that $(x, \alpha, x) \in R$.

Corollary 1 can also be proven by an automata theoretic method: Given an ACD $D = (C, \text{Tr} : \mathcal{P}_C \rightarrow \mathcal{T}_{\mathcal{A}})$ with $\mathcal{A}$ finite, construct an automaton $\mathfrak{A}$ recognising all branches through D and an automaton $\mathfrak{B}$ recognising all branches through D which satisfy the trace condition. To decide whether D is a proof, check $L(\mathfrak{A}) \subseteq L(\mathfrak{B})$. See [22] for further details.

6 PSPACE-Completeness of Cyclic Proof Checking

In [13], Lee et al. also prove that the decision problem of the criterion is PSPACE-complete. This result has been transferred to the setting of cyclic proofs by Nollet et al. [16] who prove that checking the global trace condition of linear logic with least and greatest fixed-points (μ MALL) is PSPACE-complete. This result readily extends to abstract cyclic derivations. The proof proceeds analogously to Nollet et al. [16] by a reduction to BOOLE program termination. Indeed, the ACDs given in Definition 10 are obtained by representing the cyclic derivations given by Nollet et al. as ACDs, applying the compression procedure (Theorem 1) and removing some spurious elements their trace sets. We only provide a proof sketch. The curious reader can fill in the remaining details by consulting Nollet et al. [16].

Definition 9. A BOOLE program is a numbered sequence of instructions $1 : I_1; 2 : I_2; \dots; m : I_m$ composed according to the following grammar

$$I ::= X := \neg X \mid \text{IF } X \text{ THEN } \ell \text{ ELSE } \ell'$$

where the labels $\ell, \ell' \in \{0, \dots, m\}$ and the variables X stem from some stock of variable letters.

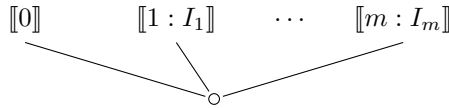
Fix a BOOLE program p of length m making use of the variables $\Xi := \{X_1, \dots, X_n\}$. Given assignments $\sigma, \sigma' : \Xi \rightarrow \mathbb{B}$ and labels $\ell, \ell' \in \{0, \dots, m\}$ we write $(\ell, \sigma) \rightsquigarrow (\ell', \sigma')$ if $\ell \neq 0$ and either:

1. the instruction labeled by ℓ in p is $X := \neg X$ for some $X \in \Xi$, $\ell' \equiv \ell + 1 \pmod{m+1}$ and $\sigma' = \sigma[X \mapsto \neg\sigma(X)]$, or
2. the instruction labeled by ℓ in p is $\text{IF } X \text{ THEN } \ell_1 \text{ ELSE } \ell_0$ for some $X \in \Xi$, $\ell' = \ell_{\sigma(X)}$ and $\sigma' = \sigma$.

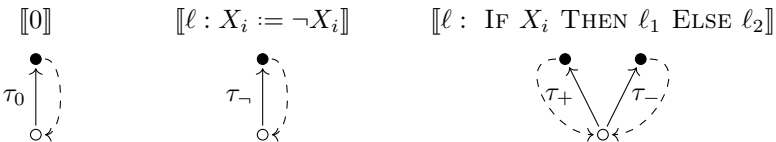
Writing ϕ for the constant assignment $x \mapsto 0$, define

$$\text{BOOLE}_{\text{false}} := \{p \text{ a BOOLE program} \mid (1, \phi) \rightsquigarrow^* (0, \phi) \text{ under } p\}.$$

Definition 10. For a program $p = 1 : I_1, \dots, m : I_m$ making use of variables $X_1, \dots, X_n$ its associated ACD $\llbracket p \rrbracket := (C, \text{Tr} : \mathcal{P}_C \rightarrow \mathcal{T}_{\mathcal{A}})$ for an arbitrary activation algebra $\mathcal{A}$ is



with $\text{Tr}(s) = \{X_1^+, X_1^-, \dots, X_n^+, X_n^-, R, G_0, \dots, G_m\}$ for every node s of C . The ‘subtrees’ $\llbracket 0 \rrbracket$ and $\llbracket \ell : I_\ell \rrbracket$ each comprise of one or two buds whose companion is the root of $\llbracket p \rrbracket$ with associated morphisms:



The morphisms are specified below. A dashed line between A and B indicates the pair $(A, 0, B)$, a bold line the pair (A, α, B) and the absence of a connecting line between A and B the absence of all pairs (A, c, B) . In τ_- , $\ell' := \ell + 1 \bmod (m+1)$.

$$\begin{array}{c}
 \tau_0 : \begin{array}{cccccccccccccccc}
 X_1^+ & X_1^- & \cdots & X_n^+ & X_n^- & R & G_0 & G_1 & \cdots & G_\ell & \cdots & G_m \\
 & \diagdown & & & \diagdown & & | & \diagup & & \diagup & & \\
 X_1^+ & X_1^- & \cdots & X_n^+ & X_n^- & R & G_0 & G_1 & \cdots & G_\ell & \cdots & G_m
 \end{array} \\
 \\
 \tau_- : \begin{array}{cccccccccccccccc}
 X_1^+ & X_1^- & \cdots & X_i^+ & X_i^- & \cdots & X_n^+ & X_n^- & R & G_0 & \cdots & G_\ell & G_{\ell'} & \cdots & G_m \\
 | & | & & \diagdown & \diagup & & | & | & & | & \diagdown & | & \diagup & & \\
 X_1^+ & X_1^- & \cdots & X_i^+ & X_i^- & \cdots & X_n^+ & X_n^- & R & G_0 & \cdots & G_\ell & G_{\ell'} & \cdots & G_m
 \end{array} \\
 \\
 \tau_+ : \begin{array}{cccccccccccccccc}
 X_1^+ & X_1^- & \cdots & X_i^+ & X_i^- & \cdots & X_n^+ & X_n^- & R & G_0 & \cdots & G_\ell & \cdots & G_{\ell_1} & \cdots & G_m \\
 | & | & & | & | & & | & | & & | & \diagdown & | & \diagup & & & \\
 X_1^+ & X_1^- & \cdots & X_i^+ & X_i^- & \cdots & X_n^+ & X_n^- & R & G_0 & \cdots & G_\ell & \cdots & G_{\ell_1} & \cdots & G_m
 \end{array} \\
 \\
 \tau_- : \begin{array}{cccccccccccccccc}
 X_1^+ & X_1^- & \cdots & X_i^+ & X_i^- & \cdots & X_n^+ & X_n^- & R & G_0 & \cdots & G_\ell & \cdots & G_{\ell_2} & \cdots & G_m \\
 | & | & & | & | & & | & | & & | & \diagdown & | & \diagup & & & \\
 X_1^+ & X_1^- & \cdots & X_i^+ & X_i^- & \cdots & X_n^+ & X_n^- & R & G_0 & \cdots & G_\ell & \cdots & G_{\ell_2} & \cdots & G_m
 \end{array}
 \end{array}$$

Lemma 2. For any given BOOLE program p , $p \in \text{BOOLE}_{\text{false}}$ iff and only if $\llbracket p \rrbracket$ does not satisfy the trace condition of $\mathcal{T}_A$.

Theorem 5. For finite $\mathcal{A}$, checking the TC on ACDs in $\mathcal{T}_A$ is PSPACE-complete.

Proof. A BOOLE-program can be transformed into the ACD given in Definition 10 in LOGSPACE. By Lemma 2, this constitutes a LOGSPACE-reduction of a PSPACE-complete problem ($\text{BOOLE}_{\text{false}}$) to cyclic proof checking in $\mathcal{T}_A$ for any finite $\mathcal{A}$.

7 Conclusion

We have defined *abstract cyclic derivations*, finite graphs whose edges are decorated with trace information, which serve as an abstract representation of the pre-proofs of cyclic proof systems. The notion of “trace information” is formalised by morphisms of *trace categories* which generalise the trace conditions of cyclic proof systems. In particular, we have defined a family of trace categories whose trace condition is specified by *activation algebras* and which capture the trace condition of most cyclic proof systems from the literature. To demonstrate this

formalism, we have shown how to represent pre-proofs of a cyclic proof system for the modal μ -calculus. We have used the categorical aspects of our formalism to derive two results: a compression procedure for abstract cyclic derivations and an alternative to the common global trace condition justified by Ramsey’s theorem. Furthermore, we have shown that in this general setting, proof checking is PSPACE-complete.

7.1 Related Work

Most definitions and results in this article stem from Wehr’s Masters thesis [22], which also contains some further results and formalisms that are not presented in this article. Notably, it formalises the trace conditions of three further cyclic proof systems in terms of trace categories: Cyclic Arithmetic [19], HFL $_{\mathbb{N}}$ [12] and Grzegorzczuk modal logic [18].

To the best of our knowledge, Brotherston’s [5] is the only previous work on abstracting cyclic proofs. Our work can be viewed as an extension of his in two ways: First, his abstract notion of derivation is closer to the common definition of pre-proofs, being presented in terms of abstract sequents and derivation rules. Crucially, this bars him from considering the composition of trace information, which enabled us to derive the ACD compression result (Theorem 1) and the alternative soundness condition (Theorem 3). Second, Brotherston only considers trace conditions expressed in terms of the booleans $\mathbb{B}$. While this is sufficient to naturally express the most conditions in the literature, it does not extend to all trace conditions, including that of the modal μ -calculus, a problem our activation algebras alleviate. It should be noted that $\mathcal{T}_{\mathbb{B}}$ is technically sufficient to express all trace conditions expressible by finite activation algebras by emulating the recognising Büchi-automaton in the trace maps (see [22, Theorem 5.30] for details). However, the resulting formalisations of trace conditions lose much of their explanatory power.

In [13], Lee et al. propose a termination criterion for first-order programs called the *size-change principle*. This criterion can be considered an instance of the $\mathcal{T}_{\mathbb{B}}$ trace condition. Lee et al. give a decision procedure for the size-change principle which is a variant of the decision procedure for the Ramsey trace condition we give in Theorem 4, specialised to $\mathcal{T}_{\mathbb{B}}$.

The Ramsey trace condition (Definition 8) is not the first application of Ramsey’s theorem in the field of cyclic proof theory. Notably, existing proofs of the equivalence of (extensions of) cyclic arithmetic and (extensions of) Peano arithmetic [4, 8, 19] rely on (arithmetized) variants of Ramsey’s theorem for the direction from provability in cyclic arithmetic to provability in Peano arithmetic. It is required to “internalise” the soundness justification given by the global trace condition into a non-cyclic PA proof.

In some sense, every cyclic proof system in the literature might be considered related work, as one may wonder whether its trace condition can be modelled by our formalism. For most cyclic proof system we are aware of, including all those referred to in this article, this seems to be the case. The one exception is

the *bouncing-thread trace condition* in [3] which considers traces that do not run along the branches of a pre-proof.

7.2 Future Work

The modal μ -calculus has traditionally been the spawning ground for new developments in cyclic proof theory. However, these new results have sometimes been slow to transfer to other cyclic proof systems as it requires converting between the parity trace condition of modal μ -calculus and the Büchi-conditions of common cyclic proof systems. Using our framework, the relation between the $\mathbb{F}$ -activated μ -calculus and other $\mathcal{A}$ -activated systems is much more clear, which could ease the transfer of results. An exciting candidate for such an investigation is “proofs with names” [11, 20], a proof formalism which allows for more efficient proof checking and eases proof theoretic investigations (see e.g. [1, 2, 14]). Initially being put forward for the modal μ -calculus, it has so far been only adapted to alternation-free μ -calculus [14]. We hope to use our formalism to give a general account of proofs with names which will be uniformly applicable to various cyclic proof systems.

References

1. Afshari, B., Leigh, G.E.: Cut-free completeness for modal mu-calculus. In: 2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), pp. 1–12, June 2017. <https://doi.org/10.1109/LICS.2017.8005088>
2. Afshari, B., Leigh, G.E., Menéndez Turata, G.: Uniform interpolation from cyclic proofs: the case of modal mu-calculus. In: Das, A., Negri, S. (eds.) TABLEAUX 2021. LNCS (LNAI), vol. 12842, pp. 335–353. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86059-2_20
3. Baelde, D., Doumane, A., Kuperberg, D., Saurin, A.: Bouncing threads for inductive and circular proofs. [arXiv:2005.08257](https://arxiv.org/abs/2005.08257) [cs], May 2020
4. Berardi, S., Tatsuta, M.: Equivalence of inductive definitions and cyclic proofs under arithmetic. In: 2017 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), pp. 1–12 (2017). <https://doi.org/10.1109/LICS.2017.8005114>
5. Brotherston, J.: Sequent calculus proof systems for inductive definitions. Ph.D. thesis, University of Edinburgh, November 2006
6. Brotherston, J., Distefano, D., Petersen, R.L.: Automated cyclic entailment proofs in separation logic. In: Bjørner, N., Sofronie-Stokkermans, V. (eds.) CADE 2011. LNCS (LNAI), vol. 6803, pp. 131–146. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22438-6_12
7. Brotherston, J., Goriogiannis, N., Petersen, R.L.: A generic cyclic theorem prover. In: Jhala, R., Igarashi, A. (eds.) APLAS 2012. LNCS, vol. 7705, pp. 350–367. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35182-2_25
8. Das, A.: On the logical complexity of cyclic arithmetic. Log. Methods Comput. Sci. **16**(1) (2020). [https://doi.org/10.23638/LMCS-16\(1:1\)2020](https://doi.org/10.23638/LMCS-16(1:1)2020)
9. Das, A.: A circular version of Gödel’s T and its abstraction complexity. [arXiv:2012.14421](https://arxiv.org/abs/2012.14421) [cs, math], January 2021

10. Fortier, J., Santocanale, L.: Cuts for circular proofs: semantics and cut-elimination. In: Rocca, S.R.D. (ed.) *Computer Science Logic 2013 (CSL 2013)*. Leibniz International Proceedings in Informatics (LIPIcs), vol. 23, pp. 248–262. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl (2013). <https://doi.org/10.4230/LIPIcs.CSL.2013.248>
11. Jungteerapanich, N.: A tableau system for the modal μ -calculus. In: Giese, M., Waaler, A. (eds.) *TABLEAUX 2009*. LNCS (LNAI), vol. 5607, pp. 220–234. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02716-1_17
12. Kori, M., Tsukada, T., Kobayashi, N.: A cyclic proof system for HFLN. In: Baier, C., Goubault-Larrecq, J. (eds.) *29th EACSL Annual Conference on Computer Science Logic (CSL 2021)*. Leibniz International Proceedings in Informatics (LIPIcs), vol. 183, pp. 29:1–29:22. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl (2021). <https://doi.org/10.4230/LIPIcs.CSL.2021.29>
13. Lee, C.S., Jones, N.D., Ben-Amram, A.M.: The size-change principle for program termination. In: *Proceedings of the 28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2001*, pp. 81–92. Association for Computing Machinery, New York, January 2001. <https://doi.org/10.1145/360204.360210>
14. Marti, J., Venema, Y.: Focus-style proof systems and interpolation for the alternation-free μ -calculus. [arXiv:2103.01671](https://arxiv.org/abs/2103.01671) [cs], April 2021
15. Niwiński, D., Walukiewicz, I.: Games for the μ -calculus. *Theor. Comput. Sci.* **163**(1–2), 99–116 (1996). [https://doi.org/10.1016/0304-3975\(95\)00136-0](https://doi.org/10.1016/0304-3975(95)00136-0)
16. Nollet, R., Saurin, A., Tasson, C.: PSPACE-completeness of a thread criterion for circular proofs in linear logic with least and greatest fixed points. In: Cerrito, S., Popescu, A. (eds.) *TABLEAUX 2019*. LNCS (LNAI), vol. 11714, pp. 317–334. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-29026-9_18
17. Ramsey, F.P.: On a problem of formal logic. *Proc. Lond. Math. Soc.* **s2-30**(1), 264–286 (1930). <https://doi.org/10.1112/plms/s2-30.1.264>
18. Savateev, Y., Shamkanov, D.: Non-well-founded proofs for the Grzegorzcyk modal logic. *Rev. Symb. Logic* **14**(1), 22–50 (2021). <https://doi.org/10.1017/S1755020319000510>
19. Simpson, A.: Cyclic arithmetic is equivalent to Peano arithmetic. In: Esparza, J., Murawski, A.S. (eds.) *FoSSaCS 2017*. LNCS, vol. 10203, pp. 283–300. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54458-7_17
20. Stirling, C.: A proof system with names for modal μ -calculus. *Electron. Proc. Theor. Comput. Sci.* **129**, 18–29 (2013). <https://doi.org/10.4204/EPTCS.129.2>
21. Tellez, G., Brotherston, J.: Automatically verifying temporal properties of pointer programs with cyclic proof. In: de Moura, L. (ed.) *CADE 2017*. LNCS (LNAI), vol. 10395, pp. 491–508. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63046-5_30
22. Wehr, D.: An abstract framework for the analysis of cyclic derivations. M.Sc. thesis, University of Amsterdam, August 2021



Subordination Algebras as Semantic Environment of Input/Output Logic

Andrea De Domenico^{1(✉)}, Ali Farjami², Krishna Manoorkar¹,
Alessandra Palmigiano^{1,3}, Mattia Panettiere¹, and Xiaolong Wang¹

¹ Vrije Universiteit Amsterdam, Amsterdam, The Netherlands
a.de.domenico@vu.nl

² Iran University of Science and Technology, Tehran, Iran

³ Department of Mathematics and Applied Mathematics, University of
Johannesburg, Johannesburg, South Africa

Abstract. We establish a novel connection between two research areas in non-classical logics which have been developed independently of each other so far: on the one hand, *input/output logic*, introduced within a research program developing logical formalizations of normative reasoning in philosophical logic and AI; on the other hand, *subordination algebras*, investigated in the context of a research program integrating topological, algebraic, and duality-theoretic techniques in the study of the semantics of modal logic. Specifically, we propose that the basic framework of input/output logic, as well as its extensions, can be given formal semantics on (slight generalizations of) subordination algebras. The existence of this interpretation brings benefits to both research areas: on the one hand, this connection allows for a novel conceptual understanding of subordination algebras as mathematical models of the properties and behaviour of norms; on the other hand, thanks to the well developed connection between subordination algebras and modal logic, the output operators in input/output logic can be given a new formal representation as modal operators, whose properties can be explicitly axiomatised in a suitable language, and be systematically studied by means of mathematically established and powerful tools.

Keywords: input/output logic · subordination algebra · modal logic · Sahlqvist theory

1 Introduction

Input/output logic has been introduced in [25] as a formal framework for modelling the interaction between logical inferences and other agency-related notions such as conditional obligations, goals, ideals, preferences, actions, and beliefs. Although, initially, this framework was intended “not [for] studying some kind of non-classical logic, but [as] a way of using the classical one”, its generality and versatility makes it very suitable to support a range of enhancements in

its expressiveness, such as those brought about by the addition of modal operators. Moreover, recently, there has been an interest in studying the interaction between the agency-related notions mentioned above with various forms of *non-classical* reasoning [30, 34]. This interest has contextually motivated the introduction of algebraic and proof-theoretic methods in the study of input/output logic [18, 20, 36].

In this paper, we contribute to the latter research direction in the mathematical background of input/output logic¹ by introducing an algebraic semantics for it, based on (generalizations of) *subordination algebras* [4]. These can be defined as tuples $(A, \prec)$ such that A is a Boolean algebra and $\prec$ is a binary relation on A such that the direct (resp. inverse) image of each element $a \in A$ is a filter (resp. an ideal) of A .

Subordination algebras are equivalent presentations of pre-contact algebras [15] and quasi-modal algebras [6, 7]. Since their introduction, subordination algebras have been systematically connected with various modal algebras (i.e. Boolean algebras expanded with semantic modal operators). This has made it possible to endow various modal languages with algebraic semantics based on subordination algebras, and use these languages to axiomatize the properties of these subordination algebras. In particular, Sahlqvist-type canonicity for modal and tense formulas on subordination algebras has been studied in [13] using topological techniques; in [14], using algebraic techniques, the canonicity result of [13] was strengthened and captured within the more general notion of canonicity in the context of *slanted algebras*, which was established using the tools of *unified correspondence theory* [9, 11, 12]. Slanted algebras are based on general lattices, and encompass variations and generalizations of subordination algebras such as those very recently introduced by Celani in [8], which are based on distributive lattices, and for which Celani develops duality-theoretic and correspondence-theoretic results.

Structure of the Paper. In Sect. 2, we collect preliminaries about the abstract logical framework in which we are going to develop our results, input/output logics as embedded in this framework, the general environment of proto-subordination algebras and their properties, canonical extensions and slanted algebras. In Sect. 3, we associate slanted algebras to proto-subordination algebras with certain properties, and characterize their further properties in terms of the validity of modal inequalities on their associated slanted algebras. In Sect. 4, we provide an axiomatic modal characterization of the output operators of input/output logic (cf. Proposition 3), and to obtain Celani's dual characterization results for subordination lattices as consequences of standard modal correspondence (cf. Proposition 4). We conclude in Sect. 5.

¹ In the present work, we focus exclusively on the framework of *unconstrained* input/output logic [26]. Extending the results to its constrained version is an avenue for future work.

2 Preliminaries

Selfextensional Logics. In what follows, we align to the literature in abstract algebraic logic [19], and understand a *logic* to be a tuple $\mathcal{L} = (\text{Fm}, \vdash)$, such that Fm is the term algebra (in a given algebraic signature) over a set Prop of atomic propositions, and $\vdash$ is a *consequence relation* on Fm , i.e. $\vdash$ is a relation between sets of formulas and formulas such that, for all $\Gamma, \Delta \subseteq \text{Fm}$ and all $\varphi \in \text{Fm}$, (a) if $\varphi \in \Gamma$ then $\Gamma \vdash \varphi$; (b) if $\Gamma \vdash \varphi$ and $\Gamma \subseteq \Delta$, then $\Delta \vdash \varphi$; (c) if $\Delta \vdash \varphi$ and $\Gamma \vdash \psi$ for every $\psi \in \Delta$, then $\Gamma \vdash \varphi$. Clearly, any such $\vdash$ induces a preorder on Fm , which we still denote $\vdash$, by restricting to singletons. A logic $\mathcal{L}$ is *selfextensional* (cf. [22]) if the relation $\equiv \subseteq \text{Fm} \times \text{Fm}$, defined by $\varphi \equiv \psi$ iff $\varphi \vdash \psi$ and $\psi \vdash \varphi$, is a congruence of Fm . In this case, the *Lindenbaum-Tarski algebra* of $\mathcal{L}$ is the partially ordered algebra $Fm = (\text{Fm}/\equiv, \vdash)$ where, abusing notation, $\vdash$ also denotes the partial order on $\text{Fm}/\equiv$, defined as $[\varphi]_{\equiv} \vdash [\psi]_{\equiv}$ iff $\varphi \vdash \psi$. In what follows, we will also assume that each element in the class $\text{Alg}(\mathcal{L})$ of algebras canonically associated with $\mathcal{L}$ is partially ordered, and that, if φ and ψ are formulas, then $\varphi \vdash \psi$ iff $h(\varphi) \leq h(\psi)$ for every $A \in \text{Alg}(\mathcal{L})$ and every homomorphism $h : \text{Fm} \rightarrow A$.

For any $\Gamma \subseteq \text{Fm}$, let $Cn(\Gamma) := \{\psi \mid \Gamma \vdash \psi\}$.² The *conjunction property* holds for $\mathcal{L}$ if a term $t(x, y) := x \wedge y$ exists such that $Cn(\varphi \wedge \psi) = Cn(\{\varphi, \psi\})$ for all $\varphi, \psi \in \text{Fm}$. The *disjunction property* holds for $\mathcal{L}$ if a term $t(x, y) := x \vee y$ exists such that $Cn(\varphi \vee \psi) = Cn(\varphi) \cap Cn(\psi)$ for all $\varphi, \psi \in \text{Fm}$.

Although the original framework of input/output logic takes $\mathcal{L}$ to be classical propositional logic, in the next subsection we present it in the more general framework of selfextensional logics just described.

Input/Output Logic and (Proto-)Subordination Algebras. The general theory of input/output logic aims at modelling relations generalizing inference, where inputs need not be included among outputs, and outputs need not be reusable as inputs [25].

Definition 1. Let $\mathcal{L} = (\text{Fm}, \vdash)$ be a logic in the sense specified above. A normative system is a relation $N \subseteq \text{Fm} \times \text{Fm}$, the elements (α, φ) of which are called conditional norms (or obligations). An input/output logic is a tuple $\mathbb{L} = (\mathcal{L}, N)$ s.t. $\mathcal{L} = (\text{Fm}, \vdash)$ is a (selfextensional) logic, and N is a normative system on Fm .

The reading of each norm $(\alpha, \varphi) \in N$ is “given α , it is obligatory that φ ”. The formula α is the *body* of the norm, and represents some situation or condition, while φ is the *head* and represents what is obligatory or desirable in that situation. For any $\Gamma \subseteq \text{Fm}$, let $N(\Gamma) := \{\psi \mid \exists \alpha (\alpha \in \Gamma \ \& \ (\alpha, \psi) \in N)\}$.

Definition 2 (Output operations). For any input/output logic $\mathbb{L} = (\mathcal{L}, N)$, and each $1 \leq i \leq 4$, the output operation out_i^N is defined as follows: for any $\Gamma \subseteq \text{Fm}$,

$$out_i^N(\Gamma) := N_i(\Gamma) = \{\psi \in \text{Fm} \mid \exists \alpha (\alpha \in \Gamma \ \& \ (\alpha, \psi) \in N_i)\}$$

² In what follows, we write e.g. $Cn(\varphi)$ for $Cn(\{\varphi\})$.

where $N_i \subseteq \text{Fm} \times \text{Fm}$ is the closure of N under (i.e. the smallest extension of N satisfying) the inference rules below, as specified in the table.

$\frac{}{(\top, \top)} (\top)$	$\frac{(\alpha, \varphi) \quad \beta \vdash \alpha}{(\beta, \varphi)} (\text{SI})$	$\frac{(\alpha, \varphi) \quad \varphi \vdash \psi}{(\alpha, \psi)} (\text{WO})$
$\frac{(\alpha, \varphi) \quad (\alpha, \psi)}{(\alpha, \varphi \wedge \psi)} (\text{AND})$	$\frac{(\alpha, \varphi) \quad (\beta, \varphi)}{(\alpha \vee \beta, \varphi)} (\text{OR})$	$\frac{(\alpha, \varphi) \quad (\alpha \wedge \varphi, \psi)}{(\alpha, \psi)} (\text{CT})$
<i>N_i Rules</i>		
$N_1 (\top), (SI), (WO), (AND)$	$N_2 (\top), (SI), (WO), (AND), (OR)$	
$N_3 (\top), (SI), (WO), (AND), (CT)$	$N_4 (\top), (SI), (WO), (AND), (OR), (CT)$	

Definition 3 ((Proto-)subordination algebra). A proto-subordination algebra is a tuple $\mathbb{S} = (A, \prec)$ such that A is a (possibly bounded) poset (with bottom denoted $\perp$ and top denoted $\top$ when they exist), and $\prec \subseteq A \times A$. A proto-subordination algebra is named as indicated in the left-hand column in the tables below when $\prec$ satisfies the properties indicated in the right-hand column. In what follows, we will refer to a proto-subordination algebra $\mathbb{S} = (A, \prec)$ as e.g. (distributive) lattice-based ((D)L-based), or Boolean-based (B-based) if A is a (distributive) lattice, a Boolean algebra, and so on. More in general, for any logic $\mathcal{L}$, we say that $\mathbb{S} = (A, \prec)$ is $\text{Alg}(\mathcal{L})$ -based if $A \in \text{Alg}(\mathcal{L})$. The reader can safely assume that A is a (bounded distributive) lattice, or a Boolean algebra, although, if this is not specified, the results presented below will hold more generally. We will flag out the assumptions we need in the statements of propositions.

$(\perp) \perp \prec \perp$	$(\top) \top \prec \top$
$(SI) a \leq b \prec x \Rightarrow a \prec x$	$(WO) b \prec x \leq y \Rightarrow b \prec y$
$(AND) a \prec x \ \& \ a \prec y \Rightarrow a \prec x \wedge y$	$(OR) a \prec x \ \& \ b \prec x \Rightarrow a \vee b \prec x$
$(D) a \prec c \Rightarrow \exists b(a \prec b \ \& \ b \prec c)$	$(S6) a \prec b \Rightarrow \neg b \prec \neg a$
$(CT) a \prec b \ \& \ a \wedge b \prec c \Rightarrow a \prec c$	$(T) a \prec b \ \& \ b \prec c \Rightarrow a \prec c$
$(DD) a \prec x_1 \ \& \ a \prec x_2 \Rightarrow \exists x(a \prec x \ \& \ x \leq x_1 \ \& \ x \leq x_2)$	
$(UD) a_1 \prec x \ \& \ a_2 \prec x \Rightarrow \exists a(a \prec x \ \& \ a_1 \leq a \ \& \ a_2 \leq a)$	
$(S9) \exists c(c \prec b \ \& \ x \prec a \vee c) \iff \exists a' \exists b'(a' \prec a \ \& \ b' \prec b \ \& \ x \leq a' \vee b')$	
$(SL1) a \prec b \vee c \Rightarrow \exists b' \exists c'(b' \prec b \ \& \ c' \prec c \ \& \ a \prec b' \vee c')$	
$(SL2) b \wedge c \prec a \Rightarrow \exists b' \exists c'(b' \prec b \ \& \ c' \prec c \ \& \ b' \wedge c' \prec a)$	

Name	Properties	Name	Properties
$\diamond$ -premonotone	(SI)	directed/monotone	(SI) (WO) (UD) (DD)
$\blacksquare$ -premonotone	(WO)	$\diamond$ -regular	(SI) (WO) (DD) (OR)
premonotone	(SI) (WO)	$\blacksquare$ -regular	(SI) (WO) (UD) (AND)
$\diamond$ -directed	(WO) (DD)	regular	(SI) (WO) (OR) (AND)
$\blacksquare$ -directed	(SI) (UD)	$\diamond$ -normal	(SI) (WO) (DD) (OR) ($\perp$)
$\diamond$ -monotone	(WO) (DD) (SI)	$\blacksquare$ -normal	(SI) (WO) (UD) (AND) ($\top$)
$\blacksquare$ -monotone	(SI) (UD) (WO)	subordination algebra	(SI) (WO) (OR) (AND) ($\perp$) ($\top$)

Normative systems can be interpreted in proto-subordination algebras as follows:

Definition 4. A model for an input/output logic $\mathbb{L} = (\mathcal{L}, N)$ is a tuple $\mathbb{M} = (\mathbb{S}, h)$ s.t. $\mathbb{S} = (A, \prec)$ is an $\text{Alg}(\mathcal{L})$ -based proto-subordination algebra (i.e. $A \in \text{Alg}(\mathcal{L})$), and $h : \text{Fm} \rightarrow A$ is a homomorphism s.t. for all $\varphi, \psi \in \text{Fm}$, if $(\varphi, \psi) \in N$, then $h(\varphi) \prec h(\psi)$.

Canonical Extensions and Slanted Algebras. In the present subsection, we adapt material from [13, Sects.2.2 and 3.1], [16, Sect. 2]. For any poset A , a subset $B \subseteq A$ is *upward closed*, or an *up-set* (resp. *downward closed*, or a *down-set*) if $\lfloor B \rfloor := \{c \in A \mid \exists b(b \in B \ \& \ b \leq c)\} \subseteq B$ (resp. $\lceil B \rceil := \{c \in A \mid \exists b(b \in B \ \& \ c \leq b)\} \subseteq B$); a subset $B \subseteq A$ is *down-directed* (resp. *up-directed*) if, for all $a, b \in B$, some $x \in B$ exists s.t. $x \leq a$ and $x \leq b$ (resp. $a \leq x$ and $b \leq x$). It is straightforward to verify that when A is a lattice, down-directed upsets and up-directed down-sets coincide with lattice filters and ideals, respectively.

Definition 5. Let A be a subposet of a complete lattice A' .

- (i) An element $k \in A'$ is closed if $k = \bigwedge F$ for some down-directed $F \subseteq A$; an element $o \in A'$ is open if $o = \bigvee I$ for some up-directed $I \subseteq A$;
- (ii) A is dense in A' if every element of A' can be expressed both as the join of closed elements and as the meet of open elements of A .
- (iii) A is compact in A' if, for all $F, I \subseteq A$ s.t. F is down-directed, I is up-directed, if $\bigwedge F \leq \bigvee I$ then $a \leq b$ for some $a \in F$ and $b \in I$.³
- (iv) The canonical extension of a poset A is a complete lattice A^δ containing A as a dense and compact subposet.

The canonical extension A^δ of any poset A always exists and is unique up to an isomorphism fixing A (cf. [16, Propositions 2.6 and 2.7]). The set of the closed (resp. open) elements of A^δ is denoted $K(A^\delta)$ (resp. $O(A^\delta)$). The following proposition collects well known facts used in the remainder of the paper.

Proposition 1. For every poset A ,

- (i) if A is a distributive lattice (DL), then A^δ is completely distributive.
- (ii) if $\neg : A \rightarrow A$ is antitone and s.t. $(A, \neg) \models \forall a \forall b (\neg a \leq b \Leftrightarrow \neg b \leq a)$, then $\neg^\sigma : A^\delta \rightarrow A^\delta$ defined as $\neg^\sigma o := \bigwedge \{\neg a \mid a \leq o\}$ for any $o \in O(A^\delta)$ and $\neg^\sigma u := \bigvee \{\neg^\sigma o \mid u \leq o\}$ for any $u \in A^\delta$ is antitone and s.t. $(A^\delta, \neg^\sigma) \models \forall u \forall v (\neg u \leq v \Leftrightarrow \neg v \leq u)$. If in addition, $(A, \neg) \models a \leq \neg \neg a$, then $(A^\delta, \neg^\sigma) \models u \leq \neg \neg u$. Hence, if $(A, \neg) \models a = \neg \neg a$ (i.e. $\neg$ is involutive), then $(A^\delta, \neg^\sigma) \models a = \neg \neg a$.
- (iii) if $\neg : A \rightarrow A$ is antitone and s.t. $(A, \neg) \models \forall a \forall b (a \leq \neg b \Leftrightarrow b \leq \neg a)$, then $\neg^\pi : A^\delta \rightarrow A^\delta$ defined as $\neg^\pi k := \bigvee \{\neg a \mid k \leq a\}$ for any $k \in K(A^\delta)$ and $\neg^\pi u := \bigwedge \{\neg^\pi k \mid k \leq u\}$ for any $u \in A^\delta$ is antitone and s.t. $(A^\delta, \neg^\pi) \models \forall u \forall v (u \leq \neg v \Leftrightarrow v \leq \neg u)$. If in addition, $(A, \neg) \models \neg \neg a \leq a$, then $(A^\delta, \neg^\pi) \models \neg \neg u \leq u$. Hence, if $\neg$ is involutive, then so is $\neg^\pi$.

Definition 6. A slanted algebra is a triple $\mathbb{A} = (A, \diamond, \blacksquare)$ such that A is a poset, and $\diamond, \blacksquare : A \rightarrow A^\delta$ s.t. $\diamond a \in K(A^\delta)$ and $\blacksquare a \in O(A^\delta)$ for every a . A slanted algebra as above is tense if $\diamond a \leq b$ iff $a \leq \blacksquare b$ for all $a, b \in A$; is monotone if $\diamond$ and $\blacksquare$ are monotone; is regular if $\diamond$ and $\blacksquare$ are regular (i.e. $\diamond(a \vee b) = \diamond a \vee \diamond b$ and $\blacksquare(a \wedge b) = \blacksquare a \wedge \blacksquare b$ for all $a, b \in A$); is normal if $\diamond$ and $\blacksquare$ are normal (i.e. they are regular and $\diamond \perp = \perp$ and $\blacksquare \top = \top$).

³ When the poset A is a lattice, the compactness can be equivalently reformulated by dropping the requirements that F be down-directed and I be up-directed.

The following definition is framed in the context of monotone slanted algebras, but can be given for arbitrary slanted algebras, albeit at the price of complicating the definition of $\Diamond^\sigma$ and $\blacksquare^\pi$. Because we are mostly going to apply it in the monotone setting, we present the simplified version here.

Definition 7. For any monotone slanted algebra $\mathbb{A} = (A, \Diamond, \blacksquare)$ the canonical extension of $\mathbb{A}$ is the (standard!) modal algebra $\mathbb{A}^\delta := (A^\delta, \Diamond^\sigma, \blacksquare^\pi)$ such that $\Diamond^\sigma, \blacksquare^\pi : A^\delta \rightarrow A^\delta$ are defined as follows: for every $k \in K(A^\delta)$, $o \in O(A^\delta)$ and $u \in A^\delta$,

$$\begin{aligned} \Diamond^\sigma k &:= \bigwedge \{ \Diamond a \mid a \in A \text{ and } k \leq a \}, & \Diamond^\sigma u &:= \bigvee \{ \Diamond^\sigma k \mid k \in K(A^\delta) \text{ and } k \leq u \}, \\ \blacksquare^\pi o &:= \bigvee \{ \blacksquare a \mid a \in A \text{ and } a \leq o \}, & \blacksquare^\pi u &:= \bigwedge \{ \blacksquare^\pi o \mid o \in O(A^\delta) \text{ and } u \leq o \}. \end{aligned}$$

For any slanted algebra $\mathbb{A}$, any assignment $v : \text{PROP} \rightarrow \mathbb{A}$ uniquely extends to a homomorphism $v : \mathcal{L} \rightarrow \mathbb{A}^\delta$ (abusing notation, the same symbol denotes both the assignment and its homomorphic extension). Hence,

Definition 8. A modal inequality $\phi \leq \psi$ is satisfied in a slanted algebra $\mathbb{A}$ under the assignment v (notation: $(\mathbb{A}, v) \models \phi \leq \psi$) if $(\mathbb{A}^\delta, e \cdot v) \models \phi \leq \psi$ in the usual sense, where $e \cdot v$ is the assignment on $\mathbb{A}^\delta$ obtained by composing the canonical embedding $e : A \rightarrow A^\delta$ to the assignment $v : \text{Prop} \rightarrow \mathbb{A}$.

Moreover, $\phi \leq \psi$ is valid in $\mathbb{A}$ (notation: $\mathbb{A} \models \phi \leq \psi$) if $(\mathbb{A}^\delta, e \cdot v) \models \phi \leq \psi$ for every assignment v into $\mathbb{A}$ (notation: $\mathbb{A}^\delta \models_{\mathbb{A}} \phi \leq \psi$).

3 Proto-Subordination Algebras and Slanted Algebras

Let $\mathbb{S} = (A, \prec)$ be a proto-subordination algebra s.t. $\mathbb{S} \models (\text{DD}) + (\text{UD})$. The slanted algebra associated with $\mathbb{S}$ is $\mathbb{S}^* = (A, \Diamond, \blacksquare)$ s.t. $\Diamond a := \bigwedge \prec[a]$ and $\blacksquare a := \bigvee \prec^{-1}[a]$ for any a . From $\mathbb{S} \models (\text{DD})$ it follows that $\prec[a]$ is down-directed for every $a \in A$, hence $\Diamond a \in K(A^\delta)$. Likewise, $\mathbb{S} \models (\text{UD})$ guarantees that $\blacksquare a \in O(A^\delta)$ for all $a \in A$.

Lemma 1. For any proto-subordination algebra $\mathbb{S} = (A, \prec)$ and all $a, b \in A$,

- (i) $a \prec b$ implies $\Diamond a \leq b$ and $a \leq \blacksquare b$.
- (ii) if $\mathbb{S} \models (\text{WO}) + (\text{DD})$, then $\Diamond a \leq b$ iff $a \prec b$.
- (iii) if $\mathbb{S} \models (\text{SI}) + (\text{UD})$, then $a \leq \blacksquare b$ iff $a \prec b$.

Proof. (i) $a \prec b$ iff $b \in \prec[a]$ iff $a \in \prec^{-1}[b]$, hence $a \prec b$ implies $b \geq \bigwedge \prec[a] = \Diamond a$ and $a \leq \bigvee \prec^{-1}[b] = \blacksquare b$.

- (ii) By (i), to complete the proof, we need to show the ‘only if’ direction. The assumption $\mathbb{S} \models (\text{DD})$ implies that $\prec[a]$ is down-directed for any $a \in A$. Hence, by compactness, $\bigwedge \prec[a] = \Diamond a \leq b$ implies that $c \leq b$ for some $c \in \prec[a]$, i.e. $a \prec c \leq b$ for some $c \in A$, and by (WO), this implies that $a \prec b$, as required.

- (iii) is proven similarly.

Lemma 2. *For any lattice-based proto-subordination algebra $\mathbb{S} = (A, \prec)$,*

- (i) $\mathbb{S} \models (\text{OR})$ implies $\mathbb{S} \models (\text{UD})$.
- (ii) $\mathbb{S} \models (\text{AND})$ implies $\mathbb{S} \models (\text{DD})$.
- (iii) If $\mathbb{S} \models (\text{SI})$, then $\mathbb{S} \models (\text{UD})$ iff $\mathbb{S} \models (\text{OR})$.
- (iv) if $\mathbb{S} \models (\text{WO})$, then $\mathbb{S} \models (\text{DD})$ iff $\mathbb{S} \models (\text{AND})$.

Proof. (i) and (ii) are straightforward. As for (iii), by (i), to complete the proof we need to show the ‘only if’ direction. Let $a, b, x \in A$ s.t. $a \prec x$ and $b \prec x$. By (UD), this implies that $c \prec x$ for some $c \in A$ such that $a \leq c$ and $b \leq c$. Since A is a lattice, this implies that $a \vee b \leq c \prec x$, and by (SI), this implies that $a \vee b \prec x$, as required. (vi) is proven similarly.

Lemma 3. *For every proto-subordination algebra $\mathbb{S} = (A, \prec)$,*

- (i) If $\mathbb{S} \models (\text{SI})$, then:
 - (a) $\diamond$ on $\mathbb{S}^*$ is monotone;
 - (b) if $\mathbb{S}$ is DL-based, then $\mathbb{S} \models (\text{AND})$ implies $\mathbb{S}^* \models \blacksquare a \wedge \blacksquare b \leq \blacksquare(a \wedge b)$;
 - (c) if $\mathbb{S} \models (\text{UD})$, then $\mathbb{S} \models (\text{AND})$ implies $\mathbb{S}^* \models \blacksquare a \wedge \blacksquare b \leq \blacksquare(a \wedge b)$.
- (ii) If $\mathbb{S} \models (\text{WO})$, then
 - (a) $\blacksquare$ on $\mathbb{S}^*$ is monotone;
 - (b) if $\mathbb{S}$ is DL-based, then $\mathbb{S} \models (\text{OR})$ implies $\mathbb{S}^* \models \diamond(a \vee b) \leq \diamond a \vee \diamond b$;
 - (c) if $\mathbb{S} \models (\text{DD})$, then $\mathbb{S} \models (\text{OR})$ implies $\mathbb{S}^* \models \diamond(a \vee b) \leq \diamond a \vee \diamond b$.
- (iii) If $\mathbb{S} \models (\perp)$, then $\mathbb{S}^* \models \diamond \perp \leq \perp$.
- (iv) If $\mathbb{S} \models (\top)$, then $\mathbb{S}^* \models \top \leq \blacksquare \top$.

Proof. (i) (a) Let $a, b \in A$ s.t. $a \leq b$. To show that $\diamond a = \bigwedge \prec[a] \leq \bigwedge \prec[b] = \diamond b$, it is enough to show that $\prec[b] \subseteq \prec[a]$, i.e. that if $x \in A$ and $b \prec x$, then $a \prec x$. Indeed, by (SI), $a \leq b \prec x$ implies $a \prec x$, as required. (ii) (a) is shown similarly.

(ii) (b) Let $a, b \in A$. By definition, $\diamond(a \vee b) = \bigwedge \prec[a \vee b] = \bigwedge \{d \mid a \vee b \prec d\}$, and, since A^δ is completely distributive when A is a DL (cf. Proposition 1(i)),

$$\diamond a \vee \diamond b = (\bigwedge \prec[a]) \vee (\bigwedge \prec[b]) = \bigwedge \{c \vee c' \mid a \prec c \text{ and } b \prec c'\}.$$

So, to show that $\diamond(a \vee b) \leq \diamond a \vee \diamond b$, it is enough to show that $\{c \vee c' \mid a \prec c \text{ and } b \prec c'\} \subseteq \{d \mid a \vee b \prec d\}$, i.e. that for all $c, c' \in A$, if $a \prec c$ and $b \prec c'$, then $a \vee b \prec c \vee c'$. By (WO), $a \prec c \leq c \vee c'$ and $b \prec c' \leq c \vee c'$ imply that $a \prec c \vee c'$ and $b \prec c \vee c'$, which by (OR) implies that $a \vee b \prec c \vee c'$, as required. (i) (b) is argued similarly. (ii) (c) To show that $\diamond(a \vee b) \leq \diamond a \vee \diamond b$, it is enough to show that for any $x \in A$, if $\diamond a \vee \diamond b \leq x$, then $\diamond(a \vee b) \leq x$. By Lemma 1 (ii) $\diamond a \vee \diamond b \leq x$ iff $a \prec x$ and $b \prec x$, we now apply (OR) and Lemma 1 (i) to finish the proof. (i)(c) is proven similarly. (iii) By assumption, $\perp \prec \perp$, i.e. $\perp \in \prec[\perp]$, which implies $\diamond \perp = \bigwedge \prec[\perp] \leq \perp$, as required. (iv) is argued similarly.

The following lemma gives a converse of Lemma 3 for $\diamond$ -directed or $\blacksquare$ -directed proto-subordination algebras.

Lemma 4. *For any proto-subordination algebra $\mathbb{S} = (A, \prec)$,*

- (i) *If $\mathbb{S} \models (\text{WO}) + (\text{DD})$, then:*
 - (a) $\mathbb{S} \models (\text{SI})$ iff $\Diamond$ on $\mathbb{S}^*$ is monotone.
 - (b) $\mathbb{S} \models (\text{OR})$ iff $\mathbb{S}^* \models \Diamond(a \vee b) \leq \Diamond a \vee \Diamond b$.
 - (c) $\mathbb{S} \models (\perp)$ iff $\mathbb{S}^* \models \Diamond \perp \leq \perp$.
- (ii) *If $\mathbb{S} \models (\text{SI}) + (\text{UD})$, then:*
 - (a) $\mathbb{S} \models (\text{WO})$ iff $\blacksquare$ on $\mathbb{S}^*$ is monotone;
 - (b) $\mathbb{S} \models (\text{AND})$ iff $\mathbb{S}^* \models \blacksquare a \wedge \blacksquare b \leq \blacksquare(a \wedge b)$;
 - (c) $\mathbb{S} \models (\top)$ iff $\mathbb{S}^* \models \top \leq \blacksquare \top$.

Proof. We only show the items in (i), the proofs of those in (ii) being similar. (a) By Lemma 3 (i)(a), the proof is complete if we show the ‘if’ direction. Let $a, b, x \in A$ s.t. $a \leq b \prec x$. By Lemma 1 (ii), to show that $a \prec x$, it is enough to show that $\Diamond a \leq x$. Since $\Diamond$ is monotone, $a \leq b$ implies $\Diamond a \leq \Diamond b$, and, again by Lemma 1 (ii), $b \prec x$ implies that $\Diamond b \leq x$. Hence, $\Diamond a \leq x$, as required.

(b) By Lemma 3 (ii)(c), the proof is complete if we show the ‘if’ direction. Let $a, b, x \in A$ s.t. $a \prec x$ and $b \prec x$. By Lemma 1 (ii), to show that $a \vee b \prec x$, it is enough to show that $\Diamond(a \vee b) \leq x$, and since $\mathbb{S}^* \models \Diamond(a \vee b) \leq \Diamond a \vee \Diamond b$, it is enough to show that $\Diamond a \vee \Diamond b \leq x$, i.e. that $\Diamond a \leq x$ and $\Diamond b \leq x$. These two inequalities hold by Lemma 1 (ii), and the assumptions on a, b and x .

(c) By Lemma 1 (ii), $\perp \prec \perp$ is equivalent to $\Diamond \perp \leq \perp$, as required.

Corollary 1. *For every directed proto-subordination algebra $\mathbb{S} = (A, \prec)$,*

- (i) $\mathbb{S}$ is monotone iff $\mathbb{S}^*$ is monotone;
- (ii) $\mathbb{S}$ is regular iff $\mathbb{S}^*$ is regular;
- (iii) $\mathbb{S}$ is a subordination algebra iff $\mathbb{S}^*$ is normal.

Lemma 5. *For any proto-subordination algebra $\mathbb{S} = (A, \prec)$, for all $a, b \in A$, $k \in K(A^\delta)$, and $o \in O(A^\delta)$, and all $D, U \subseteq A$,*

- (i) *if $\mathbb{S} \models (\text{SI}) + (\text{DD}) + (\text{WO})$, then*
 - (a) *if $D \subseteq A$ is down-directed, then so is $\prec[D] := \{c \mid \exists a(a \in D \ \& \ a \prec c)\}$;*
 - (b) *if $k = \bigwedge D$ for some down-directed $D \subseteq A$, then $\Diamond k = \bigwedge \prec[D] \in K(A^\delta)$;*
 - (c) *$\Diamond k \leq b$ implies $a \prec b$ for some $a \in A$ s.t. $k \leq a$.*
 - (d) *$\Diamond k \leq o$ implies $a \prec b$ for some $a, b \in A$ s.t. $k \leq a$ and $b \leq o$.*
- (ii) *if $\mathbb{S} \models (\text{WO}) + (\text{UD}) + (\text{SI})$, then*
 - (a) *if $U \subseteq A$ is up-directed, then so is $\prec^{-1}[U] := \{c \mid \exists a(a \in U \ \& \ c \prec a)\}$;*
 - (b) *if $o = \bigvee U$ for some up-directed $U \subseteq A$, then $\blacksquare o = \bigvee \prec^{-1}[U] \in O(A^\delta)$;*
 - (c) *$a \leq \blacksquare o$ implies $a \prec b$ for some $b \in A$ s.t. $b \leq o$.*
 - (d) *$k \leq \blacksquare o$ implies $a \prec b$ for some $a, b \in A$ s.t. $k \leq a$ and $b \leq o$.*

Proof. We only prove (i), the proof of (ii) being similar.

- (a) If $c_i \in \prec[D]$ for $1 \leq i \leq 2$, then $a_i \prec c_i$ for some $a_i \in D$. Since D is down-directed, some $a \in D$ exists s.t. $a \leq a_i$ for each i . Thus, (SI) implies that $a \prec c_i$, from which the claim follows by (DD).

- (b) By definition, $\diamond k = \bigwedge \{\diamond a \mid a \in A, k \leq a\} = \bigwedge \{c \mid \exists a(a \prec c \ \& \ k \leq a)\}$. Since $k = \bigwedge D$ for some $D \subseteq A$ down-directed, by compactness, $k \leq a$ implies $d \leq a$ for some $d \in D$, thus $\diamond k = \bigwedge \{c \mid \exists a(a \prec c \ \& \ k \leq a)\} = \bigwedge \{c \mid \exists a(a \in \downarrow D \ \& \ a \prec c)\} = \bigwedge \prec[D] \in K(A^\delta)$, the last membership holding by (a).
- (c) By (b), $\diamond k \in K(A^\delta)$. Hence, $\diamond k \leq b$ implies by compactness that $c \leq b$ for some $c \in A$ s.t. $a \prec c$ for some $a \in D$ (hence $k = \bigwedge D \leq a$). By (WO), this implies that $a \prec b$ for some $a \in A$ s.t. $k \leq a$, as required.
- (d) By (b), $\diamond k \in K(A^\delta)$. Since $o \in O(A^\delta)$, some updirected $U \subseteq A$ exists s.t. $o = \bigvee U$. Hence, by compactness, $\diamond k \leq o$ implies that $a \prec b$ for some $a \in A$ s.t. $k \leq a$ and some $b \in U$ (for which $b \leq o$).

Proposition 2. *For any proto-subordination algebra $\mathbb{S} = (A, \prec)$,*

- (i) $\mathbb{S} \models \prec \subseteq \leq$ iff $\mathbb{S}^* \models a \leq \diamond a$ iff $\mathbb{S}^* \models \blacksquare a \leq a$.
- (ii) If $\mathbb{S} \models (\text{WO}) + (\text{DD})$, then $\mathbb{S} \models \leq \subseteq \prec$ iff $\mathbb{S}^* \models \diamond a \leq a$;
- (iii) if $\mathbb{S} \models (\text{WO}) + (\text{DD}) + (\text{SI})$, then
 - (a) $\mathbb{S} \models (\text{T})$ iff $\mathbb{S}^* \models \diamond a \leq \diamond \diamond a$.
 - (b) $\mathbb{S} \models (\text{D})$ iff $\mathbb{S}^* \models \diamond \diamond a \leq \diamond a$.
- (iv) if $\mathbb{S} \models (\text{WO}) + (\text{DD}) + (\text{SI})$ and is meet-semilattice based, then
 - (a) $\mathbb{S} \models (\text{CT})$ iff $\mathbb{S}^* \models \diamond a \leq \diamond(a \wedge \diamond a)$.
 - (b) $\mathbb{S} \models (\text{SL2})$ iff $\mathbb{S}^* \models \diamond(\diamond a \wedge \diamond b) \leq \diamond(a \wedge b)$.
- (v) if $\mathbb{S} \models (\text{SI})$, then $\mathbb{S} \models (\text{CT})$ implies $\mathbb{S} \models (\text{T})$.
- (vi) if $\mathbb{S}$ is directed and based on $(A, \neg)$ with $\neg$ antitone, involutive, and (left or right) self-adjoint,
 - (a) $\mathbb{S} \models (\text{S6})$ iff $\mathbb{S}^* \models \neg \diamond a = \blacksquare \neg a$, thus $\blacksquare a := \neg \diamond \neg a$.
 - (b) $\mathbb{S} \models (\text{S6})$ iff $\mathbb{S}^* \models \diamond \neg a = \neg \blacksquare a$, thus $\diamond a := \neg \blacksquare \neg a$.
- (vii) If $\mathbb{S} \models (\text{SI}) + (\text{UD}) + (\text{WO})$ and is join-semilattice based, then
 - (a) $\mathbb{S} \models (\text{S9} \Rightarrow)$ iff $\mathbb{S}^* \models \blacksquare(a \vee \blacksquare b) \leq \blacksquare a \vee \blacksquare b$.
 - (b) $\mathbb{S} \models (\text{S9} \Leftarrow)$ iff $\mathbb{S}^* \models \blacksquare a \vee \blacksquare b \leq \blacksquare(a \vee \blacksquare b)$.
 - (c) $\mathbb{S} \models (\text{SL1})$ iff $\mathbb{S}^* \models \blacksquare(a \vee b) \leq \blacksquare(\blacksquare a \vee \blacksquare b)$.

Proof. (i) By definition, $\forall a(a \leq \diamond a)$ iff $\forall a(a \leq \bigwedge \prec[a])$ iff $\forall a(a \leq \bigwedge \{b \in A \mid a \prec b\})$ iff $\forall a \forall b(a \prec b \Rightarrow a \leq b)$ iff $\prec \subseteq \leq$. The second part of the statement is proved similarly.

(ii) By Lemma 1 (i), if $a \prec a$, then $\diamond a \leq a$. Hence, the left-to-right direction follows from the reflexivity of $\leq$ and the assumption. Conversely, $\diamond a \leq a$ and $a \leq b$ imply $\diamond a \leq b$, which, by Lemma 1 (ii) and $\mathbb{S} \models (\text{WO}) + (\text{DD})$, is equivalent to $a \prec b$.

(iii) (a) From left to right,

$$\begin{aligned}
 \diamond \diamond a &= \bigwedge \{\diamond b \mid \diamond a \leq b\} && \text{Definition 7 applied to } \diamond a \in K(A^\delta) \\
 &= \bigwedge \{\diamond b \mid a \prec b\} && \text{Lemma 1 (ii) since } \mathbb{S} \models (\text{WO}) + (\text{DD}) \\
 &= \bigwedge \{c \mid \exists b(a \prec b \ \& \ b \prec c)\} && \diamond b = \bigwedge \{c \in A \mid b \prec c\}
 \end{aligned}$$

Hence, to show that $\diamond a = \bigwedge \{c \mid a \prec c\} \leq \diamond \diamond a$, it is enough to show that $\{c \mid \exists b(a \prec b \ \& \ b \prec c)\} \subseteq \{c \mid a \prec c\}$, which is immediately implied by the assumption (T). Conversely, let $a, b, c \in A$ s.t. $a \prec b$ and $b \prec c$. To show

that $a \prec c$, by Lemma 1 (ii), and $\mathbb{S} \models (\text{WO}) + (\text{DD})$, it is enough to show that $\Diamond a \leq c$, and since $\Diamond a \leq \Diamond \Diamond a$, it is enough to show that $\Diamond \Diamond a \leq c$. The assumption $a \prec b$ implies $\Diamond a \leq b$ which implies $\Diamond \Diamond a \leq \Diamond b$, by the monotonicity of $\Diamond$ (which depends on (SI), cf. Lemma 3(i)(a)). Hence, combining the latter inequality with $\Diamond b \leq c$ (which is implied by $b \prec c$), by the transitivity of $\leq$, we get $\Diamond \Diamond a \leq c$, as required.

(iii) (b) From left to right, by the definitions spelled out in the proof of (ii) (b), it is enough to show that $\{c \mid a \prec c\} \subseteq \{c \mid \exists b(a \prec b \ \& \ b \prec c)\}$, which is immediately implied by the assumption (D). Conversely, let $a, c \in A$ s.t. $a \prec c$, and let us show that $a \prec b$ and $b \prec c$ for some $b \in A$. The assumption $a \prec c$ implies $\Diamond a \leq c$. Since $\Diamond \Diamond a \leq \Diamond a$, this implies $\Diamond \Diamond a \leq c$, i.e. (see discussion above) $\bigwedge \{d \in A \mid \exists b(a \prec b \ \& \ b \prec d)\} \leq c$.

We claim that $D := \{d \in A \mid \exists b(a \prec b \ \& \ b \prec d)\}$ is down-directed: indeed, if $d_1, d_2 \in A$ s.t. $\exists b_i(a \prec b_i \ \& \ b_i \prec d_i)$ for $1 \leq i \leq 2$, then by (DD), some $b \in A$ exists s.t. $a \prec b$ and $b \leq b_i$. By (SI), $b \leq b_i \prec d_i$ implies $b \prec d_i$. By (DD) again, this implies that some $d \in A$ exists s.t. $b \prec d$ and $d \leq d_i$, which concludes the proof of the claim.

By compactness, $d \leq c$ for some $d \in A$ s.t. $a \prec b$ and $b \prec d$ for some $b \in A$. To finish the proof, it is enough to show that $b \prec c$, which is immediately implied by $b \prec d \leq c$ and (SI).

(iv) (a) From left to right, Let $a \in A$. If $\prec[a] = \emptyset$, then $\Diamond a = \bigwedge \emptyset = \top = \Diamond(a \wedge \top) = \Diamond(a \wedge \Diamond a)$, as required. If $\prec[a] \neq \emptyset$, then $a \wedge \Diamond a = \bigwedge \{a \wedge e \mid a \prec e\} \in K(A^\delta)$, since, by (DD), $\{a \wedge e \mid a \prec e\}$ is down-directed. Hence,

$$\begin{aligned} \Diamond(a \wedge \Diamond a) &= \bigwedge \{\Diamond c \mid a \wedge \Diamond a \leq c\} && \text{definition of } \Diamond \text{ on } K(A^\delta) \\ &= \bigwedge \{d \mid \exists c(c \prec d \ \& \ a \wedge \Diamond a \leq c)\} && \text{definition of } \Diamond \text{ on } A \end{aligned}$$

By compactness, $\bigwedge \{a \wedge e \mid a \prec e\} = a \wedge \Diamond a \leq c$ is equivalent to $a \wedge e \leq c$ for some $e \in A$ such that $a \prec e$. Thus,

$$\begin{aligned} \Diamond(a \wedge \Diamond a) &= \bigwedge \{d \mid \exists c(c \prec d \ \& \ a \wedge \Diamond a \leq c)\} \\ &= \bigwedge \{d \mid \exists c(c \prec d \ \& \ \exists e(a \prec e \ \& \ a \wedge e \leq c))\}. \end{aligned}$$

To finish the proof that $\Diamond a = \bigwedge \{d \mid a \prec d\} \leq \Diamond(a \wedge \Diamond a)$, it is enough to show that if $d \in A$ is such that $\exists c(c \prec d \ \& \ \exists e(a \prec e \ \& \ a \wedge e \leq c))$ then $a \prec d$. Since $a \wedge e \leq c$ and $c \prec d$, by (SI), $a \wedge e \prec d$. Hence, by (CT), from $a \prec e$ and $a \wedge e \prec d$ it follows that $a \prec d$, as required.

Conversely, assume that $\Diamond a \leq \Diamond(a \wedge \Diamond a)$ holds for any a . By (WO), (DD), and Lemma 1 (ii), (CT) can be equivalently rewritten as follows:

$$\text{if } \Diamond a \leq b \quad \text{and} \quad \Diamond(a \wedge b) \leq c, \text{ then } \Diamond a \leq c.$$

Since $\wedge$ is monotone, $\Diamond a \leq b$ implies that $a \wedge \Diamond a \leq a \wedge b$, which implies, by the monotonicity of $\Diamond$ (which is implied by (SI), cf. Lemma 3(i)(a)), that $\Diamond(a \wedge \Diamond a) \leq \Diamond(a \wedge b)$. Hence, combining the latter inequality with $\Diamond a \leq \Diamond(a \wedge \Diamond a)$

and $\diamond(a \wedge b) \leq c$, by the transitivity of $\leq$, we get $\diamond a \leq c$, as required. The proof of (iv)(b) is dual to that of (vii)(c), and is omitted.

(v) Let $a, b, c \in A$. If $a \prec b$ and $b \prec c$, then by (SI), $a \wedge b \leq b \prec c$ implies $a \wedge b \prec c$, which, by (CT), implies $a \prec c$, as required.

(vi) By Proposition 1(ii) and (iii), both extensions $\neg^\sigma$ and $\neg^\pi$ on A^δ are involutive. Hence, the following chains of equivalences hold under both interpretations of the negation, and thus, abusing notation, we omit the superscript.

$$\begin{aligned}
& \neg \diamond a \leq \blacksquare \neg a \\
& \text{iff } \neg \blacksquare \neg a \leq \diamond a = \bigwedge \{b \mid a \prec b\} \quad \neg \text{ antitone and involutive} \\
& \text{iff } \neg \blacksquare \neg a \leq b \text{ for all } b \in A \text{ s.t. } a \prec b \\
& \text{iff } \neg b \leq \blacksquare \neg a \text{ for all } b \in A \text{ s.t. } a \prec b \quad \neg \text{ antitone and involutive} \\
& \text{iff } \neg b \prec \neg a \text{ for all } b \in A \text{ s.t. } a \prec b \quad \text{Lemma 1 (iii), (SI), (UD)} \\
\\
& \diamond \neg b \leq \neg \blacksquare b \\
& \text{iff } \bigvee \{a \mid a \prec b\} = \blacksquare b \leq \neg \diamond \neg b \quad \neg \text{ antitone and involutive} \\
& \text{iff } a \leq \neg \diamond \neg b \text{ for all } a \in A \text{ s.t. } a \prec b \\
& \text{iff } \diamond \neg b \leq \neg a \text{ for all } a \in A \text{ s.t. } a \prec b \quad \neg \text{ antitone and involutive} \\
& \text{iff } \neg b \prec \neg a \text{ for all } a \in A \text{ s.t. } a \prec b \quad \text{Lemma 1 (ii), (WO), (DD)}
\end{aligned}$$

Since $\neg$ is involutive, condition (S6) can equivalently be rewritten as $\forall a \forall b (a \prec \neg b \Rightarrow b \prec \neg a)$ and as $\forall a \forall b (\neg a \prec b \Rightarrow \neg b \prec a)$. Hence:

$$\begin{aligned}
& \neg \blacksquare a \leq \diamond \neg a = \bigwedge \{b \mid \neg a \prec b\} \\
& \text{iff } \neg \blacksquare a \leq b \text{ for all } b \in A \text{ s.t. } \neg a \prec b \\
& \text{iff } \neg b \leq \blacksquare a \text{ for all } b \in A \text{ s.t. } a \prec b \quad \neg \text{ antitone and involutive} \\
& \text{iff } \neg b \prec a \text{ for all } b \in A \text{ s.t. } \neg a \prec b \quad \text{Lemma 1 (iii), (SI), (UD)} \\
\\
& \bigvee \{a \mid a \prec \neg b\} = \blacksquare \neg b \leq \neg \diamond b \\
& \text{iff } a \leq \neg \diamond b \text{ for all } b \in A \text{ s.t. } a \prec \neg b \\
& \text{iff } \diamond b \leq \neg a \text{ for all } b \in A \text{ s.t. } a \prec \neg b \quad \neg \text{ antitone and involutive} \\
& \text{iff } b \prec \neg a \text{ for all } b \in A \text{ s.t. } a \prec \neg b \quad \text{Lemma 1 (ii), (WO), (DD)}
\end{aligned}$$

(vii) (a) From left to right, let $a, b \in A$. If $\prec^{-1}[b] = \emptyset$, then $\blacksquare b = \bigvee \emptyset = \perp$, hence $\blacksquare(a \vee \blacksquare b) = \blacksquare(a \vee \perp) = \blacksquare a = \blacksquare a \vee \perp = \blacksquare a \vee \blacksquare b$, as required. If $\prec^{-1}[b] \neq \emptyset$, then by definition, $a \vee \blacksquare b = \bigvee \{a \vee e \mid e \prec b\} \in O(A^\delta)$ since, by (UD), $\{a \vee e \mid e \prec b\}$ is up-directed. Hence:

$$\begin{aligned}
\blacksquare(a \vee \blacksquare b) &= \bigvee \{\blacksquare d \mid d \leq a \vee \blacksquare b\} && \text{definition of } \blacksquare \text{ on } O(A^\delta) \\
&= \bigvee \{c \mid \exists d (c \prec d \text{ \& } d \leq a \vee \blacksquare b)\} && \text{definition of } \blacksquare \text{ on } A \\
\\
\blacksquare a \vee \blacksquare b &= \bigvee \{y \mid y \prec a\} \vee \bigvee \{z \mid z \prec b\} \\
&= \bigvee \{y \vee z \mid y \prec a \text{ and } z \prec b\}.
\end{aligned}$$

Hence, to show that $\blacksquare(a \vee \blacksquare b) \leq \blacksquare a \vee \blacksquare b$, it is enough to show that if $c \in A$ is s.t. $\exists d (c \prec d \text{ \& } d \leq a \vee \blacksquare b)$, then $c \leq y \vee z$ for some $y, z \in A$ s.t. $y \prec a$ and $z \prec b$. By compactness, $d \leq a \vee \blacksquare b$ implies that $d \leq a \vee e$ for some $e \in A$ s.t. $e \prec b$.

By (WO), $c \prec d \leq a \vee e$ implies $c \prec a \vee e$. Summing up, $\exists e(e \prec b \ \& \ c \prec a \vee e)$. Hence, by (S9 $\Rightarrow$), $\exists a' \exists b'(a' \prec a \ \& \ b' \prec b \ \& \ c \leq a' \vee b')$, which is the required condition for $y := a'$ and $z := b'$.

Conversely, let $x, a, b \in A$ s.t. $c \prec b$ and $x \prec a \vee c$ for some $c \in A$. By Lemma 1(i), $x \prec a \vee c$ implies that $x \leq \blacksquare(a \vee c)$, and $c \prec b$ implies that $c \leq \blacksquare b$. Hence, the monotonicity of $\blacksquare$ (which is guaranteed by (WO), cf. Lemma 3(ii)(a)) and the assumption imply that the following chain of inequalities holds: $x \leq \blacksquare(a \vee c) \leq \blacksquare(a \vee \blacksquare b) \leq \blacksquare a \vee \blacksquare b = (\bigvee \prec^{-1}[a]) \vee (\bigvee \prec^{-1}[b]) = \bigvee \{a' \vee b' \mid a' \prec a \text{ and } b' \prec b\}$.

We claim that $U := \{a' \vee b' \mid a' \prec a \text{ and } b' \prec b\}$ is up-directed: indeed, if $a'_i \vee b'_i \in U$ for $1 \leq i \leq 2$, then $a'_i \prec a$ and $b'_i \prec b$ imply by (UD) that $a' \prec a$ and $b' \prec b$ for some $a', b' \in A$ s.t. $a'_i \leq a'$ and $b'_i \leq b'$, hence $a'_i \vee b'_i \leq a' \vee b' \in U$. Hence, by compactness, $x \leq a' \vee b'$ for some $a', b' \in A$ s.t. $a' \prec a$ and $b' \prec b$, as required.

(vii) (b) From left to right, let $a, b \in A$. By the definitions spelled out in the proof of (vii) (a), to show that $\blacksquare a \vee \blacksquare b \leq \blacksquare(a \vee \blacksquare b)$, it is enough to show that for all $x, y \in A$, if $y \prec a$ and $z \prec b$, then $y \vee z \prec d$ for some $d \in A$ s.t. $d \leq a \vee e$ for some $e \in A$ s.t. $e \prec b$. By (S9 $\Leftarrow$), $y \vee z \prec a \vee c$ for some $c \in A$ s.t. $c \prec b$. Then the statement is verified for $d := a \vee c$ and $e := c$.

Conversely, let $a, b, x \in A$ s.t. $a' \prec a$, $b' \prec b$ and $x \leq a' \vee b'$ for some $a', b' \in A$, and let us show that $x \prec a \vee c$ for some $c \in A$ s.t. $c \prec b$. By Lemma 1 (i), the assumptions imply that the following chain of inequalities holds: $x \leq a' \vee b' \leq \blacksquare a \vee \blacksquare b \leq \blacksquare(a \vee \blacksquare b) = \bigvee \{e \mid \exists d(e \prec d \ \& \ d \leq a \vee \blacksquare b)\}$, the last identity being discussed in the proof of (vii) (a).

We claim that the set $U := \{e \mid \exists d(e \prec d \ \& \ d \leq a \vee \blacksquare b)\}$ is up-directed: indeed, if $e_i \prec d_i$ for some $d_i \in A$ s.t. $d_i \leq a \vee \blacksquare b$ where $1 \leq i \leq 2$, then by (WO), $e_i \prec d_i \leq d_1 \vee d_2$ implies $e_i \prec d_1 \vee d_2$, hence by (UD), $e \prec d_1 \vee d_2$ for some $e \in A$ s.t. $e_i \leq e$, and finally, $e \in U$, its witness being $d := d_1 \vee d_2 \leq a \vee \blacksquare b$.

Hence, by compactness, $x \leq e$ for some $e \in A$ s.t. $e \prec d$ for some $d \in A$ s.t. $d \leq a \vee \blacksquare b = \bigvee \{a \vee c \mid c \prec b\}$. Again by compactness (which is applicable because, as discussed in the proof of (vii)(a), $\{a \vee c \mid c \prec b\}$ is up-directed), $d \leq a \vee c$ for some $c \in A$ s.t. $c \prec b$. Hence, by (WO) and (SI), $x \leq e \prec d \leq a \vee c$ implies $x \prec a \vee c$, as required.

(vii)(c) From left to right, let $a, b \in A$. By definition, $\blacksquare a \vee \blacksquare b = \bigvee \{x \vee y \mid x \prec a \text{ and } y \prec b\} \in O(A^\delta)$, since $\{x \vee y \mid x \prec a \text{ and } y \prec b\}$ is up-directed, as discussed in the proof of (vii)(a). Then,

$$\begin{aligned} \blacksquare(\blacksquare a \vee \blacksquare b) &= \bigvee \{\blacksquare c \mid c \leq \blacksquare a \vee \blacksquare b\} && \text{definition of } \blacksquare \text{ on } O(A^\delta) \\ &= \bigvee \{d \mid \exists c(d \prec c \ \& \ c \leq \blacksquare a \vee \blacksquare b)\}. && \blacksquare c := \bigvee \{d \mid d \prec c\} \end{aligned}$$

Hence, to prove that $\bigvee \{d \mid d \prec a \vee b\} = \blacksquare(a \vee b) \leq \blacksquare(\blacksquare a \vee \blacksquare b)$, it is enough to show that if $d \prec a \vee b$, then $d \prec c$ for some $c \in A$ s.t. $c \leq \blacksquare a \vee \blacksquare b$. By (SL1), $d \prec a \vee b$ implies that $d \prec a' \vee b'$ for some $a', b' \in A$ s.t. $a' \prec a$ and $b' \prec b$. Since $a' \leq \blacksquare a$ and $b' \leq \blacksquare b$, The statement is verified for $c := a' \vee b'$.

Conversely, let $a, b, c \in A$ s.t. $c \prec a \vee b$, and let us show that $c \prec a' \vee b'$ for some $a', b' \in A$ s.t. $a' \prec a$ and $b' \prec b$. From $c \prec a \vee b$ it follows that $c \leq \blacksquare(a \vee b) \leq \blacksquare(\blacksquare a \vee \blacksquare b) = \bigvee \{d \mid \exists e(d \prec e \ \& \ e \leq \blacksquare a \vee \blacksquare b)\}$.

We claim that $U := \{d \mid \exists e(d \prec e \ \& \ e \leq \blacksquare a \vee \blacksquare b)\}$ is up-directed: indeed, if $d_i \prec e_i$ and $e_i \leq \blacksquare a \vee \blacksquare b$, then $e_1 \vee e_2 \leq \blacksquare a \vee \blacksquare b$, and by (WO), $d_i \prec e_i \leq e_1 \vee e_2$ implies that $d_i \prec e_1 \vee e_2$, hence, by (UD), $d \prec e_1 \vee e_2$ for some $d \in A$ s.t. $d_i \leq d$, and $d \in U$, its witness being $e := e_1 \vee e_2$.

Hence, by compactness, $c \leq d$ for some $d \in A$ s.t. $d \prec e$ for some $e \in A$ s.t. $e \leq \blacksquare a \vee \blacksquare b = \bigvee \{x \vee y \mid x \prec a \text{ and } y \prec b\}$. Since, as discussed above, $\{x \vee y \mid x \prec a \text{ and } y \prec b\}$ is up-directed, by compactness, $e \leq a' \vee b'$ for some $a', b' \in A$ s.t. $a' \prec a$ and $b' \prec b$. By (SI) and (WO), $c \leq d \prec e \leq a' \vee b'$ implies $c \prec a' \vee b'$, as required.

4 Applications

In the present section, we discuss two independent but connected ways of using the characterization results of the previous section. Firstly, the output operators out_i^N for $1 \leq i \leq 4$ associated with a given input/output logic $\mathbb{L} = (\mathcal{L}, N)$ can be given semantic counterparts in the environment of proto-subordination algebras as follows: for every proto-subordination algebra $\mathbb{S} = (A, \prec)$, we let $\mathbb{S}_i := (A, \prec_i)$ where $\prec_i \subseteq A \times A$ is the smallest extension of $\prec$ which satisfies the properties indicated in the following table:

$\prec_i$ Properties
$\prec_1$ ($\top$), (SI), (WO), (AND)
$\prec_2$ ($\top$), (SI), (WO), (AND), (OR)
$\prec_3$ ($\top$), (SI), (WO), (AND), (CT)
$\prec_4$ ($\top$), (SI), (WO), (AND), (OR), (CT)

Then, for each $1 \leq i \leq 4$, and every $B \subseteq A$, if $k = \bigwedge B \in K(A^\delta)$, then⁴

$$\diamond_i^\sigma k := \bigwedge \{\prec_i[a] \mid a \in A \text{ and } k \leq a\}$$

encodes the algebraic counterpart of $out_i^N(\Gamma)$ for any $\Gamma \subseteq \text{Fm}$, and the characteristic properties of $\diamond_i$ for each $1 \leq i \leq 4$ are those identified in Lemma 4, Corollary 1, and Proposition 2. For any directed proto-subordination algebra $\mathbb{S} = (A, \prec)$, let $\mathbb{S}_i^* := (A, \diamond_i, \blacksquare_i)$ denote the slanted algebras associated with $\mathbb{S}_i = (A, \prec_i)$ for each $1 \leq i \leq 4$.

Proposition 3. *For any directed proto-subordination algebra $\mathbb{S} = (A, \prec)$,*

- (i) $\diamond_1$ (resp. $\diamond_2$) is the largest monotone (resp. regular) map dominated by $\diamond$ (i.e. pointwise-smaller than or equal to $\diamond$), and $\blacksquare_1$ (resp. $\blacksquare_2$) is the largest monotone (resp. regular) map dominated by $\blacksquare$.

⁴ When $\mathcal{L}$ does not have the conjunction property, this construction works only under the additional assumption that B is down-directed; however, in most common cases (e.g. when $\mathbb{S}$ is lattice-based) this assumption is not needed.

- (ii) $\diamond_3$ (resp. $\diamond_4$) is the largest monotone (resp. regular) map satisfying $\diamond a \leq \diamond(a \wedge \diamond a)$ dominated by $\diamond$, and $\blacksquare_3$ (resp. $\blacksquare_4$) is the largest monotone (resp. regular) map satisfying $\blacksquare(a \vee \blacksquare a) \leq \blacksquare a$ dominated by $\blacksquare$.

Proof. By Lemma 4 and Proposition 2, the properties stated in each item of the statement hold for $\diamond_i$ and $\blacksquare_i$. To complete the proof, we need to argue for $\diamond_i$ being the largest such map (the proof for $\blacksquare_i$ is similar). By Lemma 1 (ii), $a \prec_i b$ iff $\diamond_i a \leq b$ for all $a, b \in A$ and $1 \leq i \leq 4$. Any $f : A \rightarrow A^\delta$ s.t. $f(a) \in K(A^\delta)$ for every $a \in A$ induces a proto-subordination relation $\prec_f \subseteq A \times A$ defined as $a \prec_f b$ iff $f(a) \leq b$. Clearly, if $f(a) \leq f'(a)$ for every $a \in A$, then $\prec_{f'} \subseteq \prec_f$. Moreover, if $f(a) < f'(a)$, then, by denseness, $f(a) \leq b$ for some $b \in A$ s.t. $f'(a) \not\leq b$, hence $\prec_{f'} \subset \prec_f$.

If $\diamond_i$ is not the largest map endowed with the properties mentioned in the statement and dominated by $\diamond$, then a map f exists which is endowed with these properties such that $\diamond_i a \leq f(a) \leq \diamond a$ for all $a \in A$, and $\diamond_i b < f(b)$ for some $b \in A$. Then, by the argument in the previous paragraph, $\prec = \prec_\diamond \subseteq \prec_f \subset \prec_{\diamond_i} = \prec_i$. As f is endowed with the the properties mentioned in the statement, $\prec_f$ is an extension of $\prec$ which enjoys the required properties, and is strictly contained in $\prec_i$. Hence, $\prec_i$ is not the smallest such extension.

As to the second application, in [8], Celani introduces an expansion of Priestley's duality for bounded distributive lattices to *subordination lattices*, i.e. tuples $\mathbb{S} = (A, \prec)$ such that A is a distributive lattice and $\prec \subseteq A \times A$ is a subordination relation.⁵ The dual structure of any subordination lattice $\mathbb{S} = (A, \prec)$ is referred to as the (*Priestley*) *subordination space* of $\mathbb{S}$, and is defined as $\mathbb{S}_* := (X(A), R_\prec)$, where $X(A)$ is (the Priestley space dual to A , based on) the set of prime filters of A , and $R_\prec \subseteq X(A) \times X(A)$ is defined as follows: for P, Q prime filters of A ,

$$(P, Q) \in R_\prec \quad \text{iff} \quad \prec[P] := \{x \in A \mid \exists a(a \in P \ \& \ a \prec x)\} \subseteq Q.$$

Up to isomorphism, we can equivalently define the subordination space of $\mathbb{S}$ as follows:

Definition 9. *The subordination space associated with a subordination lattice $\mathbb{S} = (A, \prec)$ is $\mathbb{S}_* := (J^\infty(A^\delta), R_\prec)$, where $J^\infty(A^\delta)$ is the set of the completely join-irreducible elements of A^δ , and $R_\prec \subseteq J^\infty(A^\delta) \times J^\infty(A^\delta)$ such that $(j, i) \in R_\prec$ iff $i \leq \diamond j$.*

Lemma 6. *For any subordination lattice $\mathbb{S} = (A, \prec)$, the subordination spaces $\mathbb{S}_*$ given according to the two definitions above are isomorphic.*

Proof. As is well known, in the canonical extension A^δ of any distributive lattice A , the set $J^\infty(A^\delta)$ of the completely join-irreducible elements of A^δ coincides with the set of its completely join-prime elements, which are in dual order-isomorphism with the prime filters of A . Specifically, if $P \subseteq A$ is a prime filter,

⁵ In the terminology of the present paper, subordination lattices are subordination algebras based on bounded distributive lattices (cf. Definition 3).

then $j_P := \bigwedge P \in K(A^\delta)$ is a completely join-prime element of A^δ ; conversely, if j is a completely join-prime element of A^δ , then $P_j := \{a \in A \mid j \leq a\}$ is a prime filter of A . Clearly, $j = \bigwedge P_j = j_{P_j}$ for any $j \in J^\infty(A^\delta)$; moreover, it is easy to show, by applying compactness, that $P_{j_P} = \{a \in A \mid \bigwedge P \leq a\} = P$ for any prime filter P of A .

To complete the proof and show that the two relations $R_{\prec}$ can be identified modulo the identifications above, it is enough to show that $\prec[P] \subseteq Q$ iff $\bigwedge Q \leq \bigwedge \prec[P]$ for all prime filters P and Q of A . Clearly, $\prec[P] \subseteq Q$ implies $\bigwedge Q \leq \bigwedge \prec[P]$. Conversely, if $b \in \prec[P]$, then $\bigwedge Q \leq \bigwedge \prec[P] \leq b$, hence, by compactness and Q being an up-set, $b \in Q$, as required.

In [8], some properties of subordination lattices are dually characterized in terms properties of their associated subordination spaces, including those listed in the following proposition, which can be obtained as consequences of the dual characterizations in Proposition 2, slanted canonicity [14], and correspondence theory for distributive modal logic [10].

Proposition 4 (cf. [8], Theorem 5.7). *For any subordination lattice $\mathbb{S}$,*

- (i) $\mathbb{S} \models \prec \subseteq \leq$ iff $R_{\prec}$ is reflexive;
- (ii) $\mathbb{S} \models (D)$ iff $R_{\prec}$ is transitive, i.e. $R_{\prec} \circ R_{\prec} \subseteq R_{\prec}$;
- (iii) $\mathbb{S} \models (T)$ iff $R_{\prec}$ is dense, i.e. $R_{\prec} \subseteq R_{\prec} \circ R_{\prec}$;
- (iv) $\mathbb{S} \models (a = \perp) \vee (\blacksquare a \neq \perp)$ iff $R_{\prec}$ is proper.

Proof. We will give proof only of (i), since the other arguments are similar. By Proposition 2 (i), $\mathbb{S} \models \prec \subseteq \leq$ iff $\mathbb{S}^* \models a \leq \Diamond a$; the inequality $a \leq \Diamond a$ is analytic inductive (cf. [21, Definition 55]), and hence *slanted* canonical by [14, Theorem 4.1]. Hence, from $\mathbb{S}^* \models a \leq \Diamond a$ it follows that $(\mathbb{S}^*)^\delta \models a \leq \Diamond a$, where $(\mathbb{S}^*)^\delta$ is a *standard* (perfect) distributive modal algebra. By algorithmic correspondence theory for distributive modal logic (cf. [10, Theorems 8.1 and 9.8]), $(\mathbb{S}^*)^\delta \models a \leq \Diamond a$ iff $(\mathbb{S}^*)^\delta \models \forall \mathbf{j} (\mathbf{j} \leq \Diamond \mathbf{j})$ where $\mathbf{j}$ ranges in the set $J^\infty((\mathbb{S}^*)^\delta)$. By Definition 9, this is equivalent to $R_{\prec}$ being reflexive.

Likewise, items (iv), (vi), and (vii) of Proposition 2 can be used to extend Celani's results and provide relational characterizations, on subordination spaces, of conditions (CT), (S9), (SL1), (SL2), noticing that the modal inequalities corresponding to those conditions are all analytic inductive.

Proposition 5. *For any subordination lattice $\mathbb{S}$,*

- (i) $\mathbb{S} \models (CT)$ iff $jR_{\prec}i$ implies $jR_{\prec}k, kR_{\prec}i$, for some $k \leq j$.
- (ii) $\mathbb{S} \models (S9)$ iff $i_3R_{\prec}i_1, i_3R_{\prec}i_2 \Leftrightarrow (\exists j \leq i_1) jR_{\prec}i_2, i_3R_{\prec}j$.
- (iii) $\mathbb{S} \models (SL1)$ iff $i_4R_{\prec}i_1, i_4R_{\prec}i_2, i_3R_{\prec}i_4 \Rightarrow (\exists j \leq i_1 \wedge i_2) i_3R_{\prec}j$.
- (iv) $\mathbb{S} \models (SL2)$ iff $i_1R_{\prec}i_4, i_2R_{\prec}i_4, i_4R_{\prec}i_3 \Rightarrow (\exists j \leq i_1 \wedge i_2) jR_{\prec}i_3$.

5 Conclusions and Related Works

We have established a novel connection between the research fields of subordination algebras and of input/output logics. The present paper focuses only on conditional obligations; however, similarly to the duality between box and diamond operators in modal logic, conditional permission (aka negative permission) has been introduced and analysed by Makinson and van der Torre as the dual concept of conditional obligation [27]. In future work, we will study conditional permission in the environment of pre-contact algebras [15], algebraic structures defined dually to subordination algebras. Another interesting future direction concerns extending the present framework to constrained input/output logic [26, 28, 31], which are used for handling contrary-to-duty reasoning and also applied in the formalization of normative systems [1] in philosophical logic and AI [26]. We will also investigate the connections between our approach and joining-systems [24], an algebraic formalization of normative reasoning alternative to input/output logic.

We have presented a bi-modal characterization of input/output logic in the context of selfextensional logics, a class of logics defined in terms of minimal properties, which are satisfied both by classical propositional logic and by the best known nonclassical logics. The present approach is different from other modal formulations of input/output logic [25, 29, 35], also on a non-classical propositional base, in that the output operators themselves are semantically characterized as (suitable restrictions of) modal operators, and their properties characterized in terms of modal axioms (inequalities).

Legal Informatics has recently received a lot of attention from industry and institutions due to the rise of RegTech and FinTech. The input/output logic is expressive enough to support reasoning about constitutive, regulative and defeasible rules; these notions play an important role in the legal domains [5]. For example, reified input/output logic is a suitable formalism for expressing legal statements like those in the General Data Protection Regulation, for more details see the DAPRECO knowledge base [32]. There are several active projects for implementing input/output reasoners [3, 17, 23, 33]. One of the current challenges is scalability of legal (and I/O) reasoners. Subordination algebras and its correspondence theorems in first-order logic can be used to understand algorithmic correspondence of input/output operations in first-order logic for designing scalable I/O reasoners, for example see [2], for legal applications.

References

1. Alchourrón, C., Bulygin, E.: Normative Systems. Springer, Wien, New York (1971)
2. Bellomarini, L., Benedetto, D., Gottlob, G., Sallinger, E.: Vatalog: a modern architecture for automated reasoning with large knowledge graphs. *Inf. Syst.* **105**, 101528 (2020)
3. Benzmüller, C., Farjami, A., Meder, P., Parent, X.: I/O logic in HOL. *J. Appl. Log. - IfCoLoG J. Log. Appl. (Spec. Issue Reason. Legal AI)* **6**(5), 715–732 (2019)

4. Bezhanishvili, G., Bezhanishvili, N., Sourabh, S., Venema, Y.: Irreducible equivalence relations, Gleason spaces, and de Vries duality. *Appl. Categorical Struct.* **25**(3), 381–401 (2017)
5. Boella, G., van der Torre, L.: Regulative and constitutive norms in normative multiagent systems. In: *Proceedings of 9th International Conference on the Principles of Knowledge Representation and Reasoning (KR 2004)*. *Proceedings of 9th International Conference on the Principles of Knowledge Representation and Reasoning (KR 2004)*, vol. 4, pp. 255–265. AAAI Press (2004)
6. Celani, S.: Quasi-modal algebras. *Math. Bohemica* **126**(4), 721–736 (2001)
7. Celani, S.: Precontact relations and quasi-modal operators in Boolean algebras. *ACTAS DEL XIII CONGRESO DR. ANTONIO A. R. MONTEIRO XIII*, pp. 63–79 (2016)
8. Celani, S.: Subordinations on bounded distributive lattices. *Order* (2022). <https://doi.org/10.1007/s11083-021-09580-5>
9. Conradie, W., Ghilardi, S., Palmigiano, A.: Unified correspondence. In: Baltag, A., Smets, S. (eds.) *Johan van Benthem on Logic and Information Dynamics*. OCL, vol. 5, pp. 933–975. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06025-5_36
10. Conradie, W., Palmigiano, A.: Algorithmic correspondence and canonicity for distributive modal logic. *Ann. Pure Appl. Log.* **163**(3), 338–376 (2012)
11. Conradie, W., Palmigiano, A.: Algorithmic correspondence and canonicity for non-distributive logics. *Ann. Pure Appl. Log.* **170**(9), 923–974 (2019)
12. Conradie, W., Palmigiano, A.: Constructive canonicity of inductive inequalities. *Log. Methods Comput. Sci.* **16**, 8:1–8:39 (2020)
13. De Rudder, L., Hansoul, G., Stetenfeld, V.: Subordination algebras in modal logic. *arXiv preprint arXiv:2004.14919* (2020)
14. De Rudder, L., Palmigiano, A.: Slanted canonicity of analytic inductive inequalities. *ACM Trans. Comput. Log. (TOCL)* **22**(3), 1–41 (2021)
15. Dimov, G., Vakarelov, D.: Topological representation of precontact algebras. In: MacCaull, W., Winter, M., Düntsch, I. (eds.) *RelMiCS 2005*. LNCS, vol. 3929, pp. 1–16. Springer, Heidelberg (2006). https://doi.org/10.1007/11734673_1
16. Dunn, J.M., Gehrke, M., Palmigiano, A.: Canonical extensions and relational completeness of some substructural logics. *J. Symb. Log.* **70**(3), 713–740 (2005)
17. Farjami, A.: Discursive input/output logic: deontic modals, and computation. Ph.D. thesis, University of Luxembourg, Luxembourg (2020)
18. Farjami, A.: New algebraic normative theories for ethical and legal reasoning in the LogiKey framework. *arXiv preprint arXiv:2107.11838* (2021)
19. Font, J.M., Jansana, R., Pigozzi, D.: A survey of abstract algebraic logic. *Stud. Log.* **74**(1), 13–97 (2003)
20. Gabbay, D., Parent, X., van der Torre, L.: A geometrical view of I/O logic. *arXiv preprint arXiv:1911.12837* (2019)
21. Greco, G., Ma, M., Palmigiano, A., Tzimoulis, A., Zhao, Z.: Unified correspondence as a proof-theoretic tool. *J. Log. Comput.* **28**(7), 1367–1442 (2018)
22. Jansana, R., Palmigiano, A.: Referential semantics: duality and applications. *Rep. Math. Log.* **41**, 63–93 (2006)
23. Lellmann, B.: From input/output logics to conditional logics via sequents – with provers. In: Das, A., Negri, S. (eds.) *TABLEAUX 2021*. LNCS (LNAI), vol. 12842, pp. 147–164. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86059-2_9
24. Lindahl, L., Odelstad, J.: The theory of joining-systems. In: Gabbay, D., Horty, J., Parent, X., van der Meyden, R., van der Torre, L. (eds.) *Handbook of Deontic Logic and Normative Systems*, vol. 1, pp. 545–634. College Publications (2013)

25. Makinson, D., van der Torre, L.: Input/output logics. *J. Philos. Log.* **29**(4), 383–408 (2000)
26. Makinson, D., van der Torre, L.: Constraints for input/output logics. *J. Philos. Log.* **30**(2), 155–185 (2001)
27. Makinson, D., van der Torre, L.: Permission from an input/output perspective. *J. Philos. Log.* **32**(4), 391–416 (2003)
28. Olszewski, M., Parent, X., van der Torre, L.: Input/output logic with a consistency check—the case of permission. In: Liu, F., Marra, A., Portner, P., Putte, F.V.D. (eds.) *Deontic Logic and Normative Systems – 15th International Conference, DEON 2020/2021, Munich, Germany, 21–24 July 2021*, pp. 358–375. College Publications (2021)
29. Parent, X.: A modal translation of an intuitionistic I/O operation. *J. Appl. Log. - IfCoLoG J. Log. Appl. (Spec. Issue Intuition. Modal Log. Appl.)* **8**(8), 2350–2362 (2021)
30. Parent, X., Gabbay, D., Torre, L.: Intuitionistic basis for input/output logic. In: Hansson, S.O. (ed.) *David Makinson on Classical Methods for Non-Classical Problems*. OCL, vol. 3, pp. 263–286. Springer, Dordrecht (2014). https://doi.org/10.1007/978-94-007-7759-0_13
31. Parent, X., van der Torre, L.: I/O logics with a consistency check. In: Broersen, J., Condoravdi, C., Nair, S., Pigozzi, G. (eds.) *Deontic Logic and Normative Systems – 14th International Conference, DEON 2018, Utrecht, The Netherlands, 3–6 July 2018*, pp. 285–299. College Publications (2018)
32. Robaldo, L., Bartolini, C., Palmirani, M., Rossi, A., Martoni, M., Lenzini, G.: Formalizing GDPR provisions in reified I/O logic: the DAPRECO knowledge base. *J. Log. Lang. Inf.* **29**, 401–449 (2019)
33. Steen, A.: Goal-directed decision procedures for input/output logics. In: *Deontic Logic and Normative Systems: 15th International Conference (DEON 2020/2021)*, pp. 1–13. College Publications (2021)
34. Stolpe, A.: A concept approach to input/output logic. *J. Appl. Log.* **13**(3), 239–258 (2015)
35. Straßer, C., Beirlaen, M., van de Putte, F.: Adaptive logic characterizations of input/output logic. *Stud. Log.* **104**(5), 869–916 (2016)
36. Sun, X.: Proof theory, semantics and algebra for normative systems. *J. Log. Comput.* **28**(8), 1757–1779 (2018)



Material Dialogues for First-Order Logic in Constructive Type Theory

Dominik Wehr¹(✉) and Dominik Kirst²

¹ University of Gothenburg, Gothenburg, Sweden
dominik.wehr@gu.se

² Saarland Informatics Campus, Saarland University, Saarbrücken, Germany

Abstract. Material dialogues are turn taking games which model debates about the satisfaction of logical formulas. A novel variant played over first-order structures gives rise to a notion of first-order satisfaction. We study the induced notion of validity for classical and intuitionistic first-order logic in the constructive setting of the calculus of inductive constructions. We prove that such material dialogue semantics for classical first-order logic admits constructive soundness and completeness proofs, setting it apart from standard model theoretic semantics of first-order logic. Furthermore, we prove that completeness with regards to intuitionistic material dialogues fails in constructive and classical settings. The results concerning classical material dialogues have been mechanized using the Coq interactive theorem prover.

Keywords: Dialogue Games · Game Semantics · First-Order Logic · Constructive Type Theory

1 Introduction

Logical dialogues were introduced by Paul Lorenzen [12, 13], a philosopher and constructive mathematician active throughout the latter half of the twentieth century. They are a result of his search for a constructively acceptable account of mathematics, beginning with his work on operative mathematics [11]. Logical dialogues are turn-taking games which model a debate in which the proponent defends the validity of a formula against the criticisms of an opponent. The games' moves are modeled after speech acts: asserting formulas and questioning assertions made by the other player. In this, they differ from the more widespread logical games in the style of Hintikka [8], in which a formula is reduced to its atoms by both players, the turn order being determined by the syntactical structure of the formula. Although logical dialogues were initially put forward as a semantics for intuitionistic logic, they can also capture classical logic [10].

Any dialogue begins with the proponent asserting a formula to be discussed. The players then take turns, starting with the opponent. The player at turn can

The research of the first author is supported by the Knut and Alice Wallenberg Foundation [2020.0199] and the Swedish Research Council [2017-05111].

choose between two possible moves: Either they can attack an assertion made by the opposing player or they can defend against such an attack, usually by asserting another formula. Some attacks require the attacking player to assert a formula while carrying out the attack, in which case the assertion in question is called an admission. As an example, attacking the asserted formula $\varphi \rightarrow \psi$ requires the attacker to admit φ while attacking. To defend against the attack on $\varphi \rightarrow \psi$, the attacked player must assert ψ . As in many games, finite plays are won by the player who made the last move. Infinite plays are always won by the opponent. To prevent the opponent from winning by stalling indefinitely, for example through repeated attacks on the proponent's initial assertion, additional restrictions are imposed on the opponent's legal moves. The most common such restriction is to only allow the opponent to react to the proponent's previous move (see Krabbe [9] for alternative restrictions).

While the meaning of logical connectives can be captured by attacks and defenses allowing the players to break down assertions into subformulas, this approach does not extend to atomic formulas. There are two different ways of incorporating atomic formulas in logical dialogues. Material dialogues, the variant originally proposed by Lorenzen [12, 13], permit attacks on atomic formulas. To defend against such an attack, the attacked player is required to demonstrate the validity of the atomic formula. In Lorenzen's original formulation this meant deriving a word according to a grammar, a remainder of his operative semantics of mathematics [11]. Formal dialogues, which were put forward in the dissertation of Lorenzen's student Kuno Lorenz [10], treat atomic formulas without appealing to their "underlying meaning", i.e. by purely syntactic means. In this setting, atomic formulas cannot be attacked by either player and an additional restriction is imposed on the proponent: They may only assert those atomic formulas which the opponent has asserted previously. Historically, the study of logical dialogues after Lorenzen's and Lorenz' initial work has been focused on formal dialogues due to their greater simplicity [9]. Sørensen and Urzyczyn [15] have demonstrated that the winning strategies of formal dialogues for propositional logic are structurally similar to sequent calculus derivations, a result which has been extended to first-order logic in [4].

If one fixes the demonstration method of atomic formulas in a material dialogue to be a demonstration of its satisfaction in a previously agreed upon first-order structure, this induces a model-theoretic notion of satisfaction and, by quantifying over all models, validity. In this article, we study the arising semantics of first-order logic in the constructive setting of the calculus of inductive constructions [2, 14]. This extends our previous investigation [4, 5] into the constructivity of completeness theorems for various semantics of first-order logic, including formal dialogues for intuitionistic first-order logic.

1.1 Outline and Contributions

This section summarizes the article's results. Section 2 covers some basic definitions and results we rely on throughout the article. We close with a brief discussion of various questions arising from this article in Sect. 5. Note that our

use of “soundness” and “completeness” may be somewhat unusual. When speaking of the soundness of a semantics, the intended meaning is the soundness of some suitable deduction system with regards to said semantics (and similarly for “completeness”). While non-standard, this terminology allows us to be more concise about the results derived in this article.

Classical Material Dialogues. In Sect. 3, we define material dialogues for classical first-order logic with all connectives. We prove their soundness with regards to a cut-free, classical sequent calculus. Notably, classical material dialogues are sound on any first-order structure, whereas classical Tarski semantics require the underlying structure to satisfy all instances of the law of the excluded middle (LEM), a property not necessarily held by all structures in a constructive setting. One could thus say that the “classicality” of classical material dialogues is within their rules of engagement, not the underlying structures. We further prove that classical material validity entails exploding classical Tarski validity, a constructively stricter notion than standard classical Tarski validity. We then use the constructive completeness of exploding classical Tarski validity to deduce the same for classical material dialogues. Notably, we obtain completeness for the full syntax of first-order logic. The only analogous result for Tarski semantics we are aware of relies on the full LEM [5]. The results of Sect. 3 have been mechanized in Coq, the mechanization being located at [17].

Intuitionistic Material Dialogues. In Sect. 4 we analyze material dialogues for first-order logic with the usual rules of dialogues of intuitionistic logics. We prove that standard Tarski validity on the fragment $\mathfrak{F}^{\text{d}}$ given below entails intuitionistic material validity.

$$\begin{array}{ll} a, b : \mathfrak{A} :: = \perp \mid P\bar{t} \mid a \wedge b \mid a \vee b \mid \exists x.a & P : \Sigma, \bar{t} : \mathfrak{T}, x : \mathbf{V} \\ \varphi, \psi : \mathfrak{F}^{\text{d}} :: = a \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid a \rightarrow \psi \mid \forall x.\varphi \mid \exists x.\varphi & x : \mathbf{V} \end{array}$$

This means that proving completeness with regards to intuitionistic material dialogues is tantamount to disproving non-constructive principles on the meta-level. As such principles are consistent with most constructive theories, completeness cannot be established without additional axioms. In fact, intuitionistic and classical material dialogues completely coincide under the full LEM. The standard rendition of intuitionistic material dialogues is thus ill-suited as a semantics of intuitionistic first-order logic.

2 Preliminaries

2.1 The Calculus of Inductive Constructions

The results of this article are all derived within the Calculus of Inductive Constructions (CIC) [2, 14], the type theory underlying the interactive theorem prover Coq. The CIC consists of a predicative hierarchy of type universes $\mathbb{T}_i$ above an impredicative universe $\mathbb{P}$ of propositions. Each type universe contains

an empty type, products $A \times B$, sums $A + B$, function types $A \rightarrow B$, dependent products $\Pi a : A. B(a)$ and dependent sums $\Sigma a : A. B(a)$. In $\mathbb{P}$, we denote them by their respective Curry-Howard correspondents $\perp, \wedge, \vee, \rightarrow, \forall, \exists$. Allowing unrestricted elimination from $\mathbb{P}$ into the $\mathbb{T}_i$ results in an inconsistency [6]. However, this restriction can be lifted for some types in $\mathbb{P}$, including types of at most one constructor, such as $\perp$ and the equality type $= : \Pi A. A \rightarrow A \rightarrow \mathbb{P}$ with a sole constructor of type $\forall(a : A). a = a$.

We use inductive types for the natural numbers ($n : \mathbb{N} := 0 \mid Sn$), option types ($\mathcal{O}(A) := \ulcorner a \urcorner \mid \emptyset$) and list types ($l : \mathcal{L}(A) := [] \mid a::l$). We denote *list membership* by $a \in l$ and the *list appending operation* by $l ++ l'$.

2.2 First-Order Predicate Logic

Fix a *signature* Σ of functions symbols f and predicate symbols P , denoting their arities by $|f|$ and $|P|$, respectively. Let $\mathbf{V}$ be the countable type of variables $x, y, z : \mathbf{V}$. The associated *term and formula language* is defined below.

$$\begin{aligned} \vec{x} &:: x \mid f \vec{t} & x &: \mathbf{V}, f : \Sigma, \vec{t} : \vec{x}^{|f|} \\ \varphi : \vec{x} &:: \perp \mid P \vec{t} \mid \varphi \dot{\wedge} \psi \mid \varphi \dot{\vee} \psi \mid \varphi \dot{\rightarrow} \psi \mid \dot{\forall} x. \varphi \mid \dot{\exists} x. \varphi & x &: \mathbf{V}, P : \Sigma, \vec{t} : \vec{x}^{|P|} \end{aligned}$$

To aid in the distinction between meta- and object-level syntax, we write small dots over the connectives of the latter. Negation is defined as $\dot{\neg} \varphi := \varphi \dot{\rightarrow} \perp$. Note that formally, especially within proofs, we are working with de Bruijn binders [1] instead of the syntax with named binders we present here. More details on de Bruijn binders are given in Sect. 2.3. However, for the sake of readability, we opt to present all definitions and theorems in the main text of this article in the familiar style of named binders.

A *structure* $\mathbf{S}$ consists of a type X , a *predicate interpretation* $P^{\mathbf{S}} : X^{|P|} \rightarrow \mathbb{P}$ for each $P : \Sigma$, a *function interpretation* $f^{\mathbf{S}} : X^{|f|} \rightarrow X$ for each $f : \Sigma$ and an *absurdity interpretation* $\perp^{\mathbf{S}} : \mathbb{P}$. A *model* consists of a structure $\mathbf{S}$ together with an *assignment* $\rho : \mathbf{V} \rightarrow X$. The term evaluation function t^ρ in a model $\mathbf{S}, \rho$ is defined as $x^\rho := \rho x$ and $(f \vec{t})^\rho := f^{\mathbf{S}} \vec{t}^\rho$. The Tarski satisfaction relation $\rho \models \varphi$ is defined below. We often write $\mathbf{S}$ for X , e.g. writing $s : \mathbf{S}$ instead of $s : X$.

$$\begin{aligned} \rho \models P \vec{t} &:: \Leftrightarrow P^{\mathbf{S}} \vec{t}^\rho & \rho \models \varphi \dot{\rightarrow} \psi &:: \Leftrightarrow \rho \models \varphi \rightarrow \rho \models \psi \\ \rho \models \varphi \dot{\wedge} \psi &:: \Leftrightarrow \rho \models \varphi \wedge \rho \models \psi & \rho \models \varphi \dot{\vee} \psi &:: \Leftrightarrow \rho \models \varphi \vee \rho \models \psi \\ \rho \models \dot{\forall} x. \varphi &:: \Leftrightarrow \forall s : \mathbf{S}, \rho[x \mapsto s] \models \varphi & \rho \models \dot{\exists} x. \varphi &:: \Leftrightarrow \exists s : \mathbf{S}, \rho[x \mapsto s] \models \varphi \\ \rho \models \perp &:: \Leftrightarrow \perp^{\mathbf{S}} \end{aligned}$$

For a finite context Γ we write $\rho \models \Gamma$ if $\rho \models \varphi$ for all $\varphi \in \Gamma$. A structure $\mathbf{S}$ is *classical* if for all assignments ρ and formulas φ it satisfies the principle of double-negation elimination ($\rho \models \dot{\neg} \dot{\neg} \varphi \dot{\rightarrow} \varphi$). A structure $\mathbf{S}$ is *exploding* if for all assignments ρ and formulas φ it satisfies the principle of explosion ($\rho \models \perp \dot{\rightarrow} \varphi$). A structure is *standard* if $\perp^{\mathbf{S}}$ is contradictory (i.e. $\neg \perp^{\mathbf{S}}$ holds). Observe that all standard structures are exploding. An entailment between a finite context

Γ and a formula φ is *valid in classical exploding structures*, written $\Gamma \models^E \varphi$, if for all classical, exploding structures $\mathbf{S}$ and all assignments ρ it is the case that $\rho \models \Gamma$ entails $\rho \models \varphi$. *Validity of entailments in classical standard structures*, $\Gamma \models^S \varphi$, is defined analogously.

Let $\Gamma \Rightarrow \Delta$ denote derivability in the cut-free sequent calculus for classical first-order logic defined below. The system's presentation is somewhat non-standard as the principal formulas of derivation rules are not removed from Γ and Δ . This is convenient when proving soundness as it parallels the requirements imposed on the proponent when attacking and defending.

$$\begin{array}{c}
\text{Ax} \frac{P\vec{t} \in \Gamma \quad P\vec{t} \in \Delta}{\Gamma \Rightarrow \Delta} \qquad \text{L}\bot \frac{\bot \in \Gamma}{\Gamma \Rightarrow \Delta} \\
\\
\text{L}\rightarrow \frac{\varphi \dot{\rightarrow} \psi \in \Gamma \quad \Gamma \Rightarrow \varphi, \Delta \quad \Gamma, \psi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta} \qquad \text{R}\rightarrow \frac{\varphi \dot{\rightarrow} \psi \in \Delta \quad \Gamma, \psi \Rightarrow \varphi, \Delta}{\Gamma \Rightarrow \Delta} \\
\\
\text{L}\wedge \frac{\varphi \dot{\wedge} \psi \in \Gamma \quad \Gamma, \varphi, \psi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta} \qquad \text{R}\wedge \frac{\varphi \dot{\wedge} \psi \in \Delta \quad \Gamma \Rightarrow \varphi, \Delta \quad \Gamma \Rightarrow \psi, \Delta}{\Gamma \Rightarrow \Delta} \\
\\
\text{L}\vee \frac{\varphi \dot{\vee} \psi \in \Gamma \quad \Gamma, \varphi \Rightarrow \Delta \quad \Gamma, \psi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta} \qquad \text{R}\vee \frac{\varphi \dot{\vee} \psi \in \Delta \quad \Gamma \Rightarrow \varphi, \psi, \Delta}{\Gamma \Rightarrow \Delta} \\
\\
\text{L}\forall \frac{\forall \varphi \in \Gamma \quad \Gamma, \varphi[t] \Rightarrow \Delta}{\Gamma \Rightarrow \Delta} \qquad \text{R}\forall \frac{\forall \varphi \in \Delta \quad \uparrow \Gamma \Rightarrow \varphi, \uparrow \Delta}{\Gamma \Rightarrow \Delta} \\
\\
\text{L}\exists \frac{\exists \varphi \in \Gamma \quad \uparrow \Gamma, \varphi \Rightarrow \uparrow \Delta}{\Gamma \Rightarrow \Delta} \qquad \text{R}\exists \frac{\exists \varphi \in \Delta \quad \Gamma \Rightarrow \varphi[t], \Delta}{\Gamma \Rightarrow \Delta}
\end{array}$$

The following completeness result stems from [7] based on ideas from [16]:

Theorem 1. *In constructive settings, restricting to the $\forall, \dot{\rightarrow}, \bot$ -fragment of first-order logic, $\Gamma \models^E \varphi$ entails $\Gamma \Rightarrow \varphi$.*

2.3 De Bruijn Binders

De Bruijn binders were developed by de Bruijn as part of the implementation of the AUTOMATH theorem prover [1]. They provide a formalism for treating syntax containing binders and substitutions with greater ease than the common “named binders” approach. Given below is the syntax of first-order logic using de Bruijn binders. Note especially the absence of variable names after the quantifiers.

$$\begin{array}{ll}
t : \mathfrak{T}:: = n \mid c\vec{t} & n : \mathbb{N}, c : \Sigma \\
\varphi : \mathfrak{F}:: = \bot \mid P\vec{t} \mid \varphi \dot{\wedge} \psi \mid \varphi \dot{\vee} \psi \mid \varphi \dot{\rightarrow} \psi \mid \forall \varphi \mid \exists \varphi & P : \Sigma
\end{array}$$

In de Bruijn syntax, the variables are represented by natural numbers $n : \mathbb{N}$, called de Bruijn indexes. Such an index references the quantifier binding it by counting the number of quantifiers above it in the syntax tree that need to

be skipped to arrive at the binding quantifier. For an example consider the formula $\forall x(P x \dot{\rightarrow} \forall z.Q x z)$ which is represented by $\forall(P 0 \dot{\rightarrow} \forall Q 1 0)$ in de Bruijn syntax. Figure 1 depicts the references described by the indexes. Observe that the variable x is represented by different indexes depending on its position in the syntax tree.

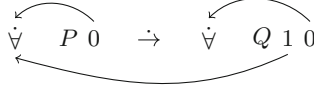


Fig. 1. De Bruijn representation of $\forall x.P x \rightarrow \forall z.Q x z$

When working with de Bruijn binders, most definitions involving logical formulas need to be adjusted slightly. For example, assignments become functions $\rho : \mathbb{N} \rightarrow \mathbf{S}$ mapping indexes to elements of the structure. The usual assignment extension operation $\rho[x \mapsto s]$ is replaced by the operation $s \cdot \rho$ which is defined as below

$$(s \cdot \rho)(n) := \begin{cases} s & n = 0 \\ \rho(m) & n = m + 1 \end{cases}$$

The quantifier rules for Tarski semantics then change appropriately as follows:

$$\rho \models \forall \varphi :\Leftrightarrow \forall s : \mathbf{S}, s \cdot \rho \models \varphi \qquad \rho \models \exists \varphi :\Leftrightarrow \exists s : \mathbf{S}, s \cdot \rho \models \varphi$$

Note that below the quantifier, the indexes of all variables not bound by that quantifier are incremented thus ensuring that each variable refers to the correct element in $s \cdot \rho$.

3 Classical Material Dialogues

We begin by defining formally the material dialogues for classical first-order logic. Fix a standard structure $\mathbf{S}$ the dialogue will be played over. Material dialogues are a turn taking game between two players. The *proponent* tries to defend the satisfaction of some formula in a model, whereas the *opponent* tries to challenge the proponent's claims in such a way that the proponent cannot respond. The dialogues we consider are so-called *E-dialogues* which restrict the opponent to only ever react to the proponent's previous move. It can be shown that the notion of satisfaction induced by E-dialogues is equivalent to that of the more intuitive D-dialogues, in which this restriction is weakened [3, 5].

We model the dialogue game as a state transition system. A triple $(\rho, A, C) : (\mathbf{V} \rightarrow \mathbf{S}) \times \mathcal{L}(\mathfrak{F}) \times \mathcal{L}(\mathbf{A})$ is called a *dialogue state*. Together $\mathbf{S}, \rho$ form the *ambient model*. The list A contains all of the opponent's *assertions* and C records all attacks that the opponent has leveled against the proponent. Each round begins with the proponent making a move, indicated by a transition

$(\rho, A, C) \rightsquigarrow_p (\rho', A', C'); m$ from a dialogue state to a dialogue state and a proponent move $m : \mathbf{M}$. This is followed by an opponent move, indicated by a transition $(\rho', A', C'); m \rightsquigarrow_o (\rho'', A'', C'')$ from a state and proponent move to a further state. We continue by defining the two transition relations $\rightsquigarrow_p$ and $\rightsquigarrow_o$.

The *type D of defenses*, defined below, features three different kinds of defenses: $D_A \varphi$ denotes the act admitting of the formula φ , $D_W \varphi(x) s$ denotes admitting $\varphi(s)$ where $s : \mathbf{S}$. Lastly, $D_M \varphi$ means claiming to be able to demonstrate that φ holds. Note that $D_M \varphi$ is only ever instantiated with atomic φ .

$$\mathbf{D}::= D_A \varphi \mid D_W \varphi(x) s \mid D_M \varphi \quad \varphi : \mathfrak{F}, s : \mathbf{S}$$

The type $\mathbf{A}$ contains all *attacks*. One writes $a \triangleright \varphi$ if $a : \mathbf{A}$ is an *attack on* φ . Each $a : \mathbf{A}$ has an associated set $\mathcal{D}_a$ of *defenses against* a . Both $\mathbf{A}$ and the associated $\mathcal{D}_a$ are laid out below.

$$\begin{array}{ll} A_{\perp} \triangleright \perp & \mathcal{D}_{A_{\perp}} = \{D_M \perp\} \\ A_P \vec{t} \triangleright P \vec{t} & \mathcal{D}_{A_P \vec{t}} = \{D_M P \vec{t}\} \\ A_{\rightarrow} \varphi \psi \triangleright \varphi \dot{\rightarrow} \psi & \mathcal{D}_{A_{\rightarrow} \varphi \psi} = \{D_A \psi\} \\ A_{\vee} \varphi \psi \triangleright \varphi \dot{\vee} \psi & \mathcal{D}_{A_{\vee} \varphi \psi} = \{D_A \varphi, D_A \psi\} \\ A_L \varphi \triangleright \varphi \dot{\wedge} \psi & \mathcal{D}_{A_L \varphi} = \{D_A \varphi\} \\ A_R \psi \triangleright \varphi \dot{\wedge} \psi & \mathcal{D}_{A_R \psi} = \{D_A \psi\} \\ A_s \varphi(x) \triangleright \dot{\vee} x. \varphi(x) & \mathcal{D}_{A_s \varphi(x)} = \{D_W \varphi(x) s\} \\ A_{\exists} \varphi(x) \triangleright \dot{\exists} x. \varphi(x) & \mathcal{D}_{A_{\exists} \varphi(x)} = \{D_W \varphi(x) s \mid s : \mathbf{S}\} \end{array}$$

Some attacks force the attacker to *admit* a formula. This is formalized by a function $\text{adm} : \mathbf{A} \rightarrow \mathcal{O}(\mathfrak{F})$ where $\text{adm } a = \ulcorner \varphi \urcorner$ means that φ must be admitted when attacking with a and $\text{adm } a = \emptyset$ means no admission need be made. The admission obligations are $\text{adm}(A_{\rightarrow} \varphi \psi) = \ulcorner \varphi \urcorner$ and $\text{adm } a = \emptyset$ for all other attacks $a : \mathbf{A}$.

Each round begins by the proponent making a *move* $m : \mathbf{M}$, as detailed below: Either challenging one of the opponent's assertions ($PA a$) or defending against a challenge previously issued by the opponent, either by asserting a formula ($PD \varphi$) or by demonstrating that an atomic formula holds in the ambient model ($PM \varphi$). The function $\text{move} : \mathbf{D} \rightarrow \mathbf{M}$ maps defenses to the proponent move that needs to be made to carry out said defense.

$$m : \mathbf{M} := PA a \mid PD \varphi \mid PM \varphi$$

$$\text{move}(D_A \varphi) = PD \varphi \quad \text{move}(D_W \varphi(x) s) = PD \varphi(x')$$

$$\text{move}(D_M \varphi) = PM \varphi$$

The effect of a proponent's defense d on the game state is defined via a function d^P as below, where x' denotes a variable which does not occur in $A, C, \varphi(x)$.

$$d^P(\rho, A, C) = \begin{cases} (\rho[x' \mapsto s], A, C) & \text{if } d = D_W \varphi(x) s \\ (\rho, A, C) & \text{otherwise} \end{cases}$$

Defenses need to be *justified* by the ambient model. While $D_A \varphi$ and $D_W \varphi s$ are always justified, $D_M \varphi$ requires $\rho \models \varphi$ to hold. All of the previous notations enable a compact definition of the state transitions the proponent may induce by making a move.

$$\text{PA} \frac{\varphi \in A \quad a \triangleright \varphi}{(\rho, A, C) \rightsquigarrow_p (\rho, A, C); PA a} \quad \text{PD} \frac{c \in C \quad d \in \mathcal{D}_c \quad \rho \text{ justifies } d}{(\rho, A, C) \rightsquigarrow_p d^P (\rho, A, C); \text{move } d}$$

The opponent must react to the proponent's previous move. If the proponent defended by asserting a formula, they must issue a new challenge by attacking said assertion (OA). If the proponent attacked one of their assertions, they can either defend against said attack (OD) or counter the attack, attacking the admission made by the proponent while attacking (OC). If the proponent demonstrated the validity of an atomic formula in the ambient model, the opponent cannot respond at all. The operation d^O is defined analogously to d^P and is used to define the transition steps the opponent can induce by making a move. In a slight abuse of notation, we write $c::A$, where c is an attack, for $\psi::A$ if $\text{adm } c = \ulcorner \psi \urcorner$ and A if $\text{adm } c = \emptyset$.

$$d^O (\rho, A, C) = \begin{cases} (\rho, \varphi::A, C) & \text{if } d = D_A \varphi \\ (\rho[x' \mapsto s], \varphi(x')::A, C) & \text{if } (D_W \varphi(x) s) \\ (\rho, A, C) & \text{if } d = D_M \varphi \end{cases}$$

$$\text{OA} \frac{c \triangleright \varphi}{(\rho, A, C); PD \varphi \rightsquigarrow_o (\rho, \psi::A, c::C)}$$

$$\text{OC} \frac{a \triangleright \varphi \quad \text{adm } a = \ulcorner \psi \urcorner \quad \psi \triangleright c}{(\rho, A, C); PA a \rightsquigarrow_o (\rho, c::A, c::C)}$$

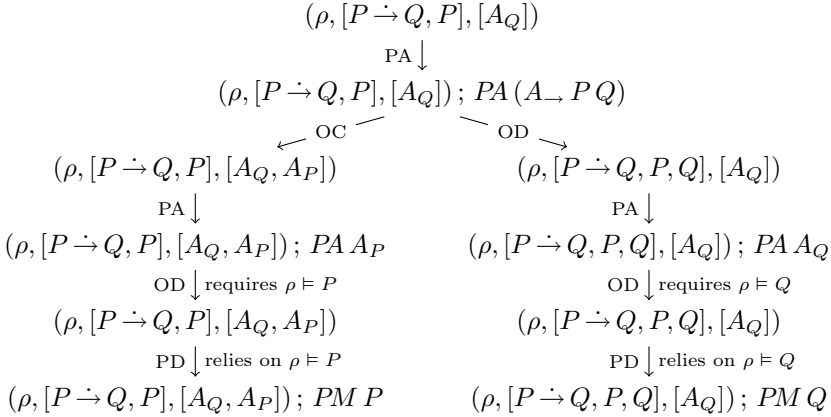
$$\text{OD} \frac{d \in \mathcal{D}_a \quad \rho \text{ justifies } d}{(\rho, A, C); PA a \rightsquigarrow_o d^O (\rho, A, C)}$$

A state can be *won* if the proponent can ensure the play always eventually ends. This is defined as an inductive predicate which is very similar to the inductive well-foundedness predicate commonly used in type theory. In the common parlance of game theory, each derivation of $\text{Win } s$ constitutes a winning strategy for the proponent on state s .

$$\frac{s \rightsquigarrow_p s'; m \quad \forall s''. s'; m \rightsquigarrow_o s'' \rightarrow \text{Win } s''}{\text{Win } s}$$

This notion of winnability extends to formulas φ denoted $\text{Win}(\rho, A, C, \varphi)$ which holds if for all attacks $c \triangleright \varphi$ it is the case that $\text{Win}(\rho, c::A, c::C)$. A formula φ is *dialogically entailed* in a context Γ , written $\Gamma \models^D \varphi$, if for all standard structures $\mathbf{S}$ and assignments $\rho : \mathbf{V} \rightarrow \mathbf{S}$ it is the case that $\text{Win}(\rho, \Gamma, [], \varphi)$.

Example 1. Sketched below is a strategy establishing $\text{Win}(\rho, [P \dot{\rightarrow} Q, P], [A_Q])$ and thus $\text{Win}(\rho, [P \dot{\rightarrow} Q, P], [], Q)$. The proponent first attacks $P \dot{\rightarrow} Q$. In both branches, the opponent can only continue if an atomic formula is satisfied by the ambient model. The proponent can thus carry out a “demonstration” (*PM*) of said atomic formula, winning the play. This strategy can be played on any model $\mathbf{S}, \rho$, meaning it suffices to establish $P \dot{\rightarrow} Q, P \models^D Q$.



The definitions of dialogue games given above were in terms of named binders. When using de Bruijn binders, some definitions require slight adjustments. The attacks and defenses with witnesses need not include the name of the binding variable anymore because it is always identified by the index 0.

$$\begin{array}{ll}
 A_s \varphi \triangleright \dot{\forall} \varphi & \mathcal{D}_{A_s} \varphi = \{D_W \varphi s\} \\
 A_{\exists} \varphi \triangleright \dot{\exists} x. \varphi & \mathcal{D}_{A_{\exists}} \varphi = \{D_W \varphi s \mid s : \mathbf{S}\}
 \end{array}$$

We denote by $\uparrow \varphi$ the operation which increments the index of each free variable occurring in φ , thus freeing the index 0. This operation can be extended to attacks and lists of attacks or formulas in the obvious way. The de Bruijn definitions of d^P and d^O employ this shifting operation to ensure that the index 0 refers to the newly introduced $s : \mathbf{S}$ in the case of $d = D_W \varphi s$.

$$\begin{aligned}
 d^P(\rho, A, C) &= \begin{cases} (s \cdot \rho, \uparrow A, \uparrow C) & \text{if } d = D_W \varphi s \\ (\rho, A, C) & \text{otherwise} \end{cases} \\
 d^O(\rho, A, C) &= \begin{cases} (\rho, \varphi :: A, C) & \text{if } d = D_A \varphi \\ (s \cdot \rho, \varphi :: \uparrow A, \uparrow C) & \text{if } d = D_W \varphi s \\ (\rho, A, C) & \text{if } d = D_M \varphi \end{cases}
 \end{aligned}$$

We continue by proving the classical material dialogues sound with regards to the cut-free classical sequent calculus from Sect. 2.2. This is the easiest soundness result to obtain because of the structural similarity between winning strategies

for dialogues and cut-free sequent calculus derivations demonstrated in [15]. Recall also that the proofs in this article work with the de Bruijn syntax. The results in this section were mechanized in Coq at [17].

The majority of the soundness proof is straightforward. A slight difficulty arises from the differing treatment of quantifiers by material dialogues and sequent calculi. Compare a typical $L\forall$ -rule with the state transition caused by the proponent attacking the admission $\forall x.\varphi(x)$ with $A_{t\rho}\varphi(x)$ and the opponent responding defending, both given below. While the sequent calculus simply instantiates the formula via a substitution $\varphi[t/x]$, material dialogues carry out the instantiation via the assignment ρ .

$$L\forall \frac{\forall x.\varphi \in \Gamma \quad \Gamma, \varphi[t/x] \Rightarrow \Delta}{\Gamma \Rightarrow \Delta} \quad (\rho, \Gamma, C) \rightsquigarrow_{po} (\rho[x' \mapsto t^\rho], \varphi(x')::\Gamma, C)$$

To prove soundness, one needs to show that these two methods of instantiation are “essentially the same”. For this, we introduce *congruence relations* on different aspects of dialogues: Given assignments ρ, ρ' and formulas φ, φ' , we define an equivalence relation $\rho, \varphi \equiv \rho', \varphi'$ which holds if φ and φ' are equal up to term evaluations in the respective assignments. This congruence can be extended to attacks and defenses and it can be shown that these relations do indeed “act as congruences” (see the mechanization for details). The following lemma is crucial to the proof of soundness.

Lemma 1. *Let (ρ, A, C) and (ρ', A', C') be dialogue states such that $\rho, A \equiv \rho', A'$ and $\rho, C \equiv \rho', C'$. If $\text{Win}(\rho, A, C)$ then $\text{Win}(\rho', A', C')$.*

Theorem 2 (Soundness). *Let Γ, φ be such that $\Gamma \Rightarrow \varphi$. Then $\Gamma \models^D \varphi$.*

Proof. For this, it suffices to show that for any standard structure **S**

$$\begin{aligned} \Gamma \Rightarrow \Delta \rightarrow \forall \rho, A, C. (\forall \delta \in \Delta. \exists c \in C. c \triangleright \delta \wedge (\forall \psi. \text{adm } c = \ulcorner \psi \urcorner \rightarrow \psi \in A)) \\ \rightarrow \Gamma \subseteq A \rightarrow \text{Win}(\rho, A, C) \end{aligned}$$

because it allows one to conclude $\text{Win}(\rho, c::\Gamma, [c])$ for any $c \triangleright \varphi$ —and thus $\text{Win}(\rho, \Gamma, [], \varphi)$ —from $\Gamma \Rightarrow \varphi$. The proof proceeds per induction on $\Gamma \Rightarrow \Delta$. Lemma 1 is used in the cases of the quantifier rules. We encourage the curious reader to consult the mechanization [17].

We continue by proving completeness. We first prove that material validity entails validity on all classical, exploding models. This is extended to traditional completeness in Theorem 4 by use of completeness for classical exploding models on the $\forall, \rightarrow, \perp$ -fragment (Theorem 1) and a de Morgan translation.

Lemma 2. *For any Γ and φ , $\Gamma \models^D \varphi$ entails $\Gamma \models^E \varphi$.*

Proof. Fix a classical, exploding structure **S**. We extend Tarski satisfaction to defenses via $\rho \models D_A \varphi :\Leftrightarrow \rho \models \varphi$ and $\rho \models D_W \varphi s :\Leftrightarrow s \cdot \rho \models \varphi$ and $\rho \models D_M \varphi :\Leftrightarrow \rho \models \varphi$. Furthermore, we define an auxiliary predicate $\Gamma \models_\rho \bigvee \mathcal{D}$ on contexts Γ ,

assignments ρ and sets of defenses $\mathcal{D}$ which intuitively states that under the assignment ρ , Γ entails the disjunction of all semantic interpretations of $\mathcal{D}$:

$$\Gamma \models_{\rho} \bigvee \mathcal{D} \Leftrightarrow \rho \models \Gamma \rightarrow \forall \vec{s} : \mathbf{S}, \alpha. (\forall d \in \mathcal{D}. \rho \models d \rightarrow (\vec{s} \cdot \rho) \models \alpha) \rightarrow (\vec{s} \cdot \rho) \models \alpha$$

The proof relies on the two intermediate results below, whose proofs are rather routine. We refer the curious reader to the mechanization [17]. Here, $\hat{\mathbf{S}}$ denotes the standard structure with $\perp^{\hat{\mathbf{S}}} = \perp$ which is otherwise exactly the same as $\mathbf{S}$.

- (1) $\text{Win}(\rho, \Gamma, C)$ in $\hat{\mathbf{S}}$ entails $\Gamma \models_{\rho} \bigvee (\bigcup_{c \in C} \mathcal{D}_c)$ in $\mathbf{S}$
- (2) If $(c::\Gamma) \models_{\rho} \bigvee (\mathcal{D}_c \cup \mathcal{D})$ for all $c \triangleright \varphi$ then $\Gamma \models_{\rho} \bigvee (\{D_A \varphi\} \cup \mathcal{D})$ in $\mathbf{S}$

Now assume $\Gamma \models^D \varphi$ and a classical, exploding $\mathbf{S}, \rho$ with $\rho \models \Gamma$. Per assumption $\text{Win}(\rho, \Gamma, [], \varphi)$ in $\hat{\mathbf{S}}$ meaning $(c::\Gamma) \models_{\rho} \bigvee \mathcal{D}_c$ in $\mathbf{S}$ for all $c \triangleright \varphi$ by (1) and thus $\Gamma \models_{\rho} \bigvee \{D_A \varphi\}$ by (2). By picking φ as α we then obtain $\rho \models \varphi$ in $\mathbf{S}$ as desired.

To conclude completeness, we employ a de Morgan translation from the full syntax of first-order logic into the $\forall, \rightarrow, \perp$ -fragment which is given below:

$$\begin{aligned} \perp^M &:= \perp & (P \vec{s})^M &:= P \vec{s} & (\varphi \wedge \psi)^M &:= \neg(\varphi^M \rightarrow \neg\psi^M) \\ (\varphi \vee \psi)^M &:= \neg\varphi^M \rightarrow \psi^M & (\forall\varphi)^M &:= \forall\varphi^M & (\exists\varphi)^M &:= \neg(\forall\neg\varphi^M) \\ (\varphi \rightarrow \psi)^M &:= \varphi^M \rightarrow \psi^M \end{aligned}$$

Furthermore, a dialogical analogue of cut-admissibility is required to derive completeness on the full syntax. A formula φ *can be cut* if for any $\mathbf{S}, \rho, A, A', C$ with $\text{Win}(\rho, A \multimap \varphi::A', C)$ and $\text{Win}(\rho, A \multimap A', C, \varphi)$ we also have that $\text{Win}(\rho, A \multimap A', C)$. The proofs below rely on the congruence principles Lemma 1. As we believe that spelling out all applications of the principle obscures the simple ideas behind the proof, we opt to leave applications of Lemma 1 implicit where possible. Readers interested in the proofs in full detail may take a look at the Coq mechanization [17]. We prove dialogical cut-admissibility with the help of a lemma:

Lemma 3. *Let φ be such that all formulas of smaller complexity can be cut. Fix $c \triangleright \varphi$ such that $\text{Win}(\rho, A, C \multimap c::C')$ and for all $d \in \mathcal{D}_c$ justified under ρ we have $\text{Win}(d^O(\rho, A, C \multimap C'))$. Then $\text{Win}(\rho, A, C \multimap C')$.*

Proof. We prove a slight generalization: Instead of just $c \triangleright \varphi$, we consider any n and $c \triangleright (\uparrow^n \varphi)$ such that $\text{Win}(\rho, A, C \multimap c::C')$ and for all $d \in \mathcal{D}_c$ justified under ρ we have $\text{Win}(d^O(\rho, A, C \multimap C'))$. We then proceed per induction on $\text{Win}(\rho, A, C \multimap c::C')$ and begin by performing a case distinction on the proponent's move in $\text{Win}(\rho, A, C \multimap c::C')$.

PA: The proponent uses $a \triangleright \psi$ on $\psi \in A$. Then the proponent of $\text{Win}(\rho, A, C \multimap C')$ copies that move. There are two possible opponent responses.

- In the case of $\text{adm } a = \ulcorner \theta \urcorner$, the opponent may counter with some $c' \triangleright \theta$. Then the proponent copies the strategy obtained from the inductive hypothesis upon the same counter.
- The opponent may defend with some $d \in \mathcal{D}_a$. Then the proponent copies the strategy obtained for the inductive hypothesis upon the same defense.

PD: Then there is a $c' \in C \multimap c::C'$ and the proponent defends with some $d \in \mathcal{D}_c$. There are two cases to distinguish:

- $c' \in C \multimap C'$: Then the proponent of $\text{Win}(\rho, A, C \multimap C')$ copies the defense. If d is not $D_M \varphi$ for some φ , then the opponent attacks the formula ψ admitted by d with some $a \triangleright \psi$. The proponent then plays according to the strategy obtained from the inductive hypothesis upon $a \triangleright \psi$.
- $c' = c$: Per assumption we have $\text{Win}(d^O(\rho, A, C \multimap C'))$. Then we perform a case distinction on the form of d .
 - $d = D_M \varphi$: Then $\text{Win}(d^O(\rho, A, C \multimap C')) = \text{Win}(\rho, A, C \multimap C')$ and we are done.
 - $d = D_A \psi$: The assumption thus is $\text{Win}(\rho, \psi::A, C \multimap C')$. From the inductive hypothesis we obtain $\text{Win}(\rho, A, C \multimap C', \psi)$. As $D_A \psi \in \mathcal{D}_c$ and $c \triangleright (\uparrow^n \varphi)$ we know that ψ is of lower complexity than φ , meaning it can be cut and we thus obtain $\text{Win}(\rho, A, C \multimap C')$.
 - $d = D_W \psi$: This case is analogous to that for $d = D_A \psi$ with a few more applications of Lemma 1.

Theorem 3 (Cut-admissibility). *All formulas can be cut.*

Proof. The proof proceeds per induction on formula complexity. Thus pick a φ such that all formulas of lower complexity can be cut. We show that

$$\text{Win}(\rho, A \multimap \uparrow^n \varphi::A', C) \rightarrow \text{Win}(\rho, A \multimap A', C, \uparrow^n \varphi) \rightarrow \text{Win}(\rho, A \multimap A', C)$$

per induction on $\text{Win}(\rho, A \multimap \uparrow^n \varphi::A', C)$ which subsumes the fact that φ can be cut. We perform a case distinction on the proponent move.

PA a : Then the proponent attacks some $\psi \in A \multimap \uparrow^n \varphi::A'$ with $a \triangleright \psi$. We distinguish two cases.

- $\psi \in A \multimap A'$: Then the proponent of $\text{Win}(\rho, A \multimap A', C)$ copies that attack and proceeds per inductive hypothesis.
- $\psi = \uparrow^n \varphi$: Then $\text{Win}(\rho, A \multimap A', C, \uparrow^n \varphi)$ yields $\text{Win}(\rho, A \multimap A', a::C)$ and the inductive hypothesis means that for all $d \in \mathcal{D}_a$ we have that $\text{Win}(d^O(\rho, A \multimap A', C))$. We may thus apply Lemma 3 to deduce $\text{Win}(\rho, A \multimap A', C)$.

PD ψ : Then there is some $c \in C$ and some $d \in \mathcal{D}_c$ such that d results in admitting ψ . The proponent of $\text{Win}(\rho, A \multimap A', C)$ thus copies that admission and proceeds per inductive hypothesis.

PM φ : Then $c \in C$ for the unique $a \triangleright \varphi$ (in either case of $\varphi = \perp$ or $\varphi = P \vec{t}$) and $\rho \models \varphi$ holds. The proponent of $\text{Win}(\rho, A \multimap A', C)$ can thus win as well by demonstrating $\rho \models \varphi$.

Combining Theorem 3 with the de Morgan translation above yields completeness on the full syntax of first-order logic.

Theorem 4 (Completeness). *For any Γ and φ , $\Gamma \models^D \varphi$ entails $\Gamma \Rightarrow \varphi$.*

Proof. First of all, $\Gamma \models^D \varphi$ entails $\Gamma^M \models^D \varphi^M$. This follows from the provability of $\varphi \leftrightarrow \varphi^M$, soundness (Theorem 2) and cut-admissibility for material dialogues (Theorem 3). We can now apply Lemma 2 to obtain $\Gamma^M \models^E \varphi^M$, which entails $\Gamma^M \Rightarrow \varphi^M$ by Theorem 1. It is well known that this entails $\Gamma \Rightarrow \varphi$.

Observe that Theorem 4 was obtained fully constructively. This is noteworthy because similar results often make use of unconstructive principles. For example, the only method of extending completeness from the $\forall, \rightarrow, \perp$ -fragment to the full syntax for classical, standard Tarski models we know of uses LEM [5]. This should be taken as an indication that material dialogues are exceptionally well-suited as a semantics for classical first-order logic in a constructive setting.

4 Intuitionistic Material Dialogues

One of the striking features of dialogue games is that classical dialogue semantics can often be transformed into intuitionistic dialogue games by a simple change in the rules governing the interactions between proponent and opponent [9, 10]: The proponent may only ever defend against the opponent's most recent attack. This is analogous to the restriction to at most one right-hand formula for obtaining intuitionistic sequent calculi. The adjusted version of the proponent move transition relation $\rightsquigarrow_p$ for the arising intuitionistic material dialogues is given below. Note that in this section, in a slight abuse of notation, $\text{Win}(\rho, A, C)$ and $\Gamma \models^D \varphi$ refer to dialogues played according to these intuitionistic rules. Further note that the rule PA remains unchanged when compared to classical material dialogues.

$$\begin{array}{c} \text{PA} \frac{\varphi \in A \quad a \triangleright \varphi}{(\rho, A, C) \rightsquigarrow_p (\rho, A, C); PA a} \\ \\ \text{PD} \frac{d \in \mathcal{D}_c \quad \rho \text{ justifies } d}{(\rho, A, c::C) \rightsquigarrow_p d^P (\rho, A, c::C); \text{move } d} \end{array}$$

This version of intuitionistic material dialogues does not admit a constructive completeness proof. To demonstrate this, we define the following fragment of first-order logic:

$$\begin{array}{l} a, b : \mathfrak{A}:: = \perp \mid P\vec{t} \mid a \dot{\wedge} b \mid a \dot{\vee} b \mid \dot{\exists} x.a \qquad P : \Sigma, \vec{t} : \mathfrak{T}^{|P|} \\ \varphi, \psi : \mathfrak{F}^{\mathfrak{D}}:: = a \mid \varphi \dot{\wedge} \psi \mid \varphi \dot{\vee} \psi \mid a \dot{\rightarrow} \psi \mid \dot{\forall} x.\varphi \mid \dot{\exists} x.\varphi \end{array}$$

$\mathfrak{A}$ is the fragment of first-order logic which allows attacking “blindly”, i.e. the same attack pattern can be used on these formulas in every winning strategy. The fragment $\mathfrak{A}$ does not include $a \dot{\rightarrow} b$ as attacking it requires being able to defend a and $\dot{\forall} x.a$ as attacking it requires a (finite) choice of $s : \mathbf{S}$. Unless specified otherwise, we are working in a fixed standard structure $\mathbf{S}$.

Lemma 4. *For any $a : \mathfrak{A}$ and any ρ, A, C with $a \in A$ one may assume $\rho \models a$ to deduce $\text{Win}(\rho, A, C)$, i.e. $(\rho \models a \rightarrow \text{Win}(\rho, A, C)) \rightarrow \text{Win}(\rho, A, C)$.*

Proof. Per induction on the structure of a . Suppose $\rho \models a$ entails $\text{Win}(\rho, a::A, C)$ we show $\text{Win}(\rho, a::A, C)$. Where appropriate, we implicitly make use of the fact that $\text{Win}(\rho, A, C)$ entails $\text{Win}(\vec{s} \cdot \rho, A', \uparrow^n C)$ where $\vec{s} : \mathbf{S}$, $|\vec{s}| = n$ and $\uparrow^n A \subseteq A'$.

$a = P\vec{t}$: Then the proponent attacks $P\vec{t}$, forcing the opponent to demonstrate $\rho \models P\vec{t}$. The proponent may then continue according to the assumption.

$a = \perp$: The proponent attacks $\perp$ and wins.

$a = a \wedge b$: The proponent starts by attacking $a \wedge b$ with A_L and A_R , leaving us to prove that $\text{Win}(\rho, a::b::A, C)$. Applying the IH for a and b means we may assume $\rho \models a$ and $\rho \models b$ to prove $\text{Win}(\rho, a::b::A, C)$. As we thus know $\rho \models a \wedge b$ the proponent can proceed per assumption.

$a = a \vee b$: The proponent attacks $a \vee b$, leaving $\text{Win}(\rho, c::a \vee c::A, C)$ for $c \in \{a, b\}$. Applying the IH for c allows us to assume $\rho \models c$, meaning the proponent can continue per assumption in either case.

$a = \exists a$: The proponent attacks $\exists a$, leaving $\text{Win}(s \cdot \rho, a::\uparrow(\exists a::A), \uparrow C)$. Per IH on a we may assume $s \cdot \rho \models a$ and thus $\rho \models \exists a$, continuing per assumption.

Theorem 5. *Pick $\varphi : \mathfrak{F}^D$, then $\rho \models \varphi$ entails $\text{Win}(\rho, A, C, \varphi)$ for any A and C .*

Proof. Proof per induction on φ .

$\varphi : \mathfrak{A}$: We only handle $\varphi = P\vec{t}$ and $\varphi = \perp$ as the other cases are subsumed by other cases of this proof. If $\rho \models \perp$ we are done. If $\varphi = P\vec{t}$ then the only possible challenge is $A_P \vec{t}$ to which the proponent responds by demonstrating $\rho \models P\vec{t}$.

$\varphi = \varphi \wedge \psi$: Then we know $\rho \models \varphi$ and $\rho \models \psi$. The possible challenges are A_L and A_R , defending against which leaves $\text{Win}(\rho, A, A_X::C, \theta)$ for some $\theta \in \{\varphi, \psi\}$. Either case holds per IH for θ .

$\varphi = \varphi \vee \psi$: Then we know $\rho \models \theta$ for $\theta \in \{\varphi, \psi\}$. The proponent thus defends against A_V by admitting θ and proceeds per IH for θ .

$\varphi = a \rightarrow \psi$: Then we know $\rho \models a$ entails $\rho \models \psi$. In attacking, the opponent will admit a , leaving $\text{Win}(\rho, a::A, A \rightarrow a \psi::C)$. We apply Lemma 4, allowing us to assume $\rho \models a$ to prove $\text{Win}(\rho, a::A, A \rightarrow a \psi::C)$. The proponent thus defends by admitting ψ and proceeds per IH on ψ as $\rho \models \psi$ per assumption.

$\varphi = \forall \varphi$: We know that $s \cdot \rho \models \varphi$ for any $s : \mathbf{S}$. The challenge will be $A_s \varphi$ for some $s : \mathbf{S}$. The proponent reacts by admitting $\varphi(s)$, proceeding per IH.

$\varphi = \exists \varphi$: Then $s \cdot \rho \models \varphi$ for some $s : \mathbf{S}$. The only possible challenge is $A_\exists \varphi$ to which the proponent responds by admitting φ with s as the witness, proceeding per IH.

Theorem 5 can be made sense of in the following way: It is known that CIC is consistent with various non-intuitionistic intermediate logics whose axiom schemes lie partially in $\mathfrak{F}^D$, e.g. classical logic ($a \vee \neg a$) and Gödel-Dummett logic LC ($(a \rightarrow b) \vee (b \rightarrow a)$) for $a, b : \mathfrak{A}$. By Theorem 5, it is thus consistent with CIC that these parts of the axiom schemes are dialogically valid according to the

rules investigated in this section. However, this means that completeness of these dialogues wrt. to some intuitionistic deduction system cannot be proven: If it could, that would mean it was consistent that parts of these non-intuitionistic axiom schemata, e.g. $P \dot{\vee} \dot{\neg} P$, were provable intuitionistically, which we know not to be the case.

Under the full law of the excluded middle, one can obtain an even stronger result: intuitionistic and classical dialogical validity, as defined here, fully coincide. This result relies on the following lemma:

Lemma 5. *Assuming LEM, the following holds for any φ in any standard $\mathbf{S}$*

1. $\forall \rho, A, C. \varphi \in A \rightarrow \rho \models \dot{\neg} \varphi \rightarrow \text{Win}(\rho, A, C)$
2. $\forall \rho, A, C. \rho \models \varphi \rightarrow \text{Win}(\rho, A, C, \varphi)$

Proof. We show prove both claims per simultaneous induction on φ . For most cases, 2. is the same as in Theorem 5 in which case we omit it. We write IH_i for the inductive hypothesis for part i of Lemma 5.

$\varphi = P\vec{t}$: 1. The proponent may force the opponent to demonstrate $\rho \models P\vec{t}$ by attacking $P\vec{t} \in A$, contradicting $\rho \models \neg P\vec{t}$.

$\varphi = \perp$: 1. The proponent may win by attacking $\perp \in A$.

$\varphi = \varphi \dot{\rightarrow} \psi$:

1. Suppose $\rho \models \dot{\neg}(\varphi \dot{\rightarrow} \psi)$, meaning $\rho \models \varphi$ and $\rho \models \dot{\neg} \psi$. The proponent then attacks $\varphi \dot{\rightarrow} \psi \in A$. If the opponent counters the attack, the proponent can win by playing the strategy obtained by IH₂ on $\rho \models \varphi$. If the opponent admits ψ , then the proponent plays according to IH₁ on $\rho \models \dot{\neg} \psi$.
2. Suppose $\rho \models \varphi \dot{\rightarrow} \psi$. The opponent attacks $\varphi \dot{\rightarrow} \psi$ with $A \rightarrow \varphi \psi$, admitting φ . By the law of the excluded middle, either $\rho \models \varphi$ or $\rho \models \dot{\neg} \varphi$. In the latter case, the proponent can now proceed per IH₂ on $\rho \models \dot{\neg} \varphi$. In the former case we have $\rho \models \psi$ per assumption and the proponent can proceed by admitting ψ and playing along IH₁ on $\rho \models \psi$.

$\varphi = \varphi \dot{\wedge} \psi$: 1. Suppose $\rho \models \dot{\neg}(\varphi \dot{\wedge} \psi)$, meaning $\rho \models \dot{\neg} \varphi$ or $\rho \models \dot{\neg} \psi$. The proponent attacks the side of the contradicted formula of $\varphi \dot{\wedge} \psi \in A$ and proceeds per IH₁.

$\varphi = \varphi \dot{\vee} \psi$: 1. Suppose $\rho \models \dot{\neg}(\varphi \dot{\vee} \psi)$, meaning $\rho \models \dot{\neg} \varphi$ and $\rho \models \dot{\neg} \psi$. By attacking $\varphi \dot{\vee} \psi \in A$, the proponent thus forces the opponent to admit either clause, being able to proceed via IH₁ in either case.

$\varphi = \dot{\forall} \varphi$: If $\rho \models \dot{\neg} \dot{\forall} \varphi$ that means there is an $s : \mathbf{S}$ with $s \cdot \rho \models \dot{\neg} \varphi$. The proponent thus attack $\dot{\forall} \varphi$ with $A_s \varphi$ and proceeds per IH₁.

$\varphi = \dot{\exists} \varphi$: Suppose $\rho \models \dot{\neg} \dot{\exists} \varphi$, meaning $s \cdot \rho \models \dot{\neg} \varphi$ for any $s : \mathbf{S}$. Then the proponent attacks $\dot{\exists} \varphi \in A$ and proceeds per IH₁.

Theorem 6. *Under LEM, classical and intuitionistic dialogical validity agree.*

Proof.

$\leftarrow$: This is the case—even without the law of excluded middle—as every intuitionistic winning strategy is also a classical winning strategy on the same state.

$\rightarrow$: Suppose $\Gamma \models^D \varphi$ classically. By Lemma 2 this means $\Gamma \models^E \varphi$. As every standard structure is exploding and under the LEM every structure is classical, this means φ is valid under Γ in every standard structure. By Lemma 5 this entails that $\Gamma \models^D \varphi$ intuitionistically.

Note that the failure of completeness is not simply due to the “wrong choice of rules” for intuitionistic dialogues. While there are examples of the proponent restriction failing to turn classical dialogues intuitionistic (see e.g. [18]) we do not believe this to be the case in this instance. Formal dialogues for intuitionistic first-order logic, which are obtained from their classical counterparts via that very restriction, are constructively sound and complete [3, 4]. Rather, we believe the cause of failure lies in the difference between formal and material dialogues: their treatment of atomic formulas.

5 Discussion

Mechanization of Active Research. While researching for this article, we mechanized the results from Sect. 3 in the interactive theorem prover Coq. The mechanization can be found at [17]. Mechanizing the results of Sect. 3 revealed some mistakes in our initial definition of the rules for material dialogues which, albeit being minor, invalidated both soundness and completeness. We missed these mistakes while working “on paper” and believe it would have taken much longer to discover them without the mechanization. Having machine checked the definitions in Sect. 3 gave us sufficient confidence in the correctness of the technical details of material dialogues to work solely on paper for the remainder of the article. It should also be noted that the mechanization took up only about a quarter of the overall time spent researching, in large part due to building on top of the a large preexisting mechanization from [5]. We believe this might be a worthwhile trade-off between the time requirement of a full mechanization of all results and the room for error in working solely on paper.

Proof Strategies for Completeness. We prove completeness by relating dialogical validity to validity in a model-theoretic semantics and appealing to a preexisting completeness result. This is the quickest way to obtain completeness in the framework set up by [5]. For classical material dialogues, we believe it would also be possible to obtain a direct constructive completeness proof with regards to natural deduction on the basis of a Henkin construction.

Benefits of Material Dialogues. When working with Tarski semantics in CIC, one’s attention needs to be restricted to classical structures (as define in Sect. 2). Many structures of interest such as the standard model of Peano arithmetic is not provably classical in constructive settings and can thus not be studied in a Tarski setting. In contrast, Sect. 3 demonstrates that classical material dialogues embody classical logic regardless of the classicality of the underlying structure. It thus seems like a promising basis on which to carry out model-theoretic investigations of classical first-order logic in constructive settings. However, we have not yet investigated these possibilities more deeply.

Faithfulness to Lorenzen's Material Dialogues. We attempted to be as faithful to Lorenzen's definitions from [12,13] as possible while implementing material dialogues played over first-order structures. Arguably, this second aspect already is in conflict with Lorenzen's ideas as he placed a lot of value in the "underlying game" for settling atomic propositions to be of a discrete nature, something completely lost in our formulation. However, all the attacks and defenses for the connectives of first-order logic are exactly as they are in Lorenzen's work. Notably, our usage of a structure defined, standard constant $\perp^S$ is very similar to Lorenzen's definitions which propose to fix some unwinnable game as a stand-in for a "demonstration" of $\perp$.

References

1. de Bruijn, N.G.: Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem. In: Proceedings of the Indagationes Mathematicae, vol. 75, pp. 381–392. Elsevier (1972)
2. Coquand, T., Huet, G.: The calculus of constructions. Ph.D. thesis, Inria (1986)
3. Felscher, W.: Dialogues, strategies, and intuitionistic provability. *Ann. Pure Appl. Logic* **28**(3), 217–254 (1985)
4. Forster, Y., Kirst, D., Wehr, D.: Completeness theorems for first-order logic analysed in constructive type theory. In: Artemov, S., Nerode, A. (eds.) LFCS 2020. LNCS, vol. 11972, pp. 47–74. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-36755-8_4
5. Forster, Y., Kirst, D., Wehr, D.: Completeness theorems for first-order logic analysed in constructive type theory: extended version. *J. Log. Comput.* **31**(1), 112–151 (2021)
6. Girard, J.Y.: Interprétation fonctionnelle et élimination des coupures de l'arithmétique d'ordre supérieur. Thèse de doctorat d'état, Université Paris 7 (1972)
7. Herbelin, H., Ilik, D.: An analysis of the constructive content of Henkin's proof of Gödel's completeness theorem (2016). <https://pauillac.inria.fr/~herbelin/articles/godel-completeness-draft16.pdf>. Draft
8. Hintikka, J.: Language-games for quantifiers. In: Rescher, N. (ed.) *Studies in Logical Theory*, pp. 46–72. Blackwell (1968)
9. Krabbe, E.C.: Dialogue logic. In: *Handbook of the History of Logic*, vol. 7, pp. 665–704. Elsevier (2006)
10. Lorenz, K.: *Arithmetik und Logik als Spiele*. Ph.D. thesis, Christian-Albrechts-Universität zu Kiel (1961)
11. Lorenzen, P.: Einführung in die operative logik und mathematik. *Grundlehren der mathematischen Wissenschaften* 78 (1955)
12. Lorenzen, P.: Logik und Agon. In: *Atti del XII Congresso Internazionale di Filosofia*, vol. 4, pp. 187–194 (1960)
13. Lorenzen, P.: Ein dialogisches Konstruktivitätskriterium. In: *Proceedings of the Symposium on Foundations of Mathematics, Warsaw, 2–9 September 1959*, pp. 193–200 (1961)
14. Paulin-Mohring, C.: Inductive definitions in the system Coq rules and properties. In: Bezem, M., Groote, J.F. (eds.) *TLCA 1993*. LNCS, vol. 664, pp. 328–345. Springer, Heidelberg (1993). <https://doi.org/10.1007/BFb0037116>

15. Sørensen, M.H., Urzyczyn, P.: Sequent calculus, dialogues, and cut elimination. In: *Reflections on Type Theory, λ -Calculus, and the Mind*, pp. 253–261 (2007)
16. Veldman, W.: An intuitionistic completeness theorem for intuitionistic predicate logic. *J. Symb. Logic* **41**(1), 159–166 (1976)
17. Wehr, D., Kirst, D.: Mechanization: material dialogues for first-order logic in constructive type theory, May 2022. <https://github.com/doweher/material-dialogues-coq/tree/wollic-2022>
18. Wolff, J.: What’s the name of the game? Dialogue game semantics for intuitionistic modal logic with strict implication. Master’s thesis, University of Gothenburg (2022). <https://gupea.ub.gu.se/handle/2077/72348>. Accepted 27 June 2022. T06:40:24Z



On the Computational Properties of the Uncountability of the Real Numbers

Sam Sanders^(✉)

Department of Philosophy II, RUB Bochum, Bochum, Germany

sasander@me.com

<https://sasander.wixsite.com/academic>

Abstract. The uncountability of the real numbers is one of their most basic properties, known (far) outside of mathematics. Cantor’s 1874 proof of the uncountability of the real numbers even appears in the very first paper on set theory, i.e. a historical milestone. Despite this famous status and history, the computational properties of the uncountability of the real numbers have not been studied much. In this paper, we study the following computational operation that witnesses that the real numbers not countable:

on input a countable set of reals, output a real not in that set.

In particular, we formulate a considerable number of operations that are computationally equivalent to the centred operation, working in Kleene’s higher-order computability theory based on his S1-S9 computation schemes. Perhaps surprisingly, our equivalent operations involve most basic properties of the *Riemann integral* and Volterra’s early work circa 1881.

1 Introduction

1.1 Motivation and Overview

Like Hilbert ([9]), we believe the infinite to be a central object of study in mathematics. That the infinite comes in ‘different sizes’ is a relatively new insight, due to Cantor around 1874 ([6]), in the guise of the *uncountability of $\mathbb{R}$* , also known simply as *Cantor’s theorem*. Cantor’s paper [6] is heralded as the first paper in set theory, complete with its own Wikipedia page [39].

Despite the aforementioned famous status, the computational properties of the uncountability of $\mathbb{R}$ have not been studied much. In this paper, we study the following computational operation witnessing that $\mathbb{R}$ is not countable:

on input a countable set $A \subset \mathbb{R}$, output $y \in \mathbb{R} \setminus A$.

This research was supported by the *Deutsche Forschungsgemeinschaft* (DFG) via the grant *Reverse Mathematics beyond the Gödel hierarchy* (SA3418/1-1).

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2022
 A. Ciabatonni et al. (Eds.): WoLLIC 2022, LNCS 13468, pp. 362–377, 2022.
https://doi.org/10.1007/978-3-031-15298-6_23

In particular, we show that this operation is computationally equivalent to the following ones. Note that a *regulated* function has left and right limits everywhere, as studied by Bourbaki for Riemann integration (see Sect. 1.3.3).

- (i). On input regulated $f : [0, 1] \rightarrow \mathbb{R}$, output a point $x \in [0, 1]$ where f is continuous (or quasi-continuous or lower semi-continuous or Darboux).
- (ii). On input regulated $f : [0, 1] \rightarrow [0, 1]$ with Riemann integral $\int_0^1 f(x)dx = 0$, output $x \in [0, 1]$ with $f(x) = 0$ (Bourbaki, [4 p. 61, Cor. 1]).
- (iii). (Volterra) On input regulated $f, g : [0, 1] \rightarrow \mathbb{R}$, output a real $x \in [0, 1]$ such that f and g are both continuous or both discontinuous at x .
- (iv). (Volterra) On input regulated $f : [0, 1] \rightarrow \mathbb{R}$, output either $q \in \mathbb{Q} \cap [0, 1]$ where f is discontinuous, or output $x \in [0, 1] \setminus \mathbb{Q}$ where f is continuous.
- (v). On input regulated $f : [0, 1] \rightarrow \mathbb{R}$, output $y \in (0, 1)$ where the function $F(x) := \lambda x. \int_0^x f(t)dt$ is differentiable with derivative equal to $f(y)$.
- (vi). On input regulated $f : [0, 1] \rightarrow \mathbb{R}$, output $a, b \in [0, 1]$ such that the set $\{x \in [0, 1] : f(a) \leq f(x) \leq f(b)\}$ is infinite.

A full list may be found in Theorem 16. We work in Kleene's higher-order computability theory, based on his S1-S9 computation schemes (see Sect. 1.3.1). Like in [28], a much weaker notion of computability should suffice, namely based on a fragment of Gödel's T . We note that by [27*Theorem 18], the operations in the above list are *hard*¹ to compute relative to the usual hierarchy based on comprehension. An explanation for this phenomenon is also in Sect. 1.3.1.

Some of the above operations, including items (iii) and (iv) in the list, stem from Volterra's early work (1881) in the spirit of -but predating- the Baire category theorem, as discussed in Sect. 1.2.

It should not be a surprise that the above equivalences crucially hinge on our definition of 'countable set', as discussed in detail in Remark 8. In a nutshell, the set theoretic definition of 'countable set' (based on injections or bijections to $\mathbb{N}$) is not suitable for studying regulated functions; our alternative definition (based on unions over $\mathbb{N}$ of finite sets) is closer to the way countable sets occur 'in the wild', at least in the context of regulated functions.

Finally, this paper is a spin-off from my project with Dag Normann (University of Oslo) on the logical and computational properties of the uncountable. The interested reader may consult [20, 25, 28] for further details on our project.

1.2 Volterra's Early Work and Related Results

We introduce Volterra's early work from [38] as it pertains to this paper.

First of all, the Riemann integral was groundbreaking for a number of reasons, including its ability to integrate functions with infinitely many points of discontinuity, as shown by Riemann himself ([29]). A natural question is then 'how

¹ The functional S_k^2 from Sect. 1.3.2 can decide Π_k^1 -formulas, but the centred operation is not computable in S_k^2 (or their union). Kleene's $\exists^3$ from Sect. 1.3.2 computes the centred operation, but the former also yields full second-order arithmetic..

discontinuous’ a Riemann integrable function can be. In this context, Thomae introduced $T : \mathbb{R} \rightarrow \mathbb{R}$ around 1875 in [36*p. 14, §20]:

$$T(x) := \begin{cases} 0 & \text{if } x \in \mathbb{R} \setminus \mathbb{Q} \\ \frac{1}{q} & \text{if } x = \frac{p}{q} \text{ and } p, q \text{ are co-prime} \end{cases} \quad (1)$$

Thomae’s function T is integrable on any interval, but has a dense set of points of discontinuity, namely $\mathbb{Q}$, and a dense set of points of continuity, namely $\mathbb{R} \setminus \mathbb{Q}$.

The perceptive student, upon seeing Thomae’s function as in (1), will ask for a function continuous at each rational point and discontinuous at each irrational one. Such a function cannot exist, as is generally proved using the Baire category theorem. However, Volterra in [38] already established this negative result about twenty years before the publication of the Baire category theorem.

Secondly, as to the content of Volterra’s paper [38], we find the following theorem on the first page, where a function is *pointwise discontinuous* if it has a dense set of continuity points.

Theorem 1 (Volterra, 1881). *There do not exist pointwise discontinuous functions defined on an interval for which the continuity points of one are the discontinuity points of the other, and vice versa.*

Volterra then states two corollaries, of which the following is perhaps well-known in ‘popular mathematics’ and constitutes the aforementioned negative result.

Corollary 2 (Volterra, 1881). *There is no $\mathbb{R} \rightarrow \mathbb{R}$ function that is continuous on $\mathbb{Q}$ and discontinuous on $\mathbb{R} \setminus \mathbb{Q}$.*

Thirdly, we shall study Volterra’s theorem and corollary restricted to regulated functions (see Sect. 1.3.3). The latter kind of functions are automatically ‘pointwise discontinuous’ in the sense of Volterra.

Fourth, Volterra’s results from [38] are generalised in [7, 33]. The following theorem is immediate from these generalisations.

Theorem 3. *For any countable dense set $D \subset [0, 1]$ and $f : [0, 1] \rightarrow \mathbb{R}$, either f is discontinuous at some point in D or continuous at some point in $\mathbb{R} \setminus D$.*

In conclusion, the perceptive reader has already noted that most equivalent principles have a ‘Baire category theorem’ flavour to them. In particular, the intuitive notion of ‘small or negligible set’ can be formalised in at least two (fairly independent) ways, namely based on ‘measure’ and ‘density/meagre’. The equivalences in this paper do not (and we believe cannot) involve the former, but heavily depend on the latter.

1.3 Preliminaries and Definitions

We briefly introduce Kleene’s *higher-order computability theory* in Sect. 1.3.1. We introduce some essential axioms (Sect. 1.3.2) and definitions (Sect. 1.3.3). A full introduction may be found in e.g. [25*§2]. Since Kleene’s computability

theory borrows heavily from type theory, we shall often use common notations from the latter; for instance, the natural numbers are type 0 objects, denoted n^0 or $n \in \mathbb{N}$. Similarly, elements of Baire space are type 1 objects, denoted $f \in \mathbb{N}^{\mathbb{N}}$ or f^1 . An overview of this kind of notations is in Sect. 3.

1.3.1 Kleene's Computability Theory

Our main results are in computability theory and we make our notion of ‘computability’ precise as follows.

- (I) We adopt ZFC, i.e. Zermelo-Fraenkel set theory with the Axiom of Choice, as the official metatheory for all results, unless explicitly stated otherwise.
- (II) We adopt Kleene's notion of *higher-order computation* as given by his nine clauses S1–S9 (see [15*Ch. 5] or [11]) as our official notion of ‘computable’.

We mention that S1–S8 are rather basic and merely introduce a kind of higher-order primitive recursion with higher-order parameters. The real power comes from S9, which essentially hard-codes the *recursion theorem* for S1–S9-computability in an ad hoc way. By contrast, the recursion theorem for Turing machines is derived from first principles in [35].

On a historical note, it is part of the folklore of computability theory that many have tried (and failed) to formulate models of computation for objects of all finite types in which one derives the recursion theorem in a natural way. For this reason, Kleene ultimately introduced S1–S9, which were initially criticised for their ad hoc nature, but eventually received general acceptance nonetheless. On a related note, an equivalent-but-more-elegant λ -calculus formulation of S1–S9-computability based on fixed point operators may be found in [28]; an additional advantage of the framework from [28] is that it accommodates partial objects.

We refer to [15, 28] for a (more) thorough overview of higher-order computability theory. We do mention the distinction between ‘normal’ and ‘non-normal’ functionals based on the following definition from [15*§5.4]. We only make use of $\exists^n$ for $n = 2, 3$, as defined in Sect. 1.3.2.

Definition 4. For $n \geq 2$, a functional of type n is called *normal* if it computes Kleene's $\exists^n$ following S1–S9, and *non-normal* otherwise.

It is a historical fact that higher-order computability theory, based on Kleene's S1–S9 schemes, has focused primarily on the world of *normal* functionals; this opinion can be found [15*§5.4]. Nonetheless, we have previously studied the computational properties of new *non-normal* functionals, namely those that compute the objects claimed to exist by:

- covering theorems due to Heine-Borel, Vitali, and Lindelöf ([18, 19, 21]),
- the Baire category theorem ([22]),
- local-global principles like *Pincherle's theorem* ([23]),
- weak fragments of the Axiom of (countable) Choice ([24]),
- the uncountability of $\mathbb{R}$ and the Bolzano-Weierstrass theorem for countable sets in Cantor space ([25, 26]),

– the Jordan decomposition theorem and related results ([27,28]).

In this paper, we greatly extend the study of the uncountability of $\mathbb{R}$ mentioned in the fifth item; the operations sketched in Sect. 1 are all non-normal, in that they do not even compute Kleene's $\exists^2$ from Sect. 1.3.2.

1.3.2 Some Comprehension Functionals

In Turing-style computability theory, computational hardness is measured in terms of where the oracle set fits in the well-known comprehension hierarchy. For this reason, we introduce some axioms and functionals related to *higher-order comprehension* in this section. We are mostly dealing with *conventional* comprehension here, i.e. only parameters over $\mathbb{N}$ and $\mathbb{N}^{\mathbb{N}}$ are allowed in formula classes like Π_k^1 and Σ_k^1 .

First of all, the functional φ^2 , also called *Kleene's quantifier* $\exists^2$, as in $(\exists^2)$ is clearly discontinuous at $f = 11 \dots$; in fact, $\exists^2$ is (computationally) equivalent to the existence of $F : \mathbb{R} \rightarrow \mathbb{R}$ such that $F(x) = 1$ if $x >_{\mathbb{R}} 0$, and 0 otherwise via Grilliot's trick (see [13*§3]).

$$(\exists \varphi^2 \leq_2 1)(\forall f^1)[(\exists n)(f(n) = 0) \leftrightarrow \varphi(f) = 0]. \quad (\exists^2)$$

Related to $(\exists^2)$, the functional μ^2 in (μ^2) is called *Feferman's μ* ([2]).

$$\begin{aligned} (\exists \mu^2)(\forall f^1)[&[(\exists n)(f(n) = 0) \rightarrow [f(\mu(f)) = 0 \wedge (\forall i < \mu(f))(f(i) \neq 0)]] \\ &\wedge [(\forall n)(f(n) \neq 0) \rightarrow \mu(f) = 0]]. \end{aligned} \quad (\mu^2)$$

We have $(\exists^2) \leftrightarrow (\mu^2)$ over Kohlenbach's base theory ([13]), while $\exists^2$ and μ^2 are also computationally equivalent.

Secondly, the functional S^2 in (S^2) is called *the Suslin functional* ([13]).

$$(\exists S^2 \leq_2 1)(\forall f^1)[(\exists g^1)(\forall n^0)(f(\bar{g}n) = 0) \leftrightarrow S(f) = 0]. \quad (S^2)$$

By definition, the Suslin functional S^2 can decide whether a Σ_1^1 -formula as in the left-hand side of (S^2) is true or false. We similarly define the functional S_k^2 which decides the truth or falsity of Σ_k^1 -formulas from L_2 . We note that the operators ν_n from [5*p. 129] are essentially S_n^2 strengthened to return a witness (if existent) to the Σ_n^1 -formula at hand.

Thirdly, the functional E^3 clearly computes $\exists^2$ and S_k^2 for any $k \in \mathbb{N}$:

$$(\exists E^3 \leq_3 1)(\forall Y^2)[(\exists f^1)(Y(f) = 0) \leftrightarrow E(Y) = 0]. \quad (\exists^3)$$

The functional from $(\exists^3)$ is also called *Kleene's quantifier* $\exists^3$, and we use the same -by now obvious- convention for other functionals.

In conclusion, the operations sketched in Sect. 1 are computable in $\exists^3$ but not in any S_k^2 , as noted in Footnote (see footnote 1) and which immediately follows from [27*Theorem 18]. Many non-normal functionals exhibit the same 'computational hardness' and we merely view this as support for the development of a separate scale for classifying non-normal functionals.

1.3.3 Some Definitions

We introduce some definitions needed in the below, mostly stemming from mainstream mathematics.

First of all, we shall study the following notions of weak continuity, going back (at least) to the days of Baire and Volterra.

Definition 5. For $f : [0, 1] \rightarrow \mathbb{R}$, we have the following definitions:

- f is *upper semi-continuous* at $x_0 \in [0, 1]$ if $f(x_0) \geq_{\mathbb{R}} \limsup_{x \rightarrow x_0} f(x)$,
- f is *lower semi-continuous* at $x_0 \in [0, 1]$ if $f(x_0) \leq_{\mathbb{R}} \liminf_{x \rightarrow x_0} f(x)$,
- f is *quasi-continuous* at $x_0 \in [0, 1]$ if for $\epsilon > 0$ and an open neighbourhood U of x_0 , there is a non-empty open $G \subset U$ with $(\forall x \in G)(|f(x_0) - f(x)| < \epsilon)$.
- f is *regulated* if for every x_0 in the domain, the ‘left’ and ‘right’ limit $f(x_0-) = \lim_{x \rightarrow x_0-} f(x)$ and $f(x_0+) = \lim_{x \rightarrow x_0+} f(x)$ exist.

Scheeffer studies discontinuous regulated functions in [32] around 1884 (without using the term ‘regulated’), while Bourbaki develops Riemann integration based on regulated functions in [3]. An interesting observation about regular functions and continuity is as follows.

Remark 6 (Continuity and regulatedness). First of all, as discussed in [13*§3], the *local* equivalence for functions on Baire space between sequential and ‘epsilon-delta’ continuity cannot be proved in ZF. By [26*Theorem 3.32], this equivalence for *regulated* functions is provable in ZF (and much weaker systems).

Secondly, μ^2 readily computes the left and right limits of regulated $f : [0, 1] \rightarrow \mathbb{R}$. In this way, the formula ‘ f is continuous at $x \in [0, 1]$ ’ is decidable using μ^2 , namely equivalent to the formula ‘ $f(x+) = f(x) = f(x-)$ ’. The usual ‘epsilon-delta’ definition of continuity involves quantifiers over $\mathbb{R}$, i.e. the previous equality is much simpler and more elementary.

Secondly, we also need the notion of ‘intermediate value property’, also called the ‘Darboux property’.

Definition 7. [Darboux property] Let $f : [0, 1] \rightarrow \mathbb{R}$ be given.

- A real $y \in \mathbb{R}$ is a left (resp. right) *cluster value* of f at $x \in [0, 1]$ if there is $(x_n)_{n \in \mathbb{N}}$ such that $y = \lim_{n \rightarrow \infty} f(x_n)$ and $x = \lim_{n \rightarrow \infty} x_n$ and $(\forall n \in \mathbb{N})(x_n \leq x)$ (resp. $(\forall n \in \mathbb{N})(x_n \geq x)$).
- A point $x \in [0, 1]$ is a *Darboux point* of $f : [0, 1] \rightarrow \mathbb{R}$ if for any $\delta > 0$ and any left (resp. right) cluster value y of f at x and $z \in \mathbb{R}$ strictly between y and $f(x)$, there is $w \in (x - \delta, x)$ (resp. $w \in (x, x + \delta)$) such that $f(w) = y$.

By definition, a point of continuity is also a Darboux point, but not vice versa.

Thirdly, as suggested in Sect. 1, the set-theoretic definition of countable set is not suitable for the study of regulated functions (in computability theory). We present our alternative notion in Definition 11, which amounts to ‘unions over $\mathbb{N}$ of finite sets’. We first provide some motivation in Remark 8. We assume subsets of $\mathbb{R}$ are given by their characteristic functions.

Remark 8 (Countable sets by any other name). First of all, we have previously investigated the computational properties of the Jordan decomposition theorem in [27, 28]. In the latter, a central object of study is as follows:

*a functional $\mathcal{E}$ that on input $A \subset [0, 1]$ **and** $Y : [0, 1] \rightarrow \mathbb{N}$ such that Y is injective orbijective on A , outputs a sequence of reals listing all reals in A .*

We stress that the functional $\mathcal{E}$ crucially has **two** inputs: a countable set $A \subset [0, 1]$ **and** a function $Y : [0, 1] \rightarrow \mathbb{N}$ witnessing that A is countable. Indeed, omitting the second input from $\mathcal{E}$ yields a choice function well-known from the Axiom of Choice. However, such functions lack any and all computational content (even in very weak instances: see [24]). Thus, the functional $\mathcal{E}$ is not a meaningful object of study in computability theory *unless* we include the second input Y . Secondly, consider the following sets definable via $\exists^2$, for regulated $f : [0, 1] \rightarrow \mathbb{R}$:

$$\begin{aligned} A &:= \{x \in (0, 1) : f(x+) \neq f(x) \vee f(x-) \neq f(x)\}, \\ A_n &:= \{x \in (0, 1) : |f(x+) - f(x)| > \tfrac{1}{2^n} \vee |f(x-) - f(x)| > \tfrac{1}{2^n}\}. \end{aligned} \quad (2)$$

Clearly, $A = \bigcup_{n \in \mathbb{N}} A_n$ collects all points in $(0, 1)$ where f is discontinuous; this set is central to many proofs involving regulated functions (see e.g. [1*Thm. 0.36]). Now, that A_n is finite follows by a standard² compactness argument. However, while A is then countable, we are unable to construct an injection from A to $\mathbb{N}$ (let alone a bijection), even assuming S_m^2 . Similarly, for a set $A \subset \mathbb{R}$ without limit points, $A \cap [-n, n]$ is finite for any $n \in \mathbb{N}$, yet we are again unable to construct an injection from A to $\mathbb{N}$, even assuming S_m^2 .

In conclusion, the countable set A in (2) is central to many proofs involving regulated functions, but it seems we cannot (in general) construct an injection from A to $\mathbb{N}$. However, this injection is a crucial input of the functional $\mathcal{E}$ as omitting it gives rise to a choice function from the Axiom of Choice.

Besides pointing out a problem with the usual definition of countable set, Remark 8 also suggests an alternative, more suitable, definition of countable set, namely ‘union over $\mathbb{N}$ of finite sets’. We now introduce the ‘usual’ definitions of countable set (Definitions 9 and 10), and our alternative one (Definition 11).

Definition 9. [Enumerable sets of reals] A set $A \subset \mathbb{R}$ is *enumerable* if there exists a sequence $(x_n)_{n \in \mathbb{N}}$ such that $(\forall x \in \mathbb{R})(x \in A \rightarrow (\exists n \in \mathbb{N})(x =_{\mathbb{R}} x_n))$.

This definition reflects the RM-notion of ‘countable set’ from [34*V.4.2]. We note that given μ^2 from Sect. 1.3.2, we may replace the final implication in Definition 9 by an equivalence. One sometimes finds ‘denumerable’ in the literature to describe Definition 9.

Definition 10. [Countable subset of $\mathbb{R}$] A set $A \subset \mathbb{R}$ is *countable* if there exists $Y : \mathbb{R} \rightarrow \mathbb{N}$ such that $(\forall x, y \in A)(Y(x) =_0 Y(y) \rightarrow x =_{\mathbb{R}} y)$. If $Y : \mathbb{R} \rightarrow \mathbb{N}$ is also *surjective*, i.e. $(\forall n \in \mathbb{N})(\exists x \in A)(Y(x) = n)$, we call A *strongly countable*.

² If A_n were infinite, the Bolzano-Weierstrass theorem implies the existence of a limit point $y \in [0, 1]$ for A_n . One readily shows that $f(y+)$ or $f(y-)$ does not exist, a contradiction as f is assumed to be regulated.

The first part of Definition 10 is from Kunen’s set theory textbook ([14*p. 63]) and the second part is taken from Hrbacek-Jech’s set theory textbook [10] (where the term ‘countable’ is used instead of ‘strongly countable’). For the rest of this paper, ‘strongly countable’ and ‘countable’ shall exclusively refer to Definition 10, *except when explicitly stated otherwise*.

Definition 11. A set $A \subset \mathbb{R}$ is *weakly countable* if there is a *height* $H : \mathbb{R} \rightarrow \mathbb{N}$ for A , i.e. for all $n \in \mathbb{N}$, $A_n := \{x \in A : H(x) < n\}$ is finite.

We note that the notion of ‘height’ is mentioned in e.g. [16*p. 33] and [37], from whence we took this notion. The following remark is crucial for the below.

Remark 12 (Inputs and heights). Similar to the functional $\mathcal{E}$ in Remark 8, a functional defined on weakly countable sets always takes **two** inputs: the set $A \subset \mathbb{R}$ **and** the height $H : \mathbb{R} \rightarrow \mathbb{N}$. We note that given a sequence of finite sets $(X_n)_{n \in \mathbb{N}}$ in $\mathbb{R}$, a height can be defined as $H(x) := (\mu n)(x \in X_n)$ using Feferman’s μ^2 . A set is therefore weakly countable iff it is the union over $\mathbb{N}$ of finite sets. In particular, the set $A = \cup_{n \in \mathbb{N}} A_n$ as defined in (2) has a height $H_A(x) := (\mu n)(x \in A_n)$, which is crucial for the results in Sect. 2.2.

2 Main Results

We establish the results sketched in Sect. 1. We generally assume $\exists^2$ from Sect. 1.3.1 to avoid the technical details involved in the representation of sets and real numbers. Given the hardness of the operations from Sect. 1, as noted in Footnote (see footnote 1) and Sect. 1.3.2, this seems like a weak assumption.

2.1 Introduction

The uncountability of $\mathbb{R}$ can be studied in numerous guises in higher-order computability theory. For instance, the following notions are from [25, 27], where it is also shown that many extremely basic operations compute these realisers.

Definition 13. [Realisers for the uncountability of $\mathbb{R}$]

- A *Cantor functional/realiser* takes as input $A \subset [0, 1]$ and $Y : [0, 1] \rightarrow \mathbb{N}$ such that Y is injective on A , and outputs $x \notin A$.
- A **weak** *Cantor realiser* takes as input $A \subset [0, 1]$ and $Y : [0, 1] \rightarrow \mathbb{N}$ such that Y is **bijective** on A , and outputs $x \notin A$.
- A *NIN-realiser* takes as input $Y : [0, 1] \rightarrow \mathbb{N}$ and outputs $x, y \in [0, 1]$ with $x \neq y \wedge Y(x) = Y(y)$.

Clearly, the realisers in Definition 13 are based on the ‘set theoretic’ definition of countability. As discussed in Remark 8, the notion of weak countability (Definition 11) seems to be have better, suggesting the following stronger notion.

Definition 14 (Strong Cantor realiser). A *strong Cantor realiser* is any functional that on input a weakly countable $A \subset [0, 1]$, outputs $y \in [0, 1] \setminus A$.

Following Remark 12, a strong Cantor realiser takes as input a countable set **and** its height as in Definition 11. Modulo $\exists^2$, this amounts to an input consisting of a sequence $(X_n)_{n \in \mathbb{N}}$ of finite sets in $[0, 1]$ and an output $y \in [0, 1] \setminus \bigcup_{n \in \mathbb{N}} X_n$. The reader will observe that a strong Cantor realiser computes a normal one. Any (weak, strong, or normal) Cantor realiser witnesses the extremely basic idea that removing a countable set from an uncountable one, there is at least one element left. Nonetheless, the functionals S_k^2 from Sect. 1.3.2, which decide Π_k^1 -formulas, cannot compute a weak Cantor realiser ([27*Theorem 18]).

We stress that (strong, weak, or normal) Cantor realisers are *partial* objects: they provide the required output $y \in [0, 1] \setminus A$ when the inputs are given as in their respective specification. If the inputs are not according to the specification, the output may be undefined or may be some real number. As noted in Sect. 1.3.1, there is an equivalent formulation of Kleene's S1-S9 schemes that accommodates partial objects.

We show in Sect. 2.2 that many operations on regulated functions are computationally equivalent to strong Cantor realisers. This provides additional justification for the introduction of our new notion of weak countability.

2.2 Computational Equivalences for Strong Cantor Realisers

In this section, we show that many operations on regulated functions are computationally equivalent to strong Cantor realisers. We recall that subsets of $\mathbb{R}$ are given by characteristic functions.

First of all, various natural functionals compute a strong Cantor realiser.

Theorem 15. *Given $\exists^2$, these functionals compute a strong Cantor realiser:*

- the structure functional Ω (see [28]) which on input finite $X \subset \mathbb{R}$ outputs a finite sequence $\Omega(X) = (x_0, \dots, x_k)$ which includes all elements of X .
- the non-monotone induction functional (see [17, 22]),
- a realiser for the Baire category theorem (see [22*§6]).

Proof. Let $(X_n)_{n \in \mathbb{N}}$ be a sequence of finite sets in $[0, 1]$. Since Ω can enumerate finite sets, the sequence $\Omega(X_0) * \Omega(X_1) * \dots$ enumerates $\bigcup_{n \in \mathbb{N}} X_n$. One readily computes $y \in [0, 1]$ not in this sequence (see e.g. [34*II.4.9] or [8]), yielding a strong Cantor realiser.

The second item computes the third item by [22*Theorem 6.5]. Regarding the third item, $O_n := [0, 1] \setminus \bigcup_{k \leq n} X_k$ is (trivially) dense and open. Clearly any $y \in \bigcap_{n \in \mathbb{N}} O_n$ by definition satisfies $y \in [0, 1] \setminus \bigcup_{n \in \mathbb{N}} X_n$, i.e. a realiser for the Baire category theorem computes a strong Cantor realiser. $\square$

Secondly, we have the following theorem. Regarding item (k), the property ‘ x is a local strict (see footnote 3) maximum of f ’ does not seem to be decidable given $\exists^2$ for (general) regulated f , in contrast to Remark 6. We also note that items (f) and (g) are based on Volterra’s Theorem 1 and Corollary 2, while item (h) is based on the associated generalisation as in Theorem 3, and sports a certain robustness.

Theorem 16. *The following are computationally equivalent modulo $\exists^2$.*

- (a). *A strong Cantor realiser.*
- (b). *A functional that on input regulated $f : [0, 1] \rightarrow \mathbb{R}$, outputs $a, b \in [0, 1]$ such that $\{x \in [0, 1] : f(a) \leq f(x) \leq f(b)\}$ is infinite.*
- (c). *A functional that on input regulated $f : [0, 1] \rightarrow \mathbb{R}$, outputs $y \in [0, 1]$ where f is continuous.*
- (d). *A functional that on input regulated $f : [0, 1] \rightarrow \mathbb{R}$, outputs $y \in [0, 1]$ where f is quasicontinuous, or lower semi-continuous, or Darboux.*
- (e). *A ‘density’ functional that on input regulated $f : [0, 1] \rightarrow \mathbb{R}$, $x \in [0, 1]$, and $k \in \mathbb{N}$, outputs $y \in [0, 1] \cap B(x, \frac{1}{2^k})$ such that f is continuous at y .*
- (f). *A functional that on input regulated $f : [0, 1] \rightarrow \mathbb{R}$, outputs either $q \in \mathbb{Q} \cap [0, 1]$ where f is discontinuous, or $x \in [0, 1] \setminus \mathbb{Q}$ where f is continuous.*
- (g). *A functional that on input regulated $f, g : [0, 1] \rightarrow \mathbb{R}$, outputs a real $x \in [0, 1]$ such that f and g are both continuous or both discontinuous at x .*
- (h). *A functional that on input regulated $f, g : [0, 1] \rightarrow \mathbb{R}$ and weakly countable (or countable, or strongly countable) and dense $D \subset [0, 1]$, outputs either $d \in D \cap [0, 1]$ where f is discontinuous, or $x \in [0, 1] \setminus D$ where f is continuous.*
- (i). *A functional that on input regulated $f : [0, 1] \rightarrow [0, 1]$ with Riemann integral $\int_0^1 f(x)dx = 0$, outputs $x \in [0, 1]$ with $f(x) = 0$ (Bourbaki, [4* p. 61, Cor. 1]).*
- (j). *A functional that on input regulated $f : [0, 1] \rightarrow \mathbb{R}$, outputs $y \in (0, 1)$ where $F(x) := \lambda x. \int_0^x f(t)dt$ is differentiable with derivative equal to $f(y)$.*
- (k). *A functional that on input regulated $f : [0, 1] \rightarrow \mathbb{R}$ with only removable discontinuities, outputs $x \in [0, 1]$ which is **not** a strict³ local maximum.*

Proof. First of all, assume item (b) and let $X := \cup_{n \in \mathbb{N}} X_n$ be the union of finite sets $X_n \subset [0, 1]$ and define h as follows:

$$h(x) := \begin{cases} 0 & x \notin X \\ \frac{1}{2^{n+1}} & x \in X_n \text{ and } n \text{ is the least such number} \end{cases}. \quad (3)$$

Since $\cup_{k \leq n} X_k$ is finite for all $n \in \mathbb{N}$, we observe that h is regulated, in particular $0 = h(0+) = h(1-) = h(x+) = h(x-)$ for any $x \in (0, 1)$. Now, in case $h(a) > 0$, then $\{x \in [0, 1] : h(a) \leq h(x) \leq h(b)\}$ is finite by the definition of h . In case $h(a) = 0$, (3) implies $a \in [0, 1] \setminus \cup_{n \in \mathbb{N}} X_n$, and item (a) follows.

Secondly, assume item (c) and note that if $f : [0, 1] \rightarrow \mathbb{R}$ is continuous at $y \in [0, 1]$, we can use μ^2 to find $N \in \mathbb{N}$ such that $(\forall q \in B(y, \frac{1}{2^N}) \cap \mathbb{Q} \cap [0, 1])(|f(y) - f(q)| < \frac{1}{2})$; this readily yields $a, b \in [0, 1]$ as required by item (b). We note that item (e) immediately yields item (c), while the reversal readily follows by rescaling. Moreover, item (c) implies item (d); applying the latter to h as in (3), we observe that any point $y \in [0, 1]$ where h is quasi- or lower semi-continuous or Darboux, is such that $y \in [0, 1] \setminus \cup_{n \in \mathbb{N}} X_n$, i.e. item (a) follows.

Thirdly, assume item (f) and fix regulated $f : [0, 1] \rightarrow \mathbb{R}$. The following case distinction is decidable using $\exists^2$:

³ A point $x \in [0, 1]$ is a strict local maximum of $f : [0, 1] \rightarrow \mathbb{R}$ in case $(\exists N \in \mathbb{N})(\forall y \in B(x, \frac{1}{2^N}))(y \neq x \rightarrow f(y) < f(x))$.

- there is $q \in \mathbb{Q} \cap [0, 1]$ with $f(q+) = f(q) = f(q-)$, or
- for all $q \in \mathbb{Q} \cap [0, 1]$, we have $f(q+) \neq f(q) \vee f(q-) \neq f(q)$

In the first case, use μ^2 to find this rational. In the second case, the output of item (f) must be $x \in [0, 1] \setminus \mathbb{Q}$ such that f is continuous at x . In each case, we have a point of continuity for f , i.e. item (c) follows.

Fourth, assume item (a) and fix regulated $f : [0, 1] \rightarrow \mathbb{R}$. Note that μ^2 can find $q \in [0, 1] \cap \mathbb{Q}$ with $f(q+) \neq f(q) \vee f(q-) \neq f(q)$, if such rational exists. In case f is continuous at all $q \in [0, 1] \cap \mathbb{Q}$, let $(q_m)_{m \in \mathbb{N}}$ be an enumeration of $\mathbb{Q} \cap [0, 1]$. Now consider the following set

$$X_n := \{x \in (0, 1) : |f(x+) - f(x)| > \frac{1}{2^n} \vee |f(x-) - f(x)| > \frac{1}{2^n}\}, \quad (4)$$

which is finite by Footnote 2. Now use the strong Cantor realiser to find $y \in [0, 1] \setminus (\cup_{n \in \mathbb{N}} Y_n)$, where $Y_n = X_n \cup \{q_n\}$. Then f must be continuous at $y \in [0, 1] \setminus \mathbb{Q}$ and item (f) follows.

Fifth, assume item (i) and let h be as in (3). Since h is regulated, it is Riemann integrable, with $\int_0^1 h(x)dx = 0$. Any $y \in [0, 1]$ such that $h(y) = 0$ is by definition not in $\cup_{n \in \mathbb{N}} X_n$, yielding a strong Cantor realiser as in item (a). Item (c) readily yields item (i) as follows: if regulated $f : [0, 1] \rightarrow [0, 1]$ has $\int_0^1 f(x)dx = 0$ and is continuous at $y \in [0, 1]$, then we must have $f(y) = 0$; indeed, in case $f(y) > 0$, the continuity of f at y implies that $f(z) > \frac{1}{2^k}$ for $z \in B(y, \frac{1}{2^k})$ for some $k \in \mathbb{N}$, implying $\int_0^1 f(x)dx > 0$, since f is non-negative on $[0, 1]$.

Sixth, item (c) implies item (j) by the (second) fundamental theorem of calculus (see e.g. [30*Thm. 8.17, p. 165]). Item (j) applied to h as in (3) yields item (a) as $\int_0^x h(t)dt = 0$ for all $x \in [0, 1]$. Indeed, $h(x_0) = \int_0^{x_0} h(t)dt = 0$ implies $x_0 \in [0, 1] \setminus \cup_{n \in \mathbb{N}} X_n$ as required for a strong Cantor realiser.

Seventh, consider item (k) and note that h as in (3) is regulated with only removable discontinuities. Now, the set $X = \cup_{n \in \mathbb{N}} X_n$ consists of the local strict maxima of h , i.e. item (k) yields a strong Cantor realiser. For the reversal, $\exists^2$ computes a functional M such that $M(g, a, b)$ is a maximum of $g \in C([0, 1])$ on $[a, b] \subset [0, 1]$ (see [13*§3]), i.e. $(\forall y \in [a, b])(g(y) \leq g(M(g, a, b)))$. Using the functional M , one readily shows that ‘ x is a strict local maximum of g ’ is decidable⁴ given $\exists^2$, for g continuous on $[0, 1]$. Now let $f : [0, 1] \rightarrow \mathbb{R}$ be regulated and with only removable discontinuities. Use $\exists^2$ to define $\tilde{f} : [0, 1] \rightarrow \mathbb{R}$ as follows: $\tilde{f}(x) := f(x+)$ for $x \in [0, 1)$ and $\tilde{f}(1) = f(1-)$. By definition, $\tilde{f}$ is continuous on $[0, 1]$, and $\exists^2$ computes a (continuous) modulus of continuity, which follows in the same way as for Baire space (see e.g. [12*§4]). In case f is discontinuous at $x \in [0, 1]$, the latter point is a strict local maximum of f if and only if $f(x) > f(x+)$ (or $f(x) > f(x-)$ in case $x = 1$). Note that μ^2

⁴ If $g \in C([0, 1])$, then $x \in [0, 1]$ is a strict local maximum iff for some $\epsilon \in \mathbb{Q}^+$:

- $g(y) < g(x)$ whenever $|x - y| < \epsilon$ for any $q \in [0, 1] \cap \mathbb{Q}$, and:
- $\sup_{y \in [a, b]} g(y) < g(x)$ whenever $x \notin [a, b]$, $a, b \in \mathbb{Q}$ and $[a, b] \subset [x - \epsilon, x + \epsilon]$.

Note that μ^2 readily yields $N \in \mathbb{N}$ such that $(\forall y \in B(x, \frac{1}{2^N}))(g(y) < g(x))$.

(together with a modulus of continuity for $\tilde{f}$) readily yields $N_{f,x} \in \mathbb{N}$ such that $(\forall y \in B(x, \frac{1}{2^{N_{f,x}}})) (f(y) < f(x))$, in case x is a strict local maximum of f . In case f is continuous at $x \in [0, 1]$, we can use $\exists^2$ to decide whether x is a local strict maximum of $\tilde{f}$. By Footnote (see footnote 4), μ^2 again yields $N_{f,x} \in \mathbb{N}$ such that $(\forall y \in B(x, \frac{1}{2^N})) (\tilde{f}(y) < \tilde{f}(x))$, in case x is a strict local maximum of $\tilde{f}$. Now consider the following set:

$$A_n := \{x \in [0, 1] : x \text{ is a strict local maximum of } \tilde{f} \text{ or } f \text{ with } n \geq N_{f,x}\}.$$

Then A_n has finitely elements as strict local maxima cannot be ‘too close’. Hence, a strong Cantor realiser yields $y \in [0, 1] \setminus \bigcup_{n \in \mathbb{N}} A_n$, which is not a local maximum of f , i.e. item (k) follows.

Next, item (h) implies (f) as $D = \mathbb{Q}$ is trivially (weakly or strongly) countable. To derive item (h) from item (a), consider X_n as in (4) and assume $D = \bigcup_{n \in \mathbb{N}} D_n$ is dense, where D_n is finite for all $n \in \mathbb{N}$. Note that μ^2 can find $q \in [0, 1] \cap D \cap \mathbb{Q}$ with $f(q+) \neq f(q) \vee f(q-) \neq f(q)$, if such rational exists. In case f is continuous at all $q \in [0, 1] \cap D \cap \mathbb{Q}$, use a strong Cantor realiser to find $y \in [0, 1] \setminus (\bigcup_{n \in \mathbb{N}} Z_n)$, where $Z_n = X_n \cup D_n$. By definition, $y \in [0, 1] \setminus D$ and f is continuous at y .

Finally, assume item (g) and note that Thomae’s function from (1) is regulated as the left and right limits are 0. Hence, we obtain item (f), as Thomae’s function is continuous on $\mathbb{R} \setminus \mathbb{Q}$ and discontinuous on $\mathbb{Q}$. To obtain item (g) from a strong Cantor realiser, let $f, g : [0, 1]$ be regulated and consider the finite set X_n as in (4). Let Y_n be the same set for g . Then use a strong Cantor realiser to obtain $y \in [0, 1] \setminus (\bigcup_{n \in \mathbb{N}} (X_n \cup Z_n))$. By definition, f and g are continuous at y , i.e. item (g) follows. $\square$

Regarding item (k), one readily shows that the following functional is computationally equivalent to Ω from Theorem 15. Note that the set of strict local maxima is countable for *any* $\mathbb{R} \rightarrow \mathbb{R}$ -function by [31* p. 261, Theorem (1.1)].

- A functional that on input regulated $f : [0, 1] \rightarrow \mathbb{R}$ with only removable discontinuities, outputs an enumeration of all strict local maxima.

In conclusion, we have established a number of equivalences for strong Cantor realisers, while many more variations are possible.

3 Some Details of Kleene’s Computability Theory

Kleene’s computability theory borrows heavily from type theory and higher-order arithmetic. We briefly sketch some of the associated notions.

First of all, Kleene’s S1-S9 schemes define ‘ Ψ is computable in terms of Φ ’ for objects Φ, Ψ of any finite type. Now, the collection of *all finite types* $\mathbf{T}$ is defined by the following two clauses:

- (i) $0 \in \mathbf{T}$ and (ii) If $\sigma, \tau \in \mathbf{T}$ then $(\sigma \rightarrow \tau) \in \mathbf{T}$,

where 0 is the type of natural numbers, and $\sigma \rightarrow \tau$ is the type of mappings from objects of type σ to objects of type τ . In this way, $1 \equiv 0 \rightarrow 0$ is the type of

functions from numbers to numbers, and $n+1 \equiv n \rightarrow 0$. We view sets X of type σ objects as given by characteristic functions $F_X^{\sigma \rightarrow 0}$.

Secondly, for variables $x^\rho, y^\rho, z^\rho, \dots$ of any finite type $\rho \in \mathbf{T}$, types may be omitted when they can be inferred from context. The constants include the type 0 objects 0, 1 and $<_0, +_0, \times_0, =_0$ which are intended to have their usual meaning as operations on $\mathbb{N}$. Equality at higher types is defined in terms of ‘ $=_0$ ’ as follows: for any objects x^τ, y^τ , we have

$$[x =_\tau y] \equiv (\forall z_1^{\tau_1} \dots z_k^{\tau_k}) [x z_1 \dots z_k =_0 y z_1 \dots z_k], \quad (5)$$

if the type τ is composed as $\tau \equiv (\tau_1 \rightarrow \dots \rightarrow \tau_k \rightarrow 0)$.

Thirdly, we introduce the usual notations for common mathematical notions, like real numbers, as also can be found in [13].

Definition 17 (Real numbers and related notions).

- a. Natural numbers correspond to type zero objects, and we use ‘ n^0 ’ and ‘ $n \in \mathbb{N}$ ’ interchangeably. Rational numbers are defined as signed quotients of natural numbers, and ‘ $q \in \mathbb{Q}$ ’ and ‘ $<_{\mathbb{Q}}$ ’ have their usual meaning.
- b. Real numbers are coded by fast-converging Cauchy sequences $q_{(\cdot)} : \mathbb{N} \rightarrow \mathbb{Q}$, i.e. such that $(\forall n^0, i^0)(|q_n - q_{n+i}| <_{\mathbb{Q}} \frac{1}{2^n})$. We use Kohlenbach’s ‘hat function’ from [13*p. 289] to guarantee that every q^1 defines a real number.
- c. We write ‘ $x \in \mathbb{R}$ ’ to express that $x^1 := (q_{(\cdot)}^1)$ represents a real as in the previous item and write $[x](k) := q_k$ for the k -th approximation of x .
- d. Two reals x, y represented by $q_{(\cdot)}$ and $r_{(\cdot)}$ are *equal*, denoted $x =_{\mathbb{R}} y$, if $(\forall n^0)(|q_n - r_n| \leq 2^{-n+1})$. Inequality ‘ $<_{\mathbb{R}}$ ’ is defined similarly. We sometimes omit the subscript ‘ $\mathbb{R}$ ’ if it is clear from context.
- e. Functions $F : \mathbb{R} \rightarrow \mathbb{R}$ are represented by $\Phi^{1 \rightarrow 1}$ mapping equal reals to equal reals, i.e. extensionality as in $(\forall x, y \in \mathbb{R})(x =_{\mathbb{R}} y \rightarrow \Phi(x) =_{\mathbb{R}} \Phi(y))$.
- f. The relation ‘ $x \leq_\tau y$ ’ is defined as in (5) but with ‘ $\leq_0$ ’ instead of ‘ $=_0$ ’. Binary sequences are denoted ‘ $f^1, g^1 \leq_1 1$ ’, but also ‘ $f, g \in C$ ’ or ‘ $f, g \in 2^{\mathbb{N}}$ ’. Elements of Baire space are given by f^1, g^1 , but also denoted ‘ $f, g \in \mathbb{N}^{\mathbb{N}}$ ’.
- g. Sets of type ρ objects $X^{\rho \rightarrow 0}, Y^{\rho \rightarrow 0}, \dots$ are given by their characteristic functions $F_X^{\rho \rightarrow 0} \leq_{\rho \rightarrow 0} 1$, i.e. we write ‘ $x \in X$ ’ for $F_X(x) =_0 1$.

For completeness, we list the following notational convention for finite sequences.

Notation 18 (Finite sequences). The type for ‘finite sequences of objects of type ρ ’ is denoted ρ^* , which we shall only use for $\rho = 0, 1$. We shall not always distinguish between 0 and 0^* . Similarly, we assume a fixed coding for finite sequences of type 1 and shall make use of the type ‘ 1^* ’. In general, we do not always distinguish between ‘ s^ρ ’ and ‘ $\langle s^\rho \rangle$ ’, where the former is ‘the object s of type ρ ’, and the latter is ‘the sequence of type ρ^* with only element s^ρ ’. The empty sequence for the type ρ^* is denoted by ‘ $\langle \rangle_\rho$ ’, usually with the typing omitted.

Furthermore, we denote by $|s| = n$ the length of the finite sequence $s^{\rho^*} = \langle s_0^{\rho}, s_1^{\rho}, \dots, s_{n-1}^{\rho} \rangle$, where $|\langle \rangle| = 0$, i.e. the empty sequence has length zero. For sequences s^{ρ^*}, t^{ρ^*} , we denote by $s * t$ the concatenation of s and t , i.e. $(s * t)(i) = s(i)$ for $i < |s|$ and $(s * t)(j) = t(|s| - j)$ for $|s| \leq j < |s| + |t|$. For a sequence s^{ρ^*} , we define $\bar{s}N := \langle s(0), s(1), \dots, s(N - 1) \rangle$ for $N^0 < |s|$. For a sequence $\alpha^{0 \rightarrow \rho}$, we also write $\bar{\alpha}N = \langle \alpha(0), \alpha(1), \dots, \alpha(N - 1) \rangle$ for *any* N^0 . By way of shorthand, $(\forall q^{\rho} \in Q^{\rho^*})A(q)$ abbreviates $(\forall i^0 < |Q|)A(Q(i))$, which is (equivalent to) quantifier-free if A is.

Acknowledgement. We thank Anil Nerode for his valuable advice. Our research was supported by the *Deutsche Forschungsgemeinschaft* via the DFG grant SA3418/1-1. Initial results were obtained during the stimulating MFO workshop (ID 2046) on proof theory and constructive mathematics in Oberwolfach in early Nov. 2020. We express our gratitude towards the aforementioned institutions.

References

1. Appell, J., Banaś, J., Merentes, N.: Bounded Variation and Around. De Gruyter Series in Nonlinear Analysis and Applications, vol. 17. De Gruyter, Berlin (2014)
2. Avigad, J., Feferman, S.: Gödel's functional ("Dialectica") interpretation, handbook of proof theory. Stud. Logic Found. Math. **137**, 337–405 (1998)
3. Bourbaki, N.: Éléments de mathématique, Livre IV: Fonctions d'une variable réelle. (Théorie élémentaire), Actualités Sci. Ind., no. 1132, Hermann et Cie., Paris (1951). (French)
4. Bourbaki, N.: Functions of a Real Variable. Elements of Mathematics, Springer, Heidelberg (2004). <https://doi.org/10.1007/978-3-642-59315-4>
5. Feferman, S., Sieg, W.: Inductive definitions and subsystems of analysis. In: Iterated Inductive Definitions and Subsystems of Analysis: Recent Proof-Theoretical Studies. LNM, vol. 897, pp. 16–77. Springer, Heidelberg (1981). <https://doi.org/10.1007/BFb0091895>
6. Cantor, G.: Ueber eine Eigenschaft des Inbegriffs aller reellen algebraischen Zahlen. J. Reine Angew. Math. **77**, 258–262 (1874)
7. Gauld, D.: Did the young Volterra know about Cantor? Math. Mag. **66**(4), 246–247 (1993)
8. Gray, R.: Georg Cantor and transcendental numbers. Amer. Math. Monthly **101**(9), 819–832 (1994)
9. Hilbert, D.: Über das Unendliche. Mathematische Annalen **95**(1), 161–190 (1926). <https://doi.org/10.1007/BF01206605>
10. Hrbacek, K., Jech, T.: Introduction to Set Theory. Monographs and Textbooks in Pure and Applied Mathematics, 3rd edn, vol. 220. Marcel Dekker Inc., New York (1999)
11. Kleene, S.C.: Recursive functionals and quantifiers of finite types I. Trans. Amer. Math. Soc. **91**, 1–52 (1959)
12. Kohlenbach, U.: Foundational and mathematical uses of higher types. In: Reflections on the Foundations of Mathematics. Lecture Notes in Logic, vol. 15, pp. 92–116. ASL (2002)
13. Kohlenbach, U.: Higher order reverse mathematics. In: Reverse Mathematics 2001. Lecture Notes in Logic, vol. 21, pp. 281–295. ASL (2005)

14. Kunen, K.: Set Theory. Studies in Logic, vol. 34. College Publications, London (2011)
15. Longley, J., Normann, D.: Higher-Order Computability. Theory and Applications of Computability, Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-47992-6_9
16. Moll, V.H.: Numbers and Functions. Student Mathematical Library, vol. 65. American Mathematical Society (2012)
17. Normann, D.: Computability and non-monotone induction, p. 41 (2020). arXiv: <http://arxiv.org/abs/2006.03389>
18. Normann, D., Sanders, S.: Nonstandard analysis, computability theory, and their connections. J. Symbol. Logic **84**(4), 1422–1465 (2019)
19. Normann, D., Sanders, S.: The strength of compactness in computability theory and nonstandard analysis. Ann. Pure Appl. Logic **170**(11), 102710 (2019)
20. Normann, D., Sanders, S.: On the mathematical and foundational significance of the uncountable. J. Math. Logic (2019). <https://doi.org/10.1142/S0219061319500016>
21. Normann, D., Sanders, S.: Representations in measure theory. arXiv: <http://arxiv.org/abs/1902.02756> (2019, submitted)
22. Normann, D., Sanders, S.: Open sets in reverse mathematics and computability theory. J. Logic Comput. **30**(8), 40 (2020)
23. Normann, D., Sanders, S.: Pincherle's theorem in reverse mathematics and computability theory. Ann. Pure Appl. Logic **171**(5), 102788, 41 (2020)
24. Normann, D., Sanders, S.: The axiom of choice in computability theory and reverse mathematics. J. Logic Comput. **31**(1), 297–325 (2021)
25. Normann, D., Sanders, S.: On the uncountability of $\mathbb{R}$, J. Symb. Logic 41. arXiv: <http://arxiv.org/abs/2007.07560> (2022, to appear)
26. Normann, D., Sanders, S.: On robust theorems due to Bolzano, Weierstrass, and Cantor in Reverse Mathematics, p. 30. arxiv: <http://arxiv.org/abs/2102.04787> (2021, submitted)
27. Normann, D., Sanders, S.: Betwixt Turing and Kleene. In: Artemov, S., Nerode, A. (eds.) LFCS 2022. LNCS, vol. 13137, pp. 236–252. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-93100-1_15
28. Normann, D., Sanders, S.: On the computational properties of basic mathematical notions, p. 43. arXiv: <http://arxiv.org/abs/2203.05250> (2022, submitted)
29. Riemann, B.: Ueber die Darstellbarkeit einer Function durch eine trigonometrische Reihe, Abhandlungen der Königlichen Gesellschaft der Wissenschaften zu Göttingen, vol. 13. Habilitation thesis defended in 1854, published in 1867, p. 47
30. Rudin, W.: Real and Complex Analysis. Lecture Notes in Computer Science, Springer, Singapore (2018). https://doi.org/10.1007/978-981-13-2886-2_4
31. Saks, S.: Theory of the Integral. Dover Publications Inc., New York (1964)
32. Scheeffer, L.: Allgemeine Untersuchungen über Rectification der Curven. Acta Math. **5**(1), 49–82 (1884). (German)
33. Silva, C.E., Wu, Y.: No functions continuous only at points in a countable dense set. arXiv: <http://arxiv.org/abs/1809.06453v3> (2018, preprint)
34. Stephen, G.: Simpson, Subsystems of Second Order Arithmetic, 2nd edn. Perspectives in Logic, CUP (2009)
35. Soare, R.I.: Recursively Enumerable Sets and Degrees. Perspectives in Mathematical Logic, Springer, Heidelberg (1987)
36. Thomae, C.J.T.: Einleitung in die Theorie der bestimmten Integrale. Halle a.S., Louis Nebert (1875)

37. Vatsa, B.S., Vatsa, S.: Discrete Mathematics, 4th revised edn. New Age International, Delhi (2009)
38. Volterra, V.: Alcune osservazioni sulle funzioni punteggiate discontinue. *Giornale di matematiche* XIX, 76–86 (1881)
39. Wikipedia Contributors: Cantor's first set theory article. Wikipedia, The Free Encyclopedia (2020). http://en.wikipedia.org/wiki/Cantor%27s_first_set_theory_article



Mining the Surface: Witnessing the Low Complexity Theorems of Arithmetic

Amirhossein Akbar Tabatabai^(✉)

University of Groningen, Bernoulliborg, Nijenborgh 9,
9747 AG Groningen, The Netherlands
`s.a.akbar.tabatabai@rug.nl`

Abstract. One of the elegant achievements in the history of proof theory is the characterization of the provably total recursive functions of an arithmetical theory by its proof-theoretic ordinal as a way to measure the time complexity of the functions. Unfortunately, the characterization is not sufficiently fine-grained to capture the subclass of bounded functions with a bounded definition. In this paper, we provide a refined characterization to capture the feasibly definable and bounded functions that are provably total in Peano arithmetic. Roughly speaking, we identify the functions as the ones that are computable by a sequence of PV-provable polynomial time modifications on an initial polynomial time value, where the computational steps are indexed by the ordinals below ϵ_0 , decreasing by the modifications. The machinery we provide is quite general and applicable to any theory for which there is a natural ordinal analysis. However, for the sake of brevity, we limit ourselves only to Peano arithmetic and leave the general case to the future work.

Keywords: Total search problems • provably total recursive functions • ordinal analysis • Peano arithmetic

1 Introduction

One of the elegant achievements in the history of proof theory is the *witnessing* techniques that connect the provability of a formula of a certain form to the existence of an entity (algorithm [29], function [19], term in a type theory [1], etc.) that *witnesses* the truth of the formula. These connections are usually useful to establish the unprovability of a formula by showing that the corresponding witness does not exist. One of the well-known witnessing techniques is *ordinal analysis* that among many other things provides a characterization for the provably total recursive functions of some mathematical theories [11, 19, 24]. (For a comprehensive high-level explanation, see [26]). The main technique in ordinal analysis is connecting the provability of the totality of a Σ_1^0 -definable function to its time complexity, measured by the proof-theoretic ordinal of the theory. The characterization, then leads to some independence results for the

formulas in the form $A = \forall x \exists y B(x, y)$, where $B \in \Sigma_1^0$ is a definition of a function with a faster growth rate and hence higher time complexity than what the theory can actually reach [19]. There are, however, some classes of formulas for which the witnessing techniques and specially the one based on ordinal analysis break down. For instance, assume that the formula A has no existential quantifier (e.g. $A = \forall x B(x)$, where B is a quantifier-free formula), then it is not clear what the witnessing techniques must witness. Secondly, even in cases where we have some existential quantifiers, if the quantifiers are bounded (e.g. in formula $\forall x \exists y \leq t(x) B(x, y)$, where all quantifiers in B are also bounded), then the best thing that we can learn from ordinal analysis is the existence of an algorithm to compute y using a huge amount of time measured by the ordinal of the theory. This is much weaker than what we started with, i.e., the provability of the totality of the function with a *bounded* definition. The reason roughly is that the algorithm leads to the existence of the definition $\exists w C(x, w, y)$ for the function, where w encodes the computation and the theory proves $\forall x \exists y w C(x, w, y)$. However, the computation w can be huge and hence unbounded by the terms in the language and in this sense proving the totality of a *bounded* function with a *bounded* definition is stronger than the existence of such an algorithm.

To solve the issue, many new witnessing techniques were designed to address either the universal formulas via short propositional proofs [12, 18, 25] or provable bounded formulas in first-order bounded theories of arithmetic in special cases [10, 15, 23] and then in general cases [4, 5, 27], second-order bounded theories of arithmetic [3, 6, 16, 21] and even in Peano arithmetic [2]. Inspired by [2], in this paper, we provide a machinery to witness the PA-provable formulas in the form $A = \forall \bar{x} \exists \bar{y} B(\bar{x}, \bar{y})$, where B is a polynomial time computable predicate by an algorithm to compute $\bar{y}$ by a sequence of PV-provable polynomial time modifications on an initial polynomial time value, where the computational steps are indexed by the ordinals below ϵ_0 , decreasing by the modifications. The machinery we provide is quite general and applicable to any theory for which there is a natural ordinal analysis. However, for the sake of brevity, we limit ourselves only to Peano arithmetic and leave the general case for the subsequent works. The result is essentially the main theorem of [2]. However, we use a simpler methodology that is quite easy to generalize. The main idea behind [2] is rewriting the ordinal analysis of PA, i.e., the continuous cut elimination technique [8, 9], replacing all primitive recursive functions by more careful polynomial time computable operations [7]. In this paper, we use an indirect technique to circumvent the tedious ordinal analysis part. We use a refined yet PRA-type of ordinal analysis [20] as a *black box* and then providing a suitable feasible notation system for the ordinals, we will provide a way to transform a proof using primitive recursive functions and a weak form of transfinite induction to the sequences of PV-provable polynomial time modifications described above. The advantage of our approach is that it is easier to implement and avoids any redoing of ordinal analysis.

Here is the structure of the paper. In Sect. 2, we recall the basic definitions of different languages and arithmetical systems we use in this paper. In Sect. 3, we

recall the polynomial time ordinal notation system of [7] to which we add some observations on the feasibility of basic ordinal arithmetic. In Sect. 4, we introduce an auxiliary system axiomatized by transfinite induction on the universally quantified polynomial time computable formulas. We also cover its relation to the theories we are interested in and its sequent calculus. Finally, in Sect. 5, we present the witnessing technique to reduce the provability of the low complexity statements to a sequence of α many PV-implications, for some α below ϵ_0 . Using the usual witnessing theorem for the universal theory PV, we can prove the characterization that we have mentioned before.

2 Preliminaries

For any first-order language $\mathcal{L}$, by an $\mathcal{L}$ -formula, we mean any expression constructible by the connectives $\{\wedge, \vee, \forall, \exists\}$ from the atomic formulas (including $\perp$ and $\top$) and their negations. The formula $\neg A$ is defined via de Morgan laws and $A \rightarrow B$ is an abbreviation for $\neg A \vee B$. By $\bar{t}$, we mean a sequence of terms in the language and $\bar{x}$ means a sequence of variables.

To introduce the system PV, let us recall Cobham's machine-independent characterization of polynomial-time computable (*ptime*, for short) functions [17]. It states that a function is ptime iff it is constructible from certain basic functions by composition and a weak sort of recursion called the bounded recursion on notation. Any such construction provides an algorithm to compute the corresponding ptime function. Let $\mathcal{L}_{\text{PV}}$ be a first-order language with a function symbol for any such algorithm. In [18], Cook introduced an equational theory over the language $\mathcal{L}_{\text{PV}}$ to reason about ptime functions. The theory essentially consists of the defining axioms for the function symbols together with a sort of induction rule. Later, a conservative first-order extension of PV, denoted by PV_1 , was introduced [22]. The theory has the induction axiom $A(0) \wedge \forall x(A(\lfloor x/2 \rfloor) \rightarrow A(x)) \rightarrow \forall x A(x)$, for any quantifier-free formula $A(x)$ and is universally axiomatizable [22]. Note that as PV_1 is a universal theory, one may use Herbrand theorem followed by a gluing argument for the witnessing terms to prove that if $\text{PV}_1 \vdash \forall \bar{x} \exists y A(\bar{x}, y)$, for a quantifier-free formula $A(\bar{x}, y)$, then there is a term $t(\bar{x})$ in $\mathcal{L}_{\text{PV}}$ such that $\text{PV}_1 \vdash \forall \bar{x} A(\bar{x}, t(\bar{x}))$. In this paper, we will only use the theory PV_1 and not PV. Therefore, by abuse of notation, we will use the name PV to denote its first-order extension PV_1 .

In a similar fashion, using the construction of the primitive recursive functions by composition and primitive recursion on certain basic functions, it is possible to extend the language $\mathcal{L}_{\text{PV}}$ by a *fresh* function symbol for any primitive recursive function. Denote this new language by $\mathcal{L}_{\text{PRA}}$ and set the first-order theory PRA over $\mathcal{L}_{\text{PRA}}$ as PV extended by the defining axioms for the new functional symbols and the induction axiom $A(0) \wedge \forall x(A(x) \rightarrow A(x+1)) \rightarrow \forall x A(x)$, for any quantifier-free formula in the new language. This is of course different from the usual definition of PRA as its language is extended by the ptime function symbols in $\mathcal{L}_{\text{PV}}$ and the theory itself is extended by the theory PV. Moreover, the formula in the induction axiom of PRA may contain the symbols

from $\mathcal{L}_{PV}$. However, as the functions in the Cobham calculus are constructible as primitive recursive functions, it is clear that the separation of the primitive recursive function symbols and ptime function symbols is just a technical point and is totally immaterial. In fact, our presentation of PRA is a conservative extension of the usual PRA and hence has nothing essentially different from the usual PRA.

By Peano arithmetic, denoted by PA, we mean the theory PV extended by full induction axiom scheme $A(0) \wedge \forall x(A(x) \rightarrow A(x+1)) \rightarrow \forall xA(x)$, for *any* formula $A(x)$. This is also different from the usual definition of PA. However, as all of the function symbols in $\mathcal{L}_{PV}$ are definable in the usual language of PA and their functionality and totality are provable in the usual PA, it is easy to see that our PA is a conservative extension of the usual PA.

By Π_2^0 , we mean the class of $\mathcal{L}_{PV}$ -formulas in the form $\forall \bar{x} \exists \bar{y} A(\bar{x}, \bar{y})$, where any quantifier in $A(\bar{x}, \bar{y})$ is bounded. For two theories T and S and a class of formulas Φ , by $T \equiv_\Phi S$, we mean $T \vdash A$ iff $S \vdash A$, for any $A \in \Phi$.

Finally, let us recall some basics of the ordinal arithmetic. Apart from addition, multiplication and exponentiation of the ordinals, it is also possible to define subtraction $\div$ from *left* such that $\alpha \div \beta = 0$, if $\alpha < \beta$ and $\alpha \div \beta = \gamma$, if $\beta \leq \alpha$, where γ is the *unique* ordinal with the property that $\beta + \gamma = \alpha$. Similarly, it is possible to define the division d from *left* such that if $\beta \neq 0$, then $d(\alpha, \beta)$ is the *unique* γ such that $\alpha = \beta\gamma + \delta$, for some $\delta < \beta$.

3 A Polynomial-time Notation System for ϵ_0

In this section, we will recall the basics of the ptime notation system for the ordinals below ϵ_0 , introduced in [7]. Define $\mathcal{O}$ and $<$ inductively and simultaneously in the following way: $\mathcal{O}$ is the least set of expressions containing the empty string $\mathbf{0}$ and is closed under the operation $(\alpha_1, \dots, \alpha_n) \mapsto \omega^{\alpha_1} a_1 + \dots + \omega^{\alpha_n} a_n$, where $a_i \neq 0$ are natural numbers and $\alpha_n < \dots < \alpha_2 < \alpha_1$ and set $\omega^{\alpha_1} a_1 + \dots + \omega^{\alpha_n} a_n < \omega^{\beta_1} b_1 + \dots + \omega^{\beta_m} b_m$, if there exists $i \leq \min\{m, n\}$ such that $\alpha_j = \beta_j$ and $a_j = b_j$, for any $j \leq i$ and one of the following takes place:

- $i = n < m$,
- $i < \min\{m, n\}$ and $\alpha_{i+1} < \beta_{i+1}$
- $i < \min\{m, n\}$ and $\alpha_{i+1} = \beta_{i+1}$ and $a_i < b_i$.

Define $\alpha \leq \beta$ as $\alpha < \beta \vee \alpha = \beta$, where $=$ is the syntactical equality between the expressions. Using some efficient method of sequence encoding, it is possible to arithmetize the set $\mathcal{O}$ and the predicates $<$ and $\leq$. It is also possible to implement the arithmetization in a way that the length of the Gödel number of $\alpha \in \mathcal{O}$ is proportional to the number of symbols in the expression α . By this fact, [7] shows that both $\mathcal{O}$ and $<$ are polynomial time computable and hence formalizable in PV. (Technically, it uses a conservative extension of PV, but the difference does not affect us here). We fix *quantifier-free* predicates $\mathcal{O}(x)$ and $x < y$ to denote the formalized versions in the language $\mathcal{L}_{PV}$. We use small Greek letters to denote the elements of $\mathcal{O}$. Therefore, by $\forall \alpha A(\alpha)$, we actually mean $\forall x(\mathcal{O}(x) \rightarrow A(x))$.

In [7], it is shown that PV proves that $<$ is a total ordering on $\mathcal{O}$. It is clear that PV also proves that $\mathbf{0}$ is the minimum element of $\mathcal{O}$. It is also easy to see that the successor function over $\mathcal{O}$ mapping α to its successor, denoted by $\alpha + 1$, is a ptime function and $\text{PV} \vdash \forall \beta [\beta < \alpha + 1 \leftrightarrow \beta < \alpha \vee \beta = \alpha]$. We can also easily observe that there are ptime predicates defining whether an ordinal is 0, or a successor or a limit. This trichotomy is also provable in PV. There is a canonical function o mapping the number n to $\omega^{\mathbf{0}}n$, reinterpreting numbers as finite ordinals. Denote $\omega^{o(1)}$ by ω . Then, the map o is ptime and it is easy to prove in PV that o is an order-isomorphism, i.e., $\text{PV} \vdash \forall x [\mathcal{O}(o(x)) \wedge o(x) < \omega]$ and $\text{PV} \vdash \forall \alpha < \omega \exists ! y o(y) = \alpha$ and $\text{PV} \vdash x < y \leftrightarrow o(x) < o(y)$. Where there is no risk of confusion, we will use the numbers and their ordinal reinterpretations, interchangeably.

In the rest of this section, we will explain how to formalize the basic ordinal arithmetic in PV, using the aforementioned representation. For that purpose, first consider the following equalities for *non-zero* inputs. The operations with one zero input are trivial. These equalities make the computation of the addition, multiplication, subtraction from left and division from left possible, using the Cantor normal form of the ordinals. We will not provide a proof for these equalities as they are just simple computations, see [28].

$$\left(\sum_{i=1}^n \omega^{\alpha_i} a_i\right) + \left(\sum_{j=1}^m \omega^{\beta_j} b_j\right) = \begin{cases} \sum_{i=1}^n \omega^{\alpha_i} a_i + \sum_{j=1}^m \omega^{\beta_j} b_j & \alpha_n > \beta_1 \\ \sum_{j=1}^m \omega^{\beta_j} b_j & \alpha_1 < \beta_1 \\ \sum_{i=1}^k \omega^{\alpha_i} a_i + \sum_{j=1}^m \omega^{\beta_j} b_j & \alpha_{k+1} < \beta_1 < \alpha_k \\ \sum_{i=1}^{k-1} \omega^{\alpha_i} a_i + \omega^{\alpha_k} (a_k + b_1) + \sum_{j=2}^m \omega^{\beta_j} b_j & \alpha_k = \beta_1 \end{cases}$$

$$\left(\sum_{i=1}^n \omega^{\alpha_i} a_i\right) \div \left(\sum_{j=1}^m \omega^{\beta_j} b_j\right) = \begin{cases} 0 & \alpha_{k+1} < \beta_{k+1} \\ \sum_{i=k+1}^n \omega^{\alpha_i} a_i & \alpha_{k+1} > \beta_{k+1} \\ \omega^{\alpha_{k+1}} (a_{k+1} - b_{k+1}) + \sum_{i=k+2}^n \omega^{\alpha_i} a_i & \alpha_{k+1} = \beta_{k+1}, a_{k+1} > b_{k+1} \\ 0 & \alpha_{k+1} = \beta_{k+1}, a_{k+1} < b_{k+1} \end{cases}$$

where k is the maximum i such that $\alpha_i = \beta_i$ and $a_i = b_i$, if there is any and otherwise $k = 0$,

$$\left(\sum_{i=1}^n \omega^{\alpha_i} a_i\right) \left(\sum_{j=1}^m \omega^{\beta_j} b_j\right) = \begin{cases} \sum_{j=1}^m \omega^{\alpha_1 + \beta_j} b_j & \beta_m > 0 \\ \sum_{j=1}^{m-1} \omega^{\alpha_1 + \beta_j} b_j + \omega^{\alpha_1} a_1 b_m + \sum_{i=2}^n \omega^{\alpha_i} a_i & \beta_m = 0, m > 1 \\ \omega^{\alpha_1} a_1 b_1 + \sum_{i=2}^n \omega^{\alpha_i} a_i & \beta_m = 0, m = 1 \end{cases}$$

$$d\left(\sum_{i=1}^n \omega^{\alpha_i} a_i, \sum_{j=1}^m \omega^{\beta_j} b_j\right) = \begin{cases} 0 & \alpha_1 < \beta_1 \\ \sum_{i=1}^k \omega^{\alpha_i \dot{-} \beta_1} a_i & \alpha_1 \geq \beta_1, \alpha_k \neq \beta_1 \\ \sum_{i=1}^{k-1} \omega^{\alpha_i \dot{-} \beta_1} a_i + d(a_k, b_1) & \alpha_1 \geq \beta_1, \alpha_k = \beta_1, (*) \\ \sum_{i=1}^{k-1} \omega^{\alpha_i \dot{-} \beta_1} a_i + (d(a_k, b_1) - 1) & \text{otherwise} \end{cases}$$

where k is the greatest i such that $\alpha_i \geq \beta_1$, $d(a_k, b_1)$ is the quotient of a_k divided by b_1 and $(*)$ is the condition that $\sum_{i=k}^n \omega^{\alpha_i} a_i \geq \omega^{\alpha_k} b_1 d(a_k, b_1) + \sum_{j=2}^m \omega^{\beta_j} b_j$.

Note that to compute any of the operations, it is enough to do constant many comparisons and basic numerical computations, a search to find the maximum index that takes at most as long as the length of the inputs and at most m or n many applications of a ptime function. Hence, all the operations are ptime and hence representable in PV. It is also easy to see that PV proves the associativity of the addition and multiplication, left distributivity of multiplication over addition, the neutrality of 0 for the addition, the neutrality of 1 for multiplication, $0\alpha = \alpha 0 = 0$ and the defining axioms of $\div$ and d , i.e., $\alpha \div \beta = 0$, if $\alpha < \beta$, $\alpha = \beta + (\alpha \div \beta)$, if $\alpha \geq \beta$, $\beta d(\alpha, \beta) \leq \alpha$ and $\alpha \div \beta d(\alpha, \beta) < \beta$. The theory PV also proves that $<$ is preserved under the left addition and the left multiplication by a non-zero ordinal. Using this property, one can show that if $\alpha = \beta + \gamma = \beta + \delta$, then $\alpha \div \beta = \gamma = \delta$. This proves that the interval $[0, \alpha \div \beta]$ in $\mathcal{O}$ is in one-to-one correspondence with $[\beta, \alpha]$ via the map $\gamma \mapsto \beta + \gamma$. Similarly, if $\beta \neq 0$, $\alpha = \beta\gamma = \beta\delta$ implies $d(\alpha, \beta) = \gamma = \delta$.

In [20], a general setting for a refined version of ordinal analysis is described. It shows that it is possible to prove the Π_2^0 consequences of PA, using only PRA augmented by a weak form of transfinite induction stating that for any $\beta < \epsilon_0$, there is no primitive recursive decreasing sequence of ordinals below β . For more, see [20, 26].

Theorem 1. $\text{PA} \equiv_{\Pi_2^0} \text{PRA} + \bigcup_{\beta \in \mathcal{O}} \text{PRWO}(<_\beta)$, where $\text{PRWO}(<_\beta)$ is the scheme $\forall \bar{x} \exists y [f(\bar{x}, y+1) \nless f(\bar{x}, y) \vee \neg \mathcal{O}(f(\bar{x}, y)) \vee f(\bar{x}, y) \nless \beta]$, for any function symbol f in $\mathcal{L}_{\text{PRA}}$.

4 The System $\text{TI}(\mathbb{V}_1, <)$

In this section, we define an auxiliary system with a transfinite induction on the universal formulas in the language of PV. The system is powerful enough to interpret $\text{PRA} + \bigcup_{\beta \in \mathcal{O}} \text{PRWO}(<_\beta)$ and hence proves all Π_2^0 -theorems of PA. Later, we will provide a witnessing method for $\text{TI}(\mathbb{V}_1, <)$ that leads to the characterization of the low complexity theorems of PA that we are seeking.

Definition 1. Define $\mathbb{V}_1$ (resp., $\mathbb{E}_1$) as the least set of $\mathcal{L}_{\text{PV}}$ -formulas containing all atomic formulas and their negations and closed under conjunction, disjunction and universal (resp. existential) quantifiers.

Let $I\mathbb{V}_1$ (resp. $I\mathbb{E}_1$) be the theory extending PV by the $\mathbb{V}_1$ -induction (resp. $\mathbb{E}_1$ -induction) scheme, i.e., $A(0) \wedge \forall x(A(x) \rightarrow A(x+1)) \rightarrow \forall x A(x)$, for any $A(x) \in \mathbb{V}_1$ (resp. $A(x) \in \mathbb{E}_1$). Note that $I\mathbb{E}_1 = I\mathbb{V}_1$. The proof uses the usual technique of using $\mathbb{V}_1$ -induction on $B(x) = \neg A(y \div x)$ to prove $\mathbb{E}_1$ -induction on $A(y)$ and similarly for the other direction, see [13].

Lemma 1. For any primitive recursive function $f : \mathbb{N}^k \rightarrow \mathbb{N}$, there is a $\mathbb{E}_1$ -formula $D_f(\bar{x}, y)$ such that $I\mathbb{E}_1 \vdash \forall \bar{x} \exists! y D_f(\bar{x}, y)$ and $\mathbb{N} \models D_f(\bar{n}, m)$ iff $f(\bar{n}) = m$, for any $\bar{n}, m \in \mathbb{N}$.

Proof. By recursion on the construction of f , we define a quantifier-free formula $C_f(\bar{x}, w, y) \in \mathcal{L}_{PV}$ encoding that w is a computation of f with the input $\bar{x}$ and the output y . The cases for the basic functions and composition are easy. For the recursion case, if $f(\bar{x}, y)$ is defined via recursive equations $f(\bar{x}, 0) = g(\bar{x})$ and $f(\bar{x}, y + 1) = h(\bar{x}, y, f(\bar{x}, y))$, define $C_f(\bar{x}, y, \langle u, v \rangle, z)$ as $C_g(\bar{x}, u_0, v_0) \wedge \forall i \leq l(v) C_h(\bar{x}, i, v_i, u_{i+1}, v_{i+1}) \wedge v_{l(v)} = z$, where v encodes the sequence $\{f(\bar{x}, i)\}_{i=0}^{l(v)}$, the number $l(v)$ is the length of this sequence and u encodes the sequence of computations $\{u_i\}_{i=0}^{l(v)}$, where u_0 reads $\bar{x}$ and computes $v_0 = f(\bar{x}, 0)$ and u_{i+1} reads $\bar{x}$, i and $f(\bar{x}, i)$ and computes $f(\bar{x}, i + 1)$ via the function h . Note that the predicate $\forall i \leq l(v) C_h(\bar{x}, i, v_i, u_i, v_{i+1})$ is polynomial computable, as $l(v) \leq |v|$, where $|v|$ is the binary length of v . Hence there exists a polynomial time function symbol in PV like F such that PV proves that $F(\bar{x}, u, v) = 1$ iff $\forall i \leq l(v) C_h(\bar{x}, i, v_i, u_i, v_{i+1})$. Therefore, C_f can be written in a quantifier-free form. Now, set $D_f(\bar{x}, y) = \exists w C_f(\bar{x}, w, y)$. It is clear that $D_f \in \exists_1$ and $\mathbb{N} \models D_f(\bar{n}, m)$ iff $f(\bar{n}) = m$, for any $\bar{n}, m \in \mathbb{N}$. Finally, the proof of the claim that $I\exists_1 \vdash \forall \bar{x} \exists y D_f(\bar{x}, y)$ is similar to the similar claim in the representation of primitive recursive functions in $I\Sigma_1$.

Definition 2. Define the theory $\text{TI}(\mathbb{V}_1, <)$ over $\mathcal{L}_{PV}$ as the theory PV extended by the transfinite induction scheme $\forall \delta (\forall \gamma < \delta A(\gamma) \rightarrow A(\delta)) \rightarrow A(\theta)$, for any $A(\gamma) \in \mathbb{V}_1$ and any constant $\theta \in \mathcal{O}$.

Note that $\text{TI}(\mathbb{V}_1, <)$ extends the theory $I\mathbb{V}_1$ as $\text{TI}(\mathbb{V}_1, <)$ proves $\forall \delta < \omega (\forall \gamma < \delta A(\gamma) \rightarrow A(\delta)) \rightarrow \forall \delta < \omega A(\delta)$, for any $A \in \mathbb{V}_1$. Using the function o and the fact that it is an order-isomorphism between the numbers and the ordinals below ω , we will have $\forall x (\forall y < x A(y) \rightarrow A(x)) \rightarrow \forall x A(x)$ which implies $A(0) \wedge \forall x (A(x) \rightarrow A(x + 1)) \rightarrow \forall x A(x)$. Therefore, by Lemma 1, $\text{TI}(\mathbb{V}_1, <)$ represents any primitive recursive function with an $\exists_1$ -definition. As it is routine in arithmetic [13], this provides both $\forall_1$ and $\exists_1$ definitions for any atomic formula in $\mathcal{L}_{PRA}$. Hence, it is possible to interpret any $\forall_1$ -formula in $\mathcal{L}_{PRA}$ as an $\forall_1$ -formula in $\mathcal{L}_{PV}$. Using that interpretation, we can pretend that $\text{TI}(\mathbb{V}_1, <)$ has a fresh function symbol for any primitive recursive function and the $\forall_1$ -formulas in the new language are allowed in the transfinite induction. Moreover, we can also pretend that $\text{TI}(\mathbb{V}_1, <)$ extends the theory PRA. The reason simply is that the equational defining axioms in PRA are all provable in $I\mathbb{V}_1 = I\exists_1$ and hence in $\text{TI}(\mathbb{V}_1, <)$, as they are actually encoded in the definition D_f of f . For the quantifier-free induction of PRA, as we have seen before, it is possible to use the isomorphism o to prove the induction in $\text{TI}(\mathbb{V}_1, <)$.

Lemma 2. If $\text{PRA} + \bigcup_{\beta \in \mathcal{O}} \text{PRWO}(<_\beta) \vdash A$ then $\text{TI}(\mathbb{V}_1, <) \vdash A$, for any $A \in \mathcal{L}_{PV}$.

Proof. Pretend $\text{TI}(\mathbb{V}_1, <)$ has a function symbol for any primitive recursive function, allowed in the $\forall_1$ -formulas. As $\text{TI}(\mathbb{V}_1, <)$ extends PRA, it is enough to prove $\text{TI}(\mathbb{V}_1, <) \vdash \text{PRWO}(<_\beta)$, for any $\beta \in \mathcal{O}$. For the sake of contradiction, assume $\forall y [f(\bar{x}, y + 1) < f(\bar{x}, y) \wedge \mathcal{O}(f(\bar{x}, y)) \wedge f(\bar{x}, y) < \beta]$. Set

$B(\gamma, \bar{x}) = \forall y (f(\bar{x}, y) \neq \gamma)$ and note that $B(\gamma, \bar{x}) \in \mathbb{V}_1$. By transfinite induction, we prove $\forall \gamma < \beta B(\gamma, \bar{x})$. For that purpose, assume $\forall \delta < \gamma [\delta < \beta \rightarrow B(\delta, \bar{x})]$. Then, to prove $[\gamma < \beta \rightarrow B(\gamma, \bar{x})]$, if $f(\bar{x}, y) = \gamma$, for some $\gamma < \beta$, as $f(\bar{x}, y+1) < f(\bar{x}, y)$, we have $f(\bar{x}, y+1) < \gamma < \beta$. On the other hand, by $\forall \delta < \gamma [\delta < \beta \rightarrow B(\delta, \bar{x})]$, we know that none of the elements below γ is in the form of $f(\bar{x}, z)$, which contradicts with $f(\bar{x}, y+1) < \gamma$. Hence, $[\gamma < \beta \rightarrow B(\gamma, \bar{x})]$. Therefore, $\forall \delta < \gamma [\delta < \beta \rightarrow B(\delta, \bar{x})]$ implies $[\gamma < \beta \rightarrow B(\gamma, \bar{x})]$. Hence, by transfinite induction, we have $\forall \gamma < \beta B(\gamma, \bar{x})$ which for $\gamma = f(\bar{x}, 0) < \beta$ implies $\forall y (f(\bar{x}, y) \neq f(\bar{x}, 0))$ which is a contradiction.

Corollary 1. $\text{PA} \equiv_{\Pi_2^0} \text{TI}(\mathbb{V}_1, <)$.

Proof. One direction is a consequence of the fact that PA proves the transfinite induction for any formulas and hence extends $\text{TI}(\mathbb{V}_1, <)$. The other direction is a consequence of Theorem 1 and Lemma 2.

4.1 A Proof System for $\text{TI}(\mathbb{V}_1, <)$

We now present a sequent calculus for the theory $\text{TI}(\mathbb{V}_1, <)$. By a sequent over $\mathcal{L}_{\text{PV}}$, we mean an expression in the form $S = \Gamma \Rightarrow \Delta$, where Γ and Δ are multisets of formulas in $\mathcal{L}_{\text{PV}}$. Define **LPV** as the usual system **LK** augmented with the equality axioms for atomic formulas and their negations and the quantifier-free axioms of PV as the initial sequents:

Axioms:

$$\begin{array}{c}
 \frac{}{\perp \Rightarrow} \quad \frac{}{\Rightarrow \top} \\
 \\
 \frac{}{P \Rightarrow P} \quad \frac{}{\neg P \Rightarrow \neg P} \quad \frac{}{P, \neg P \Rightarrow} \quad \frac{}{\Rightarrow P, \neg P} \quad \frac{}{\Rightarrow A} \\
 \\
 \frac{}{\Rightarrow t = t} \\
 \\
 \frac{}{s_1 = t_1, \dots, s_n = t_n \Rightarrow f(\bar{s}) = f(\bar{t})} \\
 \\
 \frac{}{s_1 = t_1, \dots, s_n = t_n, Q(\bar{s}) \Rightarrow Q(\bar{t})}
 \end{array}$$

where P ranges over all atomic formulas, f ranges over all function symbols in the language, Q ranges over all atomic formulas or their negations, and A ranges over all quantifier-free axioms of PV.

Structural Rules:

$$\begin{array}{c}
 \frac{\Gamma, A, A \Rightarrow \Delta}{\Gamma, A \Rightarrow \Delta} Lc \quad \frac{\Gamma \Rightarrow A, A, \Delta}{\Gamma \Rightarrow A, \Delta} Rc \\
 \\
 \frac{\Gamma \Rightarrow \Delta}{\Gamma, A \Rightarrow \Delta} Lw \quad \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow A, \Delta} Rw
 \end{array}$$

$$\frac{\Gamma \Rightarrow A, \Delta \quad \Pi, A \Rightarrow A}{\Gamma, \Pi \Rightarrow \Delta, A} \text{ cut}$$

Logical Rules:

$$\begin{array}{c} i \in \{0, 1\} \frac{\Gamma, A_i \Rightarrow \Delta}{\Gamma, A_0 \wedge A_1 \Rightarrow \Delta} L\wedge \quad \frac{\Gamma \Rightarrow A, \Delta \quad \Gamma \Rightarrow B, \Delta}{\Gamma \Rightarrow A \wedge B, \Delta} R\wedge \\[10pt] \frac{\Gamma, A \Rightarrow \Delta \quad \Gamma, B \Rightarrow \Delta}{\Gamma, A \vee B \Rightarrow \Delta} L\vee \quad i \in \{0, 1\} \frac{\Gamma \Rightarrow A_i, \Delta}{\Gamma \Rightarrow A_0 \vee A_1, \Delta} R\vee \\[10pt] \frac{\Gamma, A(t) \Rightarrow \Delta}{\Gamma, \forall x A(x) \Rightarrow \Delta} L\forall \quad \frac{\Gamma \Rightarrow A(x), \Delta}{\Gamma \Rightarrow \forall x A(x), \Delta} R\forall \\[10pt] \frac{\Gamma, A(x) \Rightarrow \Delta}{\Gamma, \exists x A(x) \Rightarrow \Delta} L\exists \quad \frac{\Gamma \Rightarrow A(t), \Delta}{\Gamma \Rightarrow \exists x A(x), \Delta} R\exists \end{array}$$

In the rules ($R\forall$) and ($L\exists$), the variable x should not appear in the consequence. Adding the rule

$$\frac{\Gamma, \forall \gamma < \delta A(\gamma) \Rightarrow \Delta, A(\delta)}{\Gamma \Rightarrow \Delta, A(\theta)} \text{ Ind}$$

to **LPV**, we get $\mathbf{G}_0$. Note that in (Ind), the variable δ should not appear in the consequence. Moreover, the constant $\theta \in \mathcal{O}$ is arbitrary and can take any value. For more on the proof theory of first-order theories and specially arithmetic, see [13, 14].

By the usual cut reduction method [13, 14], it is easy to prove that for any $\Gamma \cup \Delta \subseteq \mathcal{V}_1$, if $\Gamma \Rightarrow \Delta$ is provable in $\text{TI}(\mathcal{V}_1, <)$ (resp., **PV**), then it has a $\mathbf{G}_0$ -proof (resp. **LPV**-proof) consisting only of $\mathcal{V}_1$ -formulas. For some practical reasons, we simplify the system $\mathbf{G}_0$ by changing the cut and the induction rules to the weak cut and weak induction rules, respectively:

$$\frac{\Gamma \Rightarrow A \quad A \Rightarrow \Delta}{\Gamma \Rightarrow \Delta} wCut \quad \frac{\Gamma, \forall \gamma < \delta A(\gamma) \Rightarrow \forall \gamma < \delta + 1 A(\gamma)}{\Gamma \Rightarrow A(\theta)} wInd$$

Denote this system by $\mathbf{G}_1$. Note that the difference between (Ind) and ($wInd$) is that in the latter Δ is omitted and $A(\delta)$ is replaced by $\forall \gamma < \delta + 1 A(\gamma)$.

Lemma 3. *For any $\Gamma \cup \Delta \subseteq \mathcal{V}_1$, if $\text{TI}(\mathcal{V}_1, <) \vdash \bigwedge \Gamma \rightarrow \bigvee \Delta$, then $\Gamma \Rightarrow \Delta$ has a $\mathbf{G}_1$ -proof only consisting of $\mathcal{V}_1$ -formulas.*

Proof. By a $\mathcal{V}_1$ -proof in $\mathbf{G}_1$ (resp. **LPV**), we mean a proof in $\mathbf{G}_1$ (resp. **LPV**) consisting only of $\mathcal{V}_1$ -proofs. We show that the cut rule and the induction rule (over $\mathcal{V}_1$ -formulas) are derivable in $\mathbf{G}_1$ (by a $\mathcal{V}_1$ -proof). We only investigate the harder case of $\mathcal{V}_1$ -proofs. The other is the same omitting the restrictions everywhere.

For cut, consider the following proof-tree in $\mathbf{G}_1$, where the double lines mean simple omitted proofs in $\mathbf{G}_1$. The tree proves $\Gamma, \Sigma \Rightarrow A, \Delta$ from $\Gamma \Rightarrow A, \Delta$ and $\Sigma, A \Rightarrow A$.

$$\begin{array}{c}
\frac{\frac{\frac{\Sigma \Rightarrow \bigwedge \Sigma}{\Gamma, \Sigma \Rightarrow \bigwedge \Sigma, \Delta}}{\Gamma, \Sigma \Rightarrow A \wedge \bigwedge \Sigma, \Delta}}{\Gamma, \Sigma \Rightarrow (A \wedge \bigwedge \Sigma) \vee \bigvee \Delta} \quad \frac{\frac{\Gamma \Rightarrow A, \Delta}{\Gamma, \Sigma \Rightarrow A, \Delta}}{\Gamma, \Sigma \Rightarrow A, \Delta} \quad \frac{\frac{\frac{\bigvee \Delta \Rightarrow \Delta}{\bigvee \Delta \Rightarrow A, \Delta}}{(A \wedge \bigwedge \Sigma) \vee \bigvee \Delta \Rightarrow A, \Delta}}{\Gamma, \Sigma \Rightarrow A, \Delta} \quad \frac{\frac{\frac{\Sigma, A \Rightarrow A}{A \wedge \bigwedge \Sigma \Rightarrow A}}{A \wedge \bigwedge \Sigma \Rightarrow A, \Delta}}{\Gamma, \Sigma \Rightarrow A, \Delta} \text{ } wCut
\end{array}$$

Note that the simulation of the cut rule in $\mathbf{G}_1$ implies that $\mathbf{G}_1$ is as powerful as $\mathbf{LPV}$. It also transforms a $\forall_1$ -proof in $\mathbf{LPV}$ to a $\forall_1$ -proof in $\mathbf{G}_1$. For the induction rule, consider the following proof-tree proving $\Gamma \Rightarrow A(\theta), \Delta$ from $\Gamma, \forall \gamma < \delta A(\gamma) \Rightarrow A(\delta), \Delta$:

$$\begin{array}{c}
\frac{\frac{\frac{\Gamma, \forall \gamma < \delta A(\gamma) \Rightarrow A(\delta), \Delta}{\Gamma, \forall \gamma < \delta A(\gamma) \Rightarrow [A(\delta) \vee \bigvee \Delta]} \quad \frac{\Gamma, \forall \gamma < \delta A(\gamma) \Rightarrow [A(\delta) \vee \bigvee \Delta]}{\Gamma, [\forall \gamma < \delta A(\gamma)] \vee \bigvee \Delta \Rightarrow [A(\delta) \vee \bigvee \Delta]} \quad *}{\Gamma, \forall \gamma < \delta [A(\gamma) \vee \bigvee \Delta] \Rightarrow [A(\delta) \vee \bigvee \Delta]} \quad ** \\
\frac{\Gamma, \forall \gamma < \delta [A(\gamma) \vee \bigvee \Delta] \Rightarrow \forall \gamma < \delta + 1 [A(\gamma) \vee \bigvee \Delta]}{\Gamma, \forall \gamma < \delta [A(\gamma) \vee \bigvee \Delta] \Rightarrow \forall \gamma < \delta + 1 [A(\gamma) \vee \bigvee \Delta]} \quad ** \\
\frac{\Gamma \Rightarrow A(\theta) \vee \bigvee \Delta}{\Gamma \Rightarrow A(\theta), \Delta} \quad \dagger \text{ } wInd
\end{array}$$

where $(*)$ is the result of a cut with the sequent $\forall \gamma < \delta [A(\gamma) \vee \bigvee \Delta] \Rightarrow [\forall \gamma < \delta A(\gamma)] \vee \bigvee \Delta$ which has a proof in $\mathbf{LPV}$ and hence a $\forall_1$ -proof in $\mathbf{LPV}$ and by the observation we have just made, a $\forall_1$ -proof in $\mathbf{G}_1$. Note that the use of cut is allowed as we showed its derivability in $\mathbf{G}_1$. Moreover, $(**)$ is the result of a cut with the PV-provable sequent $[A(\delta) \vee \bigvee \Delta], \forall \gamma < \delta [A(\gamma) \vee \bigvee \Delta] \Rightarrow \forall \gamma < \delta + 1 [A(\gamma) \vee \bigvee \Delta]$. The latter is provable in $\mathbf{LPV}$. Therefore, it has a $\forall_1$ -proof in $\mathbf{LPV}$ and hence in $\mathbf{G}_1$. Finally, $\dagger$ is the result of a cut with $A(\theta) \vee \bigvee \Delta \Rightarrow A(\theta), \Delta$ that has a trivial $\forall_1$ -proof.

5 Witnessing Flows

In this section, we first witness $\text{TI}(\forall_1, <)$ -provable implications between $\forall_1$ -formulas by a sequence of β many PV-provable implications, for some β below ϵ_0 . Then, we use the usual witnessing theorem for PV to provide the characterization that we claimed in Introduction.

Definition 3. Let $A(\bar{x}), B(\bar{x}) \in \forall_1$. A pair $(H(\gamma, \bar{x}), \beta)$ of a $\forall_1$ -formula and $\beta \in \mathcal{O}$ such that $\beta \geq 1$ is called a flow from $A(\bar{x})$ to $B(\bar{x})$, if:

- $\text{PV} \vdash A(\bar{x}) \leftrightarrow H(0, \bar{x})$.
- $\text{PV} \vdash \forall 1 \leq \delta \leq \beta [\forall \gamma < \delta H(\gamma, \bar{x}) \rightarrow H(\delta, \bar{x})]$.
- $\text{PV} \vdash H(\beta, \bar{x}) \leftrightarrow B(\bar{x})$.

We denote the existence of a flow from $A(\bar{x})$ to $B(\bar{x})$ by $A(\bar{x}) \triangleright B(\bar{x})$. For any multisets Γ and Δ of $\forall_1$ -formulas, by $\Gamma \triangleright \Delta$, we mean $\bigwedge \Gamma \triangleright \bigvee \Delta$.

In order to use flows to witness the proofs in $\text{TI}(\mathbb{V}_1, <)$, we will develop a high level calculus for this new notion, implemented in the following series of lemmas.

Lemma 4. *Let $A(\bar{x}), B(\bar{x}), C(\bar{x}) \in \mathbb{V}_1$. Then:*

- (i) *If $\text{PV} \vdash A(\bar{x}) \rightarrow B(\bar{x})$, then $A(\bar{x}) \triangleright B(\bar{x})$.*
- (ii) *If $A(\bar{x}) \triangleright B(\bar{x})$, then $A(\bar{x}) \circ C(\bar{x}) \triangleright B(\bar{x}) \circ C(\bar{x})$, for any $\circ \in \{\wedge, \vee\}$.*

Proof. For (i), set $\beta = 1$ and $H(\gamma, \bar{x}) = (\gamma = 0 \rightarrow A(\bar{x})) \wedge (\gamma = 1 \rightarrow B(\bar{x}))$. It is clear that $\text{PV} \vdash H(0, \bar{x}) \leftrightarrow A(\bar{x})$ and $\text{PV} \vdash H(1, \bar{x}) \leftrightarrow B(\bar{x})$. As $\text{PV} \vdash A(\bar{x}) \rightarrow B(\bar{x})$, we can see that $(H(\gamma, \bar{x}), \beta)$ is a flow from $A(\bar{x})$ to $B(\bar{x})$.

For (ii), we only prove the conjunction case. The disjunction case is similar. Since $A(\bar{x}) \triangleright B(\bar{x})$, by Definition 3, there exist an ordinal $\beta \geq 1$ and a formula $H(\gamma, \bar{x}) \in \mathbb{V}_1$ satisfying the conditions in Definition 3. Set $I(\gamma, \bar{x}) = H(\gamma, \bar{x}) \wedge C(\bar{x})$ and note that $I(\gamma, \bar{x}) \in \mathbb{V}_1$. It is easy to see that the pair $(I(\gamma, \bar{x}), \beta)$ is a flow from $A(\bar{x}) \wedge C(\bar{x})$ to $B(\bar{x}) \wedge C(\bar{x})$, as the PV-provability of $\forall 1 \leq \delta \leq \beta [\forall \gamma < \delta H(\gamma, \bar{x}) \rightarrow H(\delta, \bar{x})]$ implies the PV-provability of $\forall 1 \leq \delta \leq \beta [\forall \gamma < \delta (H(\gamma, \bar{x}) \wedge C(\bar{x})) \rightarrow (H(\delta, \bar{x}) \wedge C(\bar{x}))]$.

In the next lemma, we use the basic ordinal arithmetic, formalized in PV, to glue the flows together. The argument is general and applicable to any ordinal α , as long as the ordinals below α enjoys a ptime notation system that formalizes the operations $\{+, \div, \cdot, d\}$ and their basic properties in PV.

Lemma 5. (i) *If $A(\bar{x}) \triangleright B(\bar{x})$ and $B(\bar{x}) \triangleright C(\bar{x})$, then $A(\bar{x}) \triangleright C(\bar{x})$.*

(ii) *If $\Gamma, \forall \gamma < \delta A(\gamma, \bar{x}) \triangleright \forall \gamma < \delta + 1 A(\gamma, \bar{x})$, then $\Gamma \triangleright A(\theta, \bar{x})$, for any $\theta \in \mathcal{O}$.*

Proof. For (i), as $A(\bar{x}) \triangleright B(\bar{x})$, there exists a flow $(H(\gamma, \bar{x}), \beta)$ from $A(\bar{x})$ to $B(\bar{x})$. Similarly, as $B(\bar{x}) \triangleright C(\bar{x})$, there is a flow $(H'(\gamma, \bar{x}), \beta')$ from $B(\bar{x})$ to $C(\bar{x})$. Set $\beta'' = \beta + \beta'$ and $H''(\gamma, \bar{x}) = [\gamma \leq \beta \rightarrow H(\gamma, \bar{x})] \wedge [\beta < \gamma \leq \beta + \beta' \rightarrow H'(\gamma \div \beta, \bar{x})]$. We claim that the pair $(H''(\gamma, \bar{x}), \beta'')$ is a flow from $A(\bar{x})$ to $C(\bar{x})$. First, note that $H''(0, \bar{x})$ is PV-equivalent to $H(0, \bar{x})$ which is PV-equivalent to $A(\bar{x})$. Similarly, as $(\beta + \beta') \div \beta = \beta'$ is provable in PV, we know that $H''(\beta + \beta', \bar{x})$ is PV-equivalent to $H'(\beta', \bar{x})$ which is PV-equivalent to $C(\bar{x})$. To prove $\text{PV} \vdash \forall 1 \leq \delta \leq \beta'' [\forall \gamma < \delta H''(\gamma, \bar{x}) \rightarrow H''(\delta, \bar{x})]$, note that if $\delta \leq \beta$, then the claim reduces to the same claim for $H(\gamma, \bar{x})$ which is provable. If $\beta < \delta \leq \beta + \beta'$, assume $\forall \gamma < \delta H''(\gamma, \bar{x})$ to prove $H''(\delta, \bar{x})$ or equivalently $H'(\delta \div \beta, \bar{x})$. Note that $\forall \gamma < \delta H''(\gamma, \bar{x})$ implies $\forall \beta \leq \gamma < \delta H''(\gamma, \bar{x})$. As the interval $(0, \delta \div \beta)$ is isomorphic to (β, δ) , by the map $\gamma \mapsto \beta + \gamma$, then $\forall \beta \leq \gamma < \delta H''(\gamma, \bar{x})$ implies $\forall 0 < \gamma < \delta \div \beta H''(\beta + \gamma, \bar{x})$ which implies $\forall 0 < \gamma < \delta \div \beta H'(\gamma, \bar{x})$. On the other hand, $\forall \beta \leq \gamma < \delta H''(\gamma, \bar{x})$ implies $H''(\beta, \bar{x})$ which is PV-equivalent to $H(\beta, \bar{x})$, by definition. As $H(\beta, \bar{x})$ is PV-equivalent to $B(\bar{x})$ which is also PV-equivalent to $H'(0, \bar{x})$, we can claim that $H(\beta, \bar{x})$ and $H'(0, \bar{x})$ are PV-equivalent. Hence, $\forall \beta \leq \gamma < \delta H''(\gamma, \bar{x})$ implies $\forall \gamma < \delta \div \beta H'(\gamma, \bar{x})$ which also implies $H'(\delta \div \beta, \bar{x})$, as $(\beta', H'(\gamma, \bar{x}))$ is a flow.

For (ii), as $\bigwedge \Gamma \wedge \forall \gamma < \delta A(\gamma, \bar{x}) \triangleright \forall \gamma < \delta + 1 A(\gamma, \bar{x})$, by Lemma 4, we have $\bigwedge \Gamma \wedge \forall \gamma < \delta A(\gamma, \bar{x}) \triangleright \bigwedge \Gamma \wedge \forall \gamma < \delta + 1 A(\gamma, \bar{x})$. Set $B(\delta, \bar{x}) = \bigwedge \Gamma \wedge \forall \gamma < \delta A(\gamma, \bar{x})$ and let $(H(\eta, \delta, \bar{x}), \beta)$ be the flow from $B(\delta, \bar{x})$ to $B(\delta + 1, \bar{x})$. Note

that $H(0, \delta, \bar{x})$ is PV-equivalent to $B(\delta, \bar{x})$ and $H(\beta, \delta, \bar{x})$ is PV-equivalent to $H(0, \delta + 1, \bar{x})$, as both are PV-equivalent to $B(\delta + 1)$. Define $\beta' = \beta \times (\theta + 1)$ and $I(\tau, \bar{x}) = H(\tau \div \beta d(\tau, \beta), d(\tau, \beta), \bar{x})$ and note that $I(\tau, \bar{x}) \in \mathcal{V}_1$. We show that $(I(\tau, \bar{x}), \beta')$ is a flow from $B(0, \bar{x})$ to $B(\theta + 1, \bar{x})$. Note that $(I(\tau, \bar{x}), \beta')$ is nothing but the result of gluing the flows $(H(\eta, \delta, \bar{x}), \beta)$, for all $\delta < \theta + 1$, one after another as depicted in the following figure, (for simplicity, in the figures, we drop the free variables $\bar{x}$).

$$\begin{array}{ccccccc}
 B(0) & \xrightarrow{\quad\quad\quad} & B(1) & \xrightarrow{\quad\quad\quad} & \cdots & \xrightarrow{\quad\quad\quad} & B(\theta + 1) \\
 \text{III} & & \text{III} & & & & \text{III} \\
 H(0, 0) & \longrightarrow & H(1, 0) & \longrightarrow & \cdots & \longrightarrow & H(\beta, 0) \equiv H(0, 1) \longrightarrow \cdots \longrightarrow H(0, \theta + 1) \\
 \text{III} & & \text{III} & & \text{III} & & \text{III} \\
 I(0) & \longrightarrow & I(1) & \longrightarrow & \cdots & \longrightarrow & I(\beta) \longrightarrow \cdots \longrightarrow I(\beta(\theta + 1))
 \end{array}$$

First, as $d(0, \beta) = 0$ and $0 \div \beta d(0, \beta) = 0$, provably in PV, we know that $I(0, \bar{x})$ is PV-equivalent to $H(0, 0, \bar{x})$ which is itself PV-equivalent to $B(0, \bar{x})$. Secondly, as $d(\beta \times (\theta + 1), \beta) = \theta + 1$ and $\beta \times (\theta + 1) \div \beta d(\beta \times (\theta + 1), \beta) = 0$, provably in PV, we know that $I(\beta \times (\theta + 1), \bar{x})$ is PV-equivalent to $H(0, \theta + 1, \bar{x})$ which is PV-equivalent to $B(\theta + 1, \bar{x})$. For the middle condition, we must prove $\text{PV} \vdash \forall 1 \leq \tau \leq \beta \times (\theta + 1) [\forall \zeta < \tau I(\zeta, \bar{x}) \rightarrow I(\tau, \bar{x})]$. There are two cases to consider, either $\beta d(\tau, \beta) + 1 \leq \tau$ or $\beta d(\tau, \beta) = \tau$. If $\beta d(\tau, \beta) + 1 \leq \tau$, then $\tau = \beta d(\tau, \beta) + \mu$ such that $\mu = \tau \div \beta d(\tau, \beta) \geq 1$. As for any $\eta < \mu$, we have $\beta d(\tau, \beta) + \eta < \tau$, we know that $\forall \zeta < \tau I(\zeta, \bar{x})$ implies $\forall \eta < \mu H(\eta, d(\tau, \beta), \bar{x})$. As we have $\mu \geq 1$, the latter proves $H(\mu, d(\tau, \beta), \bar{x})$ which is PV-equivalent to $I(\tau, \bar{x})$.

$$\begin{array}{ccccccc}
 \cdots & \longrightarrow & H(0, d(\tau, \beta)) & \longrightarrow & H(1, d(\tau, \beta)) & \longrightarrow & \cdots \longrightarrow H(\mu, d(\tau, \beta)) \longrightarrow \cdots \\
 & & \text{III} & & \text{III} & & \text{III} \\
 \cdots & \longrightarrow & I(\beta d(\tau, \beta)) & \longrightarrow & I(\beta d(\tau, \beta) + 1) & \longrightarrow & \cdots \longrightarrow I(\tau) \longrightarrow \cdots
 \end{array}$$

If $\beta d(\tau, \beta) = \tau$, we should use $\forall \zeta < \tau I(\zeta, \bar{x})$ to prove $I(\tau, \bar{x}) = H(0, d(\tau, \beta), \bar{x})$. Again, there are two cases to consider: either $d(\tau, \beta)$ is a successor or a limit ordinal. If $d(\tau, \beta) = \rho + 1$, for some ρ , as $H(0, \rho + 1, \bar{x})$ is PV-equivalent to $H(\beta, \rho, \bar{x})$, it is enough to prove $H(\beta, \rho, \bar{x})$. As $\beta \rho + \eta < \beta \rho + \beta = \beta(\rho + 1) = \tau$, for any $\eta < \beta$, we know that $\forall \zeta < \tau I(\zeta, \bar{x})$ implies $\forall \eta < \beta H(\eta, \rho, \bar{x})$ which implies $H(\beta, \rho, \bar{x})$.

$$\begin{array}{ccccccc}
 \cdots & \longrightarrow & H(0, \rho) & \longrightarrow & H(1, \rho) & \longrightarrow & \cdots \longrightarrow H(\beta, \rho) \equiv H(0, \rho + 1) \longrightarrow \cdots \\
 & & \text{III} & & \text{III} & & \text{III} \\
 \cdots & \longrightarrow & I(\beta \rho) & \longrightarrow & I(\beta \rho + 1) & \longrightarrow & \cdots \longrightarrow I(\beta(\rho + 1)) \longrightarrow \cdots
 \end{array}$$

If $d(\tau, \beta)$ is a limit ordinal, then $\forall \zeta < \beta d(\tau, \beta) I(\zeta, \bar{x})$ implies $\forall \delta < d(\tau, \beta) H(0, \delta, \bar{x})$ which implies $\forall \delta < d(\tau, \beta) B(\delta, \bar{x})$. The latter is $\forall \delta <$

$d(\tau, \beta)[\bigwedge \Gamma \wedge \forall \gamma < \delta A(\gamma, \bar{x})]$ that implies $\bigwedge \Gamma \wedge \forall \gamma < d(\tau, \beta) A(\gamma, \bar{x})$ as $d(\tau, \beta)$ is a limit ordinal. The latter is PV-equivalent to $H(0, d(\tau, \beta), \bar{x}) = I(\tau, \bar{x})$. This completes the proof of the claim and shows that $B(0, \bar{x}) \triangleright B(\theta + 1, \bar{x})$. Now, as $\text{PV} \vdash \bigwedge \Gamma \rightarrow \bigwedge \Gamma \wedge \forall \gamma < 0 A(\gamma, \bar{x})$ and $\text{PV} \vdash \bigwedge \Gamma \wedge \forall \gamma < \theta + 1 A(\gamma, \bar{x}) \rightarrow A(\theta, \bar{x})$. By Lemma 4, $\bigwedge \Gamma \triangleright \bigwedge \Gamma \wedge \forall \gamma < 0 A(\gamma, \bar{x})$ and $\bigwedge \Gamma \wedge \forall \gamma < \theta + 1 A(\gamma, \bar{x}) \triangleright A(\theta, \bar{x})$. Hence, by part (i), we have $\bigwedge \Gamma \triangleright A(\theta, \bar{x})$ which completes the proof.

Lemma 6. (*Conjunction and Disjunction Rules*)

- (i) If $\Gamma, A \triangleright \Delta$ or $\Gamma, B \triangleright \Delta$, then $\Gamma, A \wedge B \triangleright \Delta$.
- (ii) If $\Gamma \triangleright A, \Delta$ and $\Gamma \triangleright B, \Delta$, then $\Gamma \triangleright A \wedge B, \Delta$.
- (iii) If $\Gamma \triangleright A, \Delta$ or $\Gamma \triangleright B, \Delta$, then $\Gamma \triangleright A \vee B, \Delta$.
- (iv) If $\Gamma, A \triangleright \Delta$ and $\Gamma, B \triangleright \Delta$, then $\Gamma, A \vee B \triangleright \Delta$.

Proof. For (i) and (iii), as the implications $[(\bigwedge \Gamma \wedge (A \wedge B)) \rightarrow (\bigwedge \Gamma \wedge A)]$, $[(\bigwedge \Gamma \wedge (A \wedge B)) \rightarrow (\bigwedge \Gamma \wedge B)]$, $[(\bigvee \Delta \vee A) \rightarrow (\bigvee \Delta \vee (A \vee B))]$ and $[(\bigvee \Delta \vee B) \rightarrow (\bigvee \Delta \vee (A \vee B))]$ are all provable in PV, using Lemma 4 and Lemma 5, we reach what we wanted. For (ii), if $\Gamma \triangleright \Delta, A$ then $\bigwedge \Gamma \triangleright \bigvee \Delta \vee A$, by definition. By Lemma 4, we reach $\bigwedge \Gamma \triangleright (\bigvee \Delta \vee A) \wedge \bigwedge \Gamma$. Similarly, we have $\bigwedge \Gamma \triangleright \bigvee \Delta \vee B$ and by Lemma 4, we reach $\bigwedge \Gamma \wedge (\bigvee \Delta \vee A) \triangleright (\bigvee \Delta \vee B) \wedge (\bigvee \Delta \vee A)$. Therefore, $\bigwedge \Gamma \triangleright (\bigvee \Delta \vee B) \wedge (\bigvee \Delta \vee A)$, by part (i) in Lemma 5. Finally, as $(\bigvee \Delta \vee B) \wedge (\bigvee \Delta \vee A) \rightarrow \bigvee \Delta \vee (A \wedge B)$ is provable in PV, by Lemma 4 and Lemma 5, we reach $\bigwedge \Gamma \triangleright \bigvee \Delta \vee (A \wedge B)$. The proof for (iv) is similar.

Having the required lemmas, we are now ready to prove the following theorem as the main extraction technique that witnesses the proofs in $\text{TI}(\mathbb{V}_1, <)$ by flows.

Theorem 2. Let $\Gamma(\bar{x}) \cup \Delta(\bar{x}) \subseteq \mathbb{V}_1$. Then, $\text{TI}(\mathbb{V}_1, <) \vdash \bigwedge \Gamma(\bar{x}) \rightarrow \bigvee \Delta(\bar{x})$ iff $\Gamma(\bar{x}) \triangleright \Delta(\bar{x})$.

Proof. We first prove the easier direction. Assume $\Gamma(\bar{x}) \triangleright \Delta(\bar{x})$ and $(H(\gamma, \bar{x}), \beta)$ is a flow from $\bigwedge \Gamma(\bar{x})$ to $\bigvee \Delta(\bar{x})$. As $\text{PV} \vdash \forall 1 \leq \delta \leq \beta [\forall \gamma < \delta H(\gamma, \bar{x}) \rightarrow H(\delta, \bar{x})]$ and $\text{TI}(\mathbb{V}_1, <)$ extends PV, we have $\text{TI}(\mathbb{V}_1, <) \vdash \forall 1 \leq \delta \leq \beta [\forall \gamma < \delta H(\gamma, \bar{x}) \rightarrow H(\delta, \bar{x})]$. Then, as $H(\gamma, \bar{x}) \in \mathbb{V}_1$, by the transfinite induction in $\text{TI}(\mathbb{V}_1, <)$, we reach $\text{TI}(\mathbb{V}_1, <) \vdash H(0, \bar{x}) \rightarrow H(\beta, \bar{x})$. Finally, using the PV-provable equivalences $\bigwedge \Gamma(\bar{x}) \leftrightarrow H(0, \bar{x})$ and $H(\beta, \bar{x}) \leftrightarrow \bigvee \Delta(\bar{x})$, we reach $\text{TI}(\mathbb{V}_1, <) \vdash \bigwedge \Gamma(\bar{x}) \rightarrow \bigvee \Delta(\bar{x})$.

For the other direction, assume $\text{TI}(\mathbb{V}_1, <) \vdash \bigwedge \Gamma(\bar{x}) \rightarrow \bigvee \Delta(\bar{x})$. By Lemma 3, $\Gamma(\bar{x}) \Rightarrow \Delta(\bar{x})$ has a $\mathbf{G}_1$ -proof only consisting of $\mathbb{V}_1$ -formulas. By induction on this proof, we show that for any sequent $\Sigma \Rightarrow \Lambda$ in the proof, we have $\Sigma \triangleright \Lambda$.

For the axioms, as they are provable in PV, using Lemma 4, there is nothing to prove. The case of structural rules (except for the weak cut) is easy. Weak cut and weak induction are addressed in Lemma 5. The conjunction and disjunction rules are proved in Lemma 6. For the right universal quantifier rule, if $\Sigma(\bar{x}) \Rightarrow \Lambda(\bar{x}), \forall z B(\bar{x}, z)$ is proved from $\Sigma(\bar{x}) \Rightarrow \Lambda(\bar{x}), B(\bar{x}, z)$, then by induction hypothesis, $\Sigma(\bar{x}) \triangleright \Lambda(\bar{x}), B(\bar{x}, z)$. Therefore, there exists a flow $(H(\gamma, \bar{x}, z), \beta)$ from $\bigwedge \Sigma(\bar{x})$ to $B(\bar{x}, z) \vee \bigvee \Lambda(\bar{x})$. Define

$I(\gamma, \bar{x}) = \forall z H(\gamma, \bar{x}, z)$ and note that $I(\bar{x}, z) \in \forall_1$. It is easy to see that $(I(\gamma, \bar{x}), \beta)$ is a flow from $\forall z[\bigwedge \Sigma(\bar{x})]$ to $\forall z[B(\bar{x}, z) \vee \bigvee \Lambda(\bar{x})]$, as PV-provability of $\forall \gamma < \delta H(\gamma, z, \bar{x}) \rightarrow H(\delta, z, \bar{x})$ implies the PV-provability of $\forall \gamma < \delta \forall z H(\gamma, z, \bar{x}) \rightarrow \forall z H(\delta, z, \bar{x})$. Finally, as z does not occur as a free variable in $\Sigma(\bar{x}) \cup \Lambda(\bar{x})$, we have the PV-equivalence between $\forall z[\bigwedge \Sigma(\bar{x})]$ and $\bigwedge \Sigma(\bar{x})$ and similarly between $\forall z[B(\bar{x}, z) \vee \bigvee \Lambda(\bar{x})]$ and $\bigvee \Lambda(\bar{x}) \vee \forall z B(\bar{x}, z)$. Using Lemma 4 and Lemma 5, we can prove $\bigwedge \Sigma(\bar{x}) \triangleright \bigvee \Lambda(\bar{x}) \vee \forall z B(\bar{x}, z)$. For the left universal quantifier rule, if $\Sigma(\bar{x}), \forall z B(\bar{x}, z) \Rightarrow \Lambda(\bar{x})$ is proved from $\Sigma(\bar{x}), B(\bar{x}, s(\bar{x})) \Rightarrow \Lambda(\bar{x})$, then by induction hypothesis $\Sigma(\bar{x}), B(\bar{x}, s(\bar{x})) \triangleright \Lambda(\bar{x})$. Since $\text{PV} \vdash \bigwedge \Sigma(\bar{x}) \wedge \forall z B(\bar{x}, z) \rightarrow \bigwedge \Sigma(\bar{x}) \wedge B(\bar{x}, s(\bar{x}))$, by Lemma 4 and Lemma 5, we reach $\Sigma(\bar{x}), \forall z B(\bar{x}, z) \triangleright \Lambda(\bar{x})$.

Corollary 2. $\text{PA} \vdash \bigwedge \Gamma(\bar{x}) \rightarrow \bigvee \Delta(\bar{x})$ iff $\Gamma(\bar{x}) \triangleright \Delta(\bar{x})$, for $\Gamma(\bar{x}) \cup \Delta(\bar{x}) \subseteq \forall_1$.

Proof. As any implication in the form $\bigwedge \Gamma(\bar{x}) \rightarrow \bigvee \Delta(\bar{x})$ is logically equivalent to a Π_2^0 -formula, the claim is a consequence of Theorem 2 and Corollary 1.

Finally, we can use the usual witnessing theorem of PV to push the data extraction of Corollary 2, a bit further. But first, we need two definitions.

Definition 4. Let T be a theory over the language $\mathcal{L}_{\text{PV}}$. A total search problem of T is a quantifier-free formula $A(\bar{x}, \bar{y})$ such that $T \vdash \forall \bar{x} \exists \bar{y} A(\bar{x}, \bar{y})$. A total search problem is called an NP-search problem, if there are sequences of polynomials $\bar{r}$ such that $\text{PV} \vdash A(\bar{x}, \bar{y}) \rightarrow |\bar{y}| \leq \bar{r}(|\bar{x}|)$, where $|\bar{y}| \leq \bar{r}(|\bar{x}|)$ is an abbreviation for $\bigwedge_i (|y_i| \leq r_i(|\bar{x}|))$. We denote the class of all these total search (NP-search) problems of T by $\text{TSP}(T)$ (resp. $\text{TFNP}(T)$).

Definition 5. Let $A(\bar{x}, \bar{y})$ be a quantifier-free formula in $\mathcal{L}_{\text{PV}}$ and $\beta \in \mathcal{O}$. By a $\text{LS}(<_\beta)$ -program for $A(\bar{x}, \bar{y})$, we mean the following data: an initial sequence of $\mathcal{L}_{\text{PV}}$ -terms $\bar{i}(\bar{x})$, a quantifier-free $\mathcal{L}_{\text{PV}}$ -formula $G(\gamma, \bar{x}, \bar{z})$, a sequence of $\mathcal{L}_{\text{PV}}$ -terms $\bar{N}(\gamma, \bar{x}, \bar{z})$, an $\mathcal{L}_{\text{PV}}$ -term $q(\gamma, \bar{x}, \bar{z})$, a sequence of $\mathcal{L}_{\text{PV}}$ -terms $\bar{p}(\bar{x}, \bar{z})$, such that:

- $\text{PV} \vdash G(\beta, \bar{x}, \bar{i}(\bar{x}))$,
- $\text{PV} \vdash \gamma \neq 0 \rightarrow q(\gamma, \bar{x}, \bar{z}) < \gamma$,
- $\text{PV} \vdash \gamma \neq 0 \rightarrow [G(\gamma, \bar{x}, \bar{z}) \rightarrow G(q(\gamma, \bar{x}, \bar{z}), \bar{x}, \bar{N}(\gamma, \bar{x}, \bar{z}))]$,
- $\text{PV} \vdash G(0, \bar{x}, \bar{z}) \rightarrow A(\bar{x}, \bar{p}(\bar{x}, \bar{z}))$.

By $\text{LS}(\leq_\beta)$, we mean the class of all formulas $A(\bar{x}, \bar{y})$ for which there exists a $\text{LS}(\leq_\beta)$ -program. By $\text{PLS}(\leq_\beta)$, we mean the class $\text{LS}(\leq_\beta) \cap \text{TFNP}(\text{Th}(\mathbb{N}))$.

It is easy to see that $A(\bar{x}, \bar{y}) \in \text{LS}(\leq_\beta)$ implies $\forall \bar{x} \exists \bar{y} A(\bar{x}, \bar{y})$ and the $\text{LS}(\leq_\beta)$ -program actually provides an algorithm to compute $\bar{y}$ from $\bar{x}$. Denoting $G(\gamma, \bar{x}, \bar{z})$ by G_γ , the algorithm starts at the level β with an initial value $\bar{i}(\bar{x})$ satisfying the property G_β . Then, using the feasible function q , it finds a lower level to go to and uses the modification $\bar{N}$ to update any value with the property G_γ to a value satisfying the property $G_{q(\gamma)}$. Finally, reaching the zeroth level, the algorithm uses $\bar{p}$ to compute $\bar{y}$ satisfying A from any value with the property G_0 . The next theorem that essentially reproves the main result of [2] uses $\text{LS}(\leq_\beta)$ -programs ($\text{PLS}(\leq_\beta)$ -programs) to witness the total search (NP-search) problems of PA.

Theorem 3. $\text{TSP}(\text{PA}) = \bigcup_{\beta \in \mathcal{O}} \text{LS}(\preceq_\beta)$ and $\text{TFNP}(\text{PA}) = \bigcup_{\beta \in \mathcal{O}} \text{PLS}(\preceq_\beta)$.

Proof. We only prove the first equality. The second is just a consequence. For the first direction, assume that $A(\bar{x}, \bar{y})$ has a $\text{LS}(\preceq_\beta)$ -program. Set $H(\gamma, \bar{x}) = \forall \bar{z} \neg G(\gamma, \bar{x}, \bar{z}) \wedge \forall \bar{y} \neg A(\bar{x}, \bar{y})$ and note that $H \in \mathbb{V}_1$. We claim that $(H(\gamma, \bar{x}), \beta)$ is a flow from $\forall \bar{y} \neg A(\bar{x}, \bar{y})$ to $\perp$. First, as $\text{PV} \vdash G(0, \bar{x}, \bar{z}) \rightarrow A(\bar{x}, \bar{p}(\bar{x}, \bar{z}))$, we have $\text{PV} \vdash \forall \bar{y} \neg A(\bar{x}, \bar{y}) \rightarrow \forall \bar{z} \neg G(0, \bar{x}, \bar{z})$ and hence $\text{PV} \vdash \forall \bar{y} \neg A(\bar{x}, \bar{y}) \leftrightarrow H(0, \bar{x})$. Secondly, as $\text{PV} \vdash G(\beta, \bar{x}, \bar{i}(\bar{x}))$, we reach $\text{PV} \vdash \forall \bar{z} \neg G(\beta, \bar{x}, \bar{z}) \leftrightarrow \perp$ and hence $\text{PV} \vdash \perp \leftrightarrow H(\beta, \bar{x})$. Finally, using $\text{PV} \vdash \gamma \neq 0 \rightarrow q(\gamma, \bar{x}, \bar{z}) < \gamma$, and $\text{PV} \vdash \gamma \neq 0 \rightarrow [G(\gamma, \bar{x}, \bar{z}) \rightarrow G(q(\gamma, \bar{x}, \bar{z}), \bar{x}, \bar{N}(\gamma, \bar{x}, \bar{z}))]$, it is easy to see that $\text{PV} \vdash \forall 1 \leq \delta \leq \beta [\neg G(q(\delta, \bar{x}, \bar{z}), \bar{x}, \bar{N}(\delta, \bar{x}, \bar{z})) \rightarrow \neg G(\delta, \bar{x}, \bar{z})]$ and hence we reach $\text{PV} \vdash \forall 1 \leq \delta \leq \beta [\forall \gamma < \delta \forall \bar{z} \neg G(\gamma, \bar{x}, \bar{z}) \rightarrow \forall \bar{z} \neg G(\delta, \bar{x}, \bar{z})]$. The latter implies $\text{PV} \vdash \forall 1 \leq \delta \leq \beta [\forall \gamma < \delta H(\gamma, \bar{x}) \rightarrow H(\delta, \bar{x})]$. Therefore, $(H(\gamma, \bar{x}), \beta)$ is a flow from $\forall \bar{y} \neg A(\bar{x}, \bar{y})$ to $\perp$. Hence, $\text{PA} \vdash \forall \bar{y} \neg A(\bar{x}, \bar{y}) \rightarrow \perp$, by Corollary 2 and thus $\text{PA} \vdash \forall \bar{x} \exists \bar{y} A(\bar{x}, \bar{y})$. For the converse, assume that $\text{PA} \vdash \forall \bar{x} \exists \bar{y} A(\bar{x}, \bar{y})$, where $A(\bar{x}, \bar{y}) \in \mathcal{L}_{\text{PV}}$ is quantifier-free. As $\text{PA} \vdash \forall \bar{y} \neg A(\bar{x}, \bar{y}) \rightarrow \perp$, by Corollary 2, $\forall \bar{y} \neg A(\bar{x}, \bar{y}) \triangleright \perp$. Hence, there exist $H(\gamma, \bar{x}) \in \mathbb{V}_1$ and $\beta \in \mathcal{O}$ such that $\text{PV} \vdash \forall \bar{y} \neg A(\bar{x}, \bar{y}) \leftrightarrow H(0, \bar{x})$, $\text{PV} \vdash H(\beta, \bar{x}) \leftrightarrow \perp$ and $\text{PV} \vdash \forall 1 \leq \delta \leq \beta [\forall \gamma < \delta H(\gamma, \bar{x}) \rightarrow H(\delta, \bar{x})]$. As $H \in \mathbb{V}_1$, there exists a quantifier-free formula $I(\gamma, \bar{x}, \bar{z})$ such that $H(\gamma, \bar{x})$ and $\forall \bar{z} I(\gamma, \bar{x}, \bar{z})$ are equivalent over PV. On the other hand, as the implications are provable in PV, we can witness the existential quantifiers by ptime functions. Hence, there are $\mathcal{L}_{\text{PV}}$ -terms $\bar{Y}(\bar{x}, \bar{z})$, $\bar{Z}(\gamma, \bar{x}, \bar{z})$, $\Delta(\gamma, \bar{x}, \bar{z})$ and $\bar{W}(\bar{x})$ such that $\text{PV} \vdash \neg A(\bar{x}, \bar{Y}(\bar{x}, \bar{z})) \rightarrow I(0, \bar{x}, \bar{z})$, $\text{PV} \vdash I(\beta, \bar{x}, \bar{W}(\bar{x})) \rightarrow \perp$ and $\text{PV} \vdash \forall 1 \leq \delta \leq \beta [(\Delta(\delta, \bar{x}, \bar{z}) < \delta \rightarrow I(\Delta(\delta, \bar{x}, \bar{z}), \bar{x}, \bar{Z}(\delta, \bar{x}, \bar{z}))) \rightarrow I(\delta, \bar{x}, \bar{z})]$. Define $G(\delta, \bar{x}, \bar{z}) = \neg I(\delta, \bar{x}, \bar{z}) \wedge (\delta \leq \beta)$,

$$q(\delta, \bar{x}, \bar{z}) = \begin{cases} \Delta(\delta, \bar{x}, \bar{z}) & \neg I(\delta, \bar{x}, \bar{z}) \wedge (\delta \leq \beta) \\ 0 & \text{otherwise} \end{cases}$$

$\bar{i}(\bar{x}) = \bar{W}(\bar{x})$ and $\bar{p}(\bar{x}, \bar{z}) = \bar{Y}(\bar{x}, \bar{z})$. It is easy to see that this new data is a $\text{LS}(\preceq_\beta)$ -program for $A(\bar{x}, \bar{y})$.

Acknowledgements. We are grateful to Pavel Pudlák and Neil Thapen to put the connection between the total search problems and the theories of arithmetic into our attention. We are also thankful for the fruitful discussions we had. The support by the FWF project P 33548 is also gratefully acknowledged.

References

1. Avigad, J., Feferman, S.: Gödel's functional ("Dialectica") interpretation. In: Handbook of proof theory, Studies in Logic and the Foundations of Mathematics, vol. 137, pp. 337–405. North-Holland, Amsterdam (1998)
2. Beckmann, A.: A characterisation of definable NP search problems in peano arithmetic. In: Ono, H., Kanazawa, M., de Queiroz, R. (eds.) WoLLIC 2009. LNCS (LNAI), vol. 5514, pp. 1–12. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02261-6_1

3. Beckmann, A., Buss, S.: The NP search problems of Frege and extended Frege proofs. *ACM Trans. Comput. Log.* **18**(2), 19 (2017), 11
4. Beckmann, A., Buss, S.R.: Polynomial local search in the polynomial hierarchy and witnessing in fragments of bounded arithmetic. *J. Math. Log.* **9**(1), 103–138 (2009)
5. Beckmann, A., Buss, S.R.: Characterising definable search problems in bounded arithmetic via proof notations. In: *Ways of Proof Theory*, Ontos Mathematical Logic, vol. 2, pp. 65–133. Ontos Verlag, Heusenstamm (2010)
6. Beckmann, A., Buss, S.R.: Improved witnessing and local improvement principles for second-order bounded arithmetic. *ACM Trans. Comput. Log.* **15**(1), 35 (2014), 2
7. Beckmann, A., Pollett, C., Buss, S.R.: Ordinal notations and well-orderings in bounded arithmetic. *Ann. Pure Appl. Logic* **120**(1–3), 197–223 (2003)
8. Buchholz, W.: Notation systems for infinitary derivations. *Arch. Math. Logic* **30**(5–6), 277–296 (1991)
9. Buchholz, W.: Explaining Gentzen’s consistency proof within infinitary proof theory. In: Gottlob, G., Leitsch, A., Mundici, D. (eds.) *KGC 1997*. LNCS, vol. 1289, pp. 4–17. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-63385-5_29
10. Buss, S.R.: Bounded arithmetic, *Studies in Proof Theory*. Lecture Notes, vol. 3. Bibliopolis, Naples (1986)
11. Buss, S.R.: The witness function method and provably recursive functions of Peano arithmetic. In: *Logic, Methodology and Philosophy of Science, IX* (Uppsala, 1991), *Studies in Logic and the Foundations of Mathematics*, vol. 134, pp. 29–68. North-Holland, Amsterdam (1994)
12. Buss, S.R.: Bounded arithmetic and propositional proof complexity. In: *Logic of computation* (Marktoberdorf, 1995), *NATO Advance Science Institutes Series F: Computer Systems Science*, vol. 157, pp. 67–121. Springer, Berlin (1997). https://doi.org/10.1007/978-3-642-59048-1_3
13. Buss, S.R.: First-order proof theory of arithmetic. In: *Handbook of proof theory*, *Studies in Logic and the Foundations of Mathematics*, vol. 137, pp. 79–147. North-Holland, Amsterdam (1998)
14. Buss, S.R.: An introduction to proof theory. In: *Handbook of proof theory*, *Studies in Logic and the Foundations of Mathematics*, vol. 137, pp. 1–78. North-Holland, Amsterdam (1998)
15. Buss, S.R., Krajíček, J.: An application of Boolean complexity to separation problems in bounded arithmetic. *Proc. London Math. Soc.* **69**(1), 1–21 (1994), (3)
16. Buss, S.R., Krajíček, J., Takeuti, G.: Provably total functions in bounded arithmetic theories R_3^i , U_2^i and V_2^i . In: *Arithmetic, proof theory, and computational complexity* (Prague, 1991), *Oxford Logic Guides*, vol. 23, pp. 116–161. Oxford University Press, New York (1993)
17. Cobham, A.: The intrinsic computational difficulty of functions. In: *Logic, Methodology and Philosophy Science* (Proceeding 1964 International Congress), pp. 24–30. North-Holland, Amsterdam (1965)
18. Cook, S.A.: Feasibly constructive proofs and the propositional calculus (preliminary version). In: *Seventh Annual ACM Symposium on Theory of Computing* (Albuquerque, N.M., 1975), pp. 83–97 (1975)
19. Fairtlough, M., Wainer, S.S.: Hierarchies of provably recursive functions. In: *Handbook of proof theory*, *Studies in Logic and the Foundations of Mathematics*, vol. 137, pp. 149–207. North-Holland, Amsterdam (1998)
20. Friedman, H., Sheard, M.: Elementary descent recursion and proof theory. *Ann. Pure Appl. Logic* **71**(1), 1–45 (1995)

21. Kołodziejczyk, L.A., Nguyen, P.: The provably total NP search problems of weak second order bounded arithmetic. *Ann. Pure Appl. Logic* **162**(6), 419–446 (2011)
22. Krajíček, J., Pudlák, P., Takeuti, G.: Bounded arithmetic and the polynomial hierarchy. In: *International Symposium on Mathematical Logic and its Applications* (Nagoya, 1988), vol. 52, pp. 143–153 (1991)
23. Krajíček, J., Skelley, A., Thapen, N.: NP search problems in low fragments of bounded arithmetic. *J. Symbolic Logic* **72**(2), 649–672 (2007)
24. Kreisel, G.: On the interpretation of non-finitist proofs. II. Interpretation of number theory. Applications. *J. Symbolic Logic* , **17**, 43–58 (1952)
25. Paris, J.B., Wilkie, A.J.: 0 sets and induction. *Open days in model theory and set theory*, pp. 237–248 (1981)
26. Rathjen, M.: The realm of ordinal analysis. In: *Sets and proofs* (Leeds, 1997). London Mathematical Society Lecture Note Series, vol. 258, pp. 219–279. Cambridge University, Press, Cambridge (1999)
27. Skelley, A., Thapen, N.: The provably total search problems of bounded arithmetic. *Proc. Lond. Math. Soc.* **103**(1), 106–138 (2011), (3)
28. Takeuti, G., Zaring, W.M.: *Introduction to axiomatic set theory*, vol. 1. Springer Science & Business Media (2012). <https://doi.org/10.1007/978-1-4613-8168-6>
29. Troelstra, A.S.: Realizability. In: *Handbook of proof theory*, *Studies in Logic and the Foundations of Mathematics*, vol. 137, pp. 407–473. North-Holland, Amsterdam (1998)



Non-monotonic Reasoning via Dynamic Consequence

Carlos Areces^{1,2}, Valentin Cassano^{1,2,3,4(✉)}, and Raul Fervari^{1,2,4}

¹ Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET),
Buenos Aires, Argentina

{carlos.areces,rfervari}@unc.edu.ar, valentin@dc.exa.unrc.edu.ar

² Universidad Nacional de Córdoba (UNC), Córdoba, Argentina

³ Universidad Nacional de Río Cuarto (UNRC), Río Cuarto, Argentina

⁴ Guangdong Technion - Israel Institute of Technology (GTIIT), Shantou, China

Abstract. We approach the concept of Pivotal Rule Consequence (PRC) proposed in [14,15] from a semantical perspective, resorting to model updates in Public Announcement Logic (PAL) [17]. In doing this, we take inspiration from the notion of dynamic consequence from [3,6]. Our perspective gains in interest since PRC serves as a “bridge” from Classical Logic to Default Logic –one of the most well-known non-monotonic formalisms. We show how the internalization of PRC in PAL leads to clear semantics of the former, and to completeness and transfer results. Moreover, we address the case of credulous consequence in Default Logic as a particular case of PRC. Interestingly, we cast credulous consequence as a model checking problem. We argue that our results open the way to use well-known semantic tools from modal logic to study properties of different non-monotonic logics.

1 Introduction

In spite of its logical terminology, the field of non-monotonic logic has a reputation for presenting itself in an unfamiliar way to traditional logicians. In [14,15], it is shown that this need not be the case. The argument put forth is that, provided some preconceptions and misconceptions are set aside, there are logics acting as natural “bridges” between classical consequence and some well-known non-monotonic logics. These bridges possess some of the characteristics of their classical starting points and, at the same time, they start exhibiting some of the distinctive features of their non-monotonic endpoints. Crucially, monotonicity is not given up initially. Instead, it originates as a result of introducing “consistency checks”. Using these bridges, it is possible to transition in a clear and step-wise manner from classical to non-monotonic logics.

In this paper, we focus our attention on so-called *pivotal-rule consequence* (PRC). This bridge extends the classical notion of consequence by using non-admissible rules of inference called *pivotal rules* (i.e., by using rules whose incorporation modifies the set of consequences of a logic). By imposing “consistency checks” on how pivotal rules are used, PRC takes us from classical consequence

to consequence in Default Logic [18] –one of the most prominent approaches to non-monotonic reasoning [1].

The results in [14, 15] are obtained by manipulating PRC via closure operators on sets of formulas. In this work, we revise PRC from a semantical perspective resorting to the formal machinery of Public Announcement Logic (PAL) [17]. In this revision, we take inspiration from van Benthem’s program on “logical dynamics” [3] (see also [6, 19]). On the one hand, PAL is a modal logic with an *announcement* modality $[\varphi]$ that restricts the current (relational) model to those states where φ holds. On the other hand, the program on logical dynamics shifts the static role played by premisses in a consequence relation and views them as new information brought into consideration via some action. This results in the consequence relation taking a dynamic stance. In very simple terms, the program on logical dynamics takes the premisses of a consequence relation as a sequence of announcements; i.e., $\psi_1, \dots, \psi_n \models \varphi$ shifts to $\models [\psi_1] \dots [\psi_n] \varphi$. We adapt these ideas to capture the effect brought in by the use of pivotal rules in PRC. In doing so, we show that pivotal-rule consequence can be seen as a dynamic modal logic.

In addressing PRC from a semantic perspective we complete the picture of [14, 15]. Our results establish a novel connection between PRC and Modal Logic, via PAL. In addition to offering new insights into the nature of the framework proposed in [14, 15], our approach allows for transfer results: theoretical and practical tools developed for PAL turn out relevant for PRC (and for Default Logic as a particular case). These results include: bisimulations, axiomatizations, tableaux calculi, model checking algorithms, etc.

Contributions. We show how PRC, and credulous consequence in Default Logic as a particular case, can be seen as a modal logic using dynamic update operators. In doing this, the formal machinery of PAL turns out to be sufficiently expressive. Interestingly, we obtain that credulous consequence reduces to *model checking* in PAL. Our results establish that PAL can be seen as providing a model theoretic semantics for PRC and to credulous consequence as a special case. This approach differs fundamentally from other semantic takes on Default Logic, e.g., those in [4, 7, 9]. We provide a mapping from credulous consequence to consequence in a modal logic with updates, i.e., PAL. In particular, we capture credulous consequence as particular tautologies of PAL—something which, as far as we know, has not been addressed. Moreover, we detail how our approach leads to natural axiomatizations, and to expressivity and complexity results. Concretely, our contributions consist of:

1. A novel characterization of PRC in terms of Hilbert-style deductions (something which we use in the rest of the paper).
2. A characterization of PRC as particular tautologies of an alternative presentation of PAL, which we refer to as Pivotal Modal Logic (PML).
3. A study of the logical properties of PRC via transfer results from PML. In particular, axiomatization, expressivity and complexity results.
4. A characterization of credulous consequence in PML. Again, we use our framework to investigate an axiomatization, expressivity and complexity results.

Interestingly, deciding credulous consequence boils down to a model checking problem over a particular model. This enables us to obtain a reasonable small complexity upper bound in a simple way.

Structure. Sect. 2 covers basic material on Classical Propositional Logic. Sect. 3 presents PRC. As a first novel result, we offer a way of looking at PRC via Hilbert-style deductions. Sect. 4 shows how PAL can be used to capture PRC from a semantic perspective (see Theorem 2). We use our framework to study: axiomatization, expressivity and complexity results. Sect. 4.2 discusses the case of credulous consequence from a semantic perspective (see Theorem 5) and study its properties. Sect. 5 offers some final remarks and discusses future lines of research.

2 Preliminaries

We briefly review the basics of classical propositional logic (CPL; see, e.g., [16]).

Definition 1. Let $\text{Prop} = \{p_i \mid i \in \mathbb{N}\}$ be a set of proposition symbols; the set Form_0 formulas of CPL built on Prop is given by the grammar

$$\varphi, \psi ::= p_i \mid \perp \mid \varphi \rightarrow \psi.$$

We use $\top$, $\neg\varphi$, $\varphi \vee \psi$, $\varphi \wedge \psi$, and $\varphi \leftrightarrow \psi$ as abbreviations – defined as usual.

The semantics of CPL is given resorting to assignments and valuations.

Definition 2. An assignment is a function $v : \text{Prop} \rightarrow \{0, 1\}$. A valuation is an extension of an assignment v to a function $v^* : \text{Form}_0 \rightarrow \{0, 1\}$ s.t.: $v^*(\perp) = 0$, and $v^*(\varphi \rightarrow \psi) = 1$ iff either $v^*(\varphi) = 0$, or $v^*(\psi) = 1$. A valuation v^* is a model of φ (or φ is satisfiable in v^*), written $v^* \models \varphi$, iff $v^*(\varphi) = 1$. A formula φ has a model, or is satisfiable, iff exists a valuation v^* s.t. $v^* \models \varphi$. These definitions extend to sets of formulas in the obvious way.

Definition 2 yields a relation of *semantic consequence* between sets of formulas, called premisses, and formulas, called consequences. This relation is given below.

Definition 3. A formula φ is a (semantic) consequence of a set of formulas Φ , written $\Phi \models \varphi$, iff for all valuations v^* , $v^* \models \Phi$ implies $v^* \models \varphi$. We call a formula φ a tautology, and write $\models \varphi$, iff $\emptyset \models \varphi$.

The syntactic counterpart of the relation $\models$ is defined via deductions.

Definition 4. A deduction of a formula φ from a set of formulas Φ is a finite sequence $\bar{\psi} = \psi_1 \dots \psi_n$ of formulas s.t. $\psi_n = \varphi$, and for all $k < n$, either:

1. ψ_k is an axiom of CPL (see [16]);
2. ψ_k is a premiss (i.e., $\psi_k \in \Phi$);
3. ψ_k is obtained using mp (i.e., exists $i, j < k$ s.t. $\psi_j = \psi_i \rightarrow \psi_k$).

We write $\Phi \vdash \varphi$ iff exists a deduction of φ from Φ .

Theorem 1 (Soundness-Completeness). $\Phi \vdash \varphi$ iff $\Phi \models \varphi$.

3 From Classical to Default Consequence

Deductions in CPL give rise to an operation $\text{Cn} : 2^{\text{Form}_0} \rightarrow 2^{\text{Form}_0}$ defined as $\text{Cn}(\Phi) = \{\varphi \mid \Phi \vdash \varphi\}$. This operation satisfies some well-known properties. In particular, it satisfies the *principle of monotonicity*, i.e.:

$$\Phi \subseteq \Psi \text{ implies } \text{Cn}(\Phi) \subseteq \text{Cn}(\Psi).$$

Following [14, 15], we show how to construct, in a natural way, an operation D which extends Cn but which fails to satisfy monotonicity. We introduce D by taking a detour via an operation C which retains monotonicity, but starts exhibiting some of the distinctive features of D . When seen in this light, C acts as a bridge between Cn and D , and it streamlines the presentation of the latter.

We begin the presentation of the operation C by introducing *pivotal rules*.

Definition 5. A pivotal rule is a pair (π, χ) of formulas of CPL. We use (π/χ) as notation for a pivotal rule; we call π its prerequisite and χ its consequent. The image of a set P of pivotal rules under a set Φ formulas of CPL is defined as $P(\Phi) = \{\chi \mid (\pi/\chi) \in P \text{ and } \pi \in \Phi\}$. The set Φ is closed under P if $P(\Phi) \subseteq \Phi$.

Henceforth, we will deal only with finite sets P of pivotal rules. Pivotal rules are best understood as non-admissible rules of inference, i.e., rules whose incorporation extends the set of CPL-consequences of a set of formulas. Definition 6 makes this idea clear.

Definition 6. Let P be a set of pivotal rules; define $C_P : 2^{\text{Form}_0} \rightarrow 2^{\text{Form}_0}$ as:

$$C_P(\Phi) = \bigcap \{\Psi \mid \Phi \subseteq \Psi, \text{Cn}(\Psi) \subseteq \Psi, \text{ and } P(\Psi) \subseteq \Psi\}.$$

We say that φ is a pivotal-rule consequence of Φ iff $\varphi \in C_P(\Phi)$.

The operation C_P in Definition 6 is called *pivotal-rule consequence* in [15]. From now on, we refer to it as *pivotal consequence*. In words, $C_P(\Phi)$ is the smallest superset of Φ closed under Cn and P . Note that Definition 6 does not introduce a single operation C but a family of such operations; one for each set P of pivotal rules. We drop the subscript P if it can be understood from context or not needed.

From its definition, $\text{Cn}(\Phi) \subseteq C(\Phi)$. For this reason, we say that C extends Cn . Just as Cn , C is monotonic: $\Phi \subseteq \Psi$ implies $C(\Phi) \subseteq C(\Psi)$. Moreover, C is closed: $C(C(\Phi)) = C(\Phi)$. Nevertheless, C diverges from Cn in some important ways. For instance, C may fail to satisfy the property of *disjunction*. This property states that if $\gamma \in C(\{\varphi\})$ and $\gamma \in C(\{\psi\})$, then, $\gamma \in C(\{\varphi \vee \psi\})$. To see its failure, let $P = \{(p/q), (r/q)\}$; then $q \in C_P(\{p\})$ and $q \in C_P(\{r\})$, but $q \notin C_P(\{p \vee r\})$.

Definition 7 offers another way of looking at C . This definition also serves as an intermediate step towards non-monotonicity.

Definition 7. Let $\bar{P}$ be a total ordering of a set P of pivotal rules; we use (π_i/χ_i) to indicate the i -th element of $\bar{P}$. Define an operation $C_{\bar{P}} : 2^{\text{Form}_0} \rightarrow 2^{\text{Form}_0}$ s.t. for all Φ , $C_{\bar{P}}(\Phi) = \bigcup \{C_{\bar{P}}^i(\Phi) \mid i \geq 0\}$ where:

$$C_{\bar{P}}^0(\Phi) = \text{Cn}(\Phi) \quad \text{and} \quad C_{\bar{P}}^{(i+1)}(\Phi) = \begin{cases} \text{Cn}(C_{\bar{P}}^i(\Phi) \cup \{\chi_i\}) & \text{if } \pi_i \in C_{\bar{P}}^i(\Phi) \\ C_{\bar{P}}^i(\Phi) & \text{otherwise.} \end{cases}$$

As before, we write $\bar{C}$ when there is no need to make the set P of pivotal rules explicit. The operation $\bar{C}$ does not immediately characterize the operation C . For instance, in general, $\bar{C}$ is not closed, i.e., $\bar{C}(\Phi) \neq \bar{C}(\bar{C}(\Phi))$. To see why, let $P = \{(q/r), (p/q)\}$; consider the total ordering $\bar{P}$ in which (q/r) is before (p/q) ; then, $C_{\bar{P}}(\{p\}) = \text{Cn}(\{p, q\})$; however, $C_{\bar{P}}(C_{\bar{P}}(\{p\})) = \text{Cn}(\{p, q, r\})$. Nonetheless, it is possible to establish the following result.

Proposition 1. $\varphi \in C_P(\Phi)$ iff exists $\bar{P}$ s.t. $\varphi \in C_{\bar{P}}(\Phi)$.

Proposition 1 tells us that pivotal consequence amounts to checking what is the case in *some* total ordering of pivotal rules.

The argument in [15] is that C helps us to pave the way to non-monotonicity. More precisely, we obtain a non-monotonic consequence operation by imposing a consistency constraint on the use of the pivotal rules. We make this clear below.

Definition 8. Let $\bar{P}$ be a total ordering on a set P of pivotal rules; define an operation $D_{\bar{P}} : 2^{\text{Form}_0} \rightarrow 2^{\text{Form}_0}$ s.t. for all Φ , $D_{\bar{P}}(\Phi) = \bigcup \{D_{\bar{P}}^i(\Phi) \mid i \geq 0\}$ where:

$$D_{\bar{P}}^0(\Phi) = \text{Cn}(\Phi) \quad \text{and} \quad D_{\bar{P}}^{(i+1)}(\Phi) = \begin{cases} \text{Cn}(D_{\bar{P}}^i(\Phi) \cup \{\chi_i\}) & \text{if } \pi_i \in D_{\bar{P}}^i(\Phi) \\ & \text{and } \neg\chi_i \notin D_{\bar{P}}^i(\Phi) \\ D_{\bar{P}}^i(\Phi) & \text{otherwise.} \end{cases}$$

Again, we use $\bar{D}$ when there is no need to make P explicit. The operation $\bar{D}$, and the operations Cn and $\bar{C}$ are related in the following proposition.

Proposition 2. $\text{Cn}(\Phi) \subseteq \bar{D}(\Phi) \subseteq \bar{C}(\Phi)$.

$\bar{D}$ retains some properties of $\bar{C}$. E.g., it extends Cn ; in general it is not closed; and it may fail to satisfy the property of disjunction. This said, orderings which are indistinguishable in the case of $\bar{C}$ are distinguishable in the case of $\bar{D}$. To illustrate this point, let $P = \{(p/q), (p/\neg q)\}$; consider orderings $\bar{P}_1 = \langle (p/q), (p/\neg q) \rangle$ and $\bar{P}_2 = \langle (p/\neg q), (p/q) \rangle$; then $D_{\bar{P}_1}(\{p\}) = \text{Cn}(\{p, q\})$ and $D_{\bar{P}_2}(\{p\}) = \text{Cn}(\{p, \neg q\})$. Yet, $C_{\bar{P}_1}(\{p\}) = C_{\bar{P}_2}(\{p\}) = \text{Form}_0$. More interestingly, $\bar{D}$ fails to satisfy monotonicity, i.e., it is *non-monotonic*. For example, let $P = \{(p/q)\}$; then $D_{\bar{P}}(\{p\}) = \text{Cn}(\{p, q\})$; however, $D_{\bar{P}}(\{p, \neg q\}) = \text{Cn}(\{p, \neg q\})$.

The operation $\bar{D}$ is in itself of interest. However, in some cases, we may want to depart from particular orderings. We make this idea precise in Definition 9.

Definition 9. Define D_P s.t. $\varphi \in D_P(\Phi)$ iff exists $\bar{P}$ s.t. $\varphi \in D_{\bar{P}}(\Phi)$.

We often write D instead of D_P . The following proposition is immediate.

Proposition 3. It follows that: $\bar{D}(\Phi) \subseteq D(\Phi) \subseteq C(\Phi)$.

The operation D is interesting because it coincides with *credulous* consequence for the case of normal defaults in default logic (see [14, 18]). Namely, φ is a *credulous default consequence* of Φ iff $\varphi \in D(\Phi)$. It is important to notice that this operation is not closed, i.e., in general, $D(D(\Phi)) \neq D(\Phi)$. To see why,

let $P = \{(\top/p), (\top/\neg p)\}$; it follows that $D_P(\emptyset) = \text{Cn}(\{p\}) \cup \text{Cn}(\{\neg p\})$; whereas $D_P(D_P(\emptyset)) = \text{Form}_0$. This example also shows that $D(\emptyset) \neq \text{Cn}(D(\emptyset))$. In other words, D may not be closed under Cn .

Pivotal Deductions. We conclude this section with a novel characterization of the operations C and D via what we call pivotal deductions. This is our first contribution and it provides us the right setting to investigate pivotal consequence from a dynamic logic perspective. We will retake these ideas in Sect. 4.

Definition 10. Let P be a set of pivotal rules; a P -deduction of a formula φ from a set of formulas Φ is a finite sequence $\bar{\psi} = \psi_1 \dots \psi_n$ of formulas such that $\psi_n = \varphi$, and for all $k < n$ one of the following holds:

1. ψ_k is an axiom of CPL, a premiss, or obtained using mp (see Definition 4);
or
2. exists $j < k$ s.t. $(\psi_j/\psi_k) \in P$ –called P -detachment.

We say that a pivotal rule $\rho \in P$ has been used in $\bar{\psi}$ iff $\rho = (\psi_j/\psi_k)$ for some $1 \leq j < k \leq n$. Let Ψ be the set of all consequents of the pivotal rules used in $\bar{\psi}$, we call $\bar{\psi}$ credulous iff $\text{Cn}(\Phi \cup \Psi) = \text{Form}_0$ iff $\text{Cn}(\Phi) = \text{Form}_0$. We write:

1. $\Phi \vdash_P \varphi$ iff exists a P -deduction of φ from Φ ;
2. $\Phi \vdash_P \varphi$ iff exists a credulous P -deduction of φ from Φ .

We often refer to P -deductions as *pivotal deductions* and to credulous P -deductions as *credulous deductions*. Intuitively, a pivotal deduction extends the notion of a deduction in CPL accommodating for the use of pivotal rules via P -detachment. The following propositions are immediate from the definitions.

Proposition 4. $\Phi \vdash_P \varphi$ iff exists a seq. $\bar{\rho} = \rho_1 \dots \rho_n$ of pivotal rules in P s.t.:

$$\Phi \cup X \vdash \varphi \text{ and for all } 1 \leq i \leq n, \Phi \cup X_{(i-1)} \vdash \pi_i, \quad (1)$$

where: $\pi_i = (\pi_i/\chi_i)$; $X_j = \{\chi_1, \dots, \chi_j\}$; and $X = X_n$. Moreover, $\Phi \vdash_P \varphi$ iff in addition to Eq. (1) it holds that

$$\text{for all } 1 \leq i \leq n, \Phi \cup X_{(i-1)} \not\vdash \neg \chi_i. \quad (2)$$

Proposition 5. $\varphi \in C_P(\Phi)$ iff $\Phi \vdash_P \varphi$; and $\varphi \in D_P(\Phi)$ iff $\Phi \vdash_P \varphi$;

From Propositions 4 and 5, it follows that pivotal deductions provide witnesses for $\varphi \in C(\Phi)$; whereas credulous deductions provide witnesses for $\varphi \in D(\Phi)$.

Remark 1. Definition 10 and Proposition 5 bring about a discussion on *compactness* for the operations C and D . It can easily be established that $\varphi \in C(\Phi)$ iff for some finite $\Phi' \subseteq \Phi$, $\varphi \in C(\Phi')$. The same claim is not true for D . Although $\varphi \in D(\Phi)$ implies that there is a finite $\Phi' \subseteq \Phi$ s.t. $\varphi \in D(\Phi')$, in general, the converse of this statement does not hold. This is unsurprising for, as commented on in [14], full compactness for D brings back monotonicity.

4 From Pivotal Rules to Model Updates

In this section we look at pivotal consequence through the lens of modal logic [5]. The modal logic we use is an alternative presentation of the single-agent Public Announcement Logic (PAL) [17]. PAL has as its distinguishing characteristic a modality which can “update” the model while a formula is evaluated. The inclusion of this update modality has three main benefits in our semantic exploration of pivotal consequence. First, it enables us to internalize the effect of pivotal rules in the object language, inspired by the notion of dynamic consequence [3, 6]. Second, it enables us to provide a semantics for pivotal rules. Third, it opens the door for a study of operations C and D in Sect. 3 from the perspective of modal model theory. Our framework builds a novel bridge between pivotal consequence and modal logic, and particularly, from default consequence to modal logic.

4.1 Pivotal Consequence in PAL

We begin with a brief introduction to the formal machinery behind PAL [8]. To avoid confusions with its usual presentation, we refer to our presentation of PAL as *Pivotal Modal Logic* (PML). In what follows, we use $\models_{\text{CPL}}$ to indicate the relation of semantic consequence of CPL (see Definition 3).

Definition 11. *The set Form of PML formulas is built over the set Prop of proposition symbols and is given by the grammar:*

$$\varphi, \psi ::= p_i \mid \perp \mid \varphi \rightarrow \psi \mid A\varphi \mid [\varphi]\psi.$$

Other Boolean operators are defined as usual. Moreover, we use $E\varphi$ and $\langle\varphi\rangle\psi$ to abbreviate $\neg A\neg\varphi$ and $\neg[\varphi]\neg\psi$, resp. We refer to the elements of the language of PML as pivotal (modal) formulas.

Pivotal formulas are interpreted over pivotal models.

Definition 12. *A pivotal model is a non-empty subset $\mathfrak{M} \subseteq \{0, 1\}^{\text{Prop}}$, i.e., it is a non-empty set of functions $w : \text{Prop} \rightarrow \{0, 1\}$. The pair $\mathfrak{M}, w$ is a pointed pivotal model iff $w \in \mathfrak{M}$. We say that $\mathfrak{M}, w$ is finite iff $\mathfrak{M}$ is finite.*

Intuitively, we look at pivotal models as sets of CPL-assignments (cf. Definition 2). The definition of satisfiability in a pivotal model is given immediately below.

Definition 13. *Let $\mathfrak{M}, w$ be a pointed pivotal model; for any pivotal formula φ , we define φ is satisfiable in $\mathfrak{M}$ at w , written $\mathfrak{M}, w \models \varphi$, inductively as:*

$$\begin{array}{ll} \mathfrak{M}, w \models \perp & \text{never} \\ \mathfrak{M}, w \models p & \text{iff } w(p) = 1 \\ \mathfrak{M}, w \models \varphi \rightarrow \psi & \text{iff } \mathfrak{M}, w \models \varphi \text{ implies } \mathfrak{M}, w \models \psi \\ \mathfrak{M}, w \models A\varphi & \text{iff for all } w' \in \mathfrak{M}, \mathfrak{M}, w' \models \varphi \\ \mathfrak{M}, w \models [\varphi]\psi & \text{iff } \mathfrak{M}, w \models \varphi \text{ implies } \mathfrak{M}|_{\varphi}, w \models \psi, \end{array}$$

where: $\mathfrak{M}|_\varphi = \{w \in \mathfrak{M} \mid \mathfrak{M}, w \Vdash \varphi\}$ ¹. We write $\mathfrak{M} \Vdash \varphi$ iff for all w in $\mathfrak{M}$, $\mathfrak{M}, w \Vdash \varphi$. Let Φ be a set of pivotal formulas; we define $\mathfrak{M}, w \Vdash \Phi$ and $\mathfrak{M} \Vdash \Phi$ as usual. We write $\Phi \models \varphi$ iff for all $\mathfrak{M}, w$, if $\mathfrak{M}, w \Vdash \Phi$, then, $\mathfrak{M}, w \Vdash \varphi$. We write $\models \varphi$ instead of $\emptyset \models \varphi$.

Definition 13 states that A and E are the universal and existential modalities [11]; and that $[\_]$ and $\langle \_ \rangle$ are the update modalities of PAL [17].

Let us turn our attention onto how to use PML to capture pivotal consequence from a semantic perspective. The following propositions are preliminary and are meant to shed light into the relation between pivotal models and CPL valuations.

Proposition 6. *Let φ and ψ be CPL-formulas; then: 1. $\{\varphi\} \models_{\text{CPL}} \psi$ iff $\{\varphi\} \models \psi$; and 2. $\{\varphi\} \models_{\text{CPL}} \psi$ iff $\models [\varphi]\psi$. Moreover, for any PML-formula χ , we get that: 3. $\models [\varphi][\psi]\chi$ iff $\models [\varphi \wedge \psi]\chi$.*

Proof. Item 1 is immediate. For Item 2, we prove $\{\varphi\} \not\models_{\text{CPL}} \psi$ iff $\not\models [\varphi]\psi$. Left to right: Suppose that $\{\varphi\} \not\models_{\text{CPL}} \psi$. Then, there is an assignment $v : \text{Prop} \rightarrow \{0, 1\}$ s.t. $v^*(\varphi) = 1$ and $v^*(\psi) = 0$. Let $\mathfrak{M}$ be the pivotal model which consists solely of v , i.e., $\mathfrak{M} = \{v\}$. Since $\mathfrak{M}|_\varphi = \mathfrak{M} = \{v\}$, it follows that $\mathfrak{M}, v \Vdash \varphi$ and $\mathfrak{M}|_\varphi, v \not\Vdash \psi$. Thus, $\not\models [\varphi]\psi$. Right to left: Suppose that $\not\models [\varphi]\psi$. There is $\mathfrak{M}, w$ s.t. $\mathfrak{M}, w \not\Vdash [\varphi]\psi$. This means that $\mathfrak{M}, w \Vdash \varphi$ and $\mathfrak{M}|_\varphi, w \not\Vdash \psi$. Let w^* be the CPL-valuation based on w ; immediately, $w^*(\varphi) = 1$ and $w^*(\psi) = 0$. Therefore, $\{\varphi\} \not\models_{\text{CPL}} \psi$. Item 3 follows from the fact that: if φ and ψ are CPL-formulas, then, for any pivotal model $\mathfrak{M}$, $(\mathfrak{M}|_\varphi)|_\psi = \mathfrak{M}|_{(\varphi \wedge \psi)}$. To see why, let $w \in \mathfrak{M}|_{(\varphi \wedge \psi)}$; by definition, $\mathfrak{M}, w \Vdash \varphi \wedge \psi$. Then, $\mathfrak{M}, w \Vdash \varphi$ and $\mathfrak{M}, w \Vdash \psi$. So, $w \in \mathfrak{M}|_\varphi$ and $\mathfrak{M}|_\varphi, w \Vdash \psi$. This implies $w \in (\mathfrak{M}|_\varphi)|_\psi$; from which we get, $\mathfrak{M}|_{(\varphi \wedge \psi)} \subseteq (\mathfrak{M}|_\varphi)|_\psi$. Similarly, $(\mathfrak{M}|_\varphi)|_\psi \subseteq \mathfrak{M}|_{(\varphi \wedge \psi)}$. Therefore, $(\mathfrak{M}|_\varphi)|_\psi = \mathfrak{M}|_{(\varphi \wedge \psi)}$.

Proposition 6 portrays the relation between the update modality $[\_]$ and CPL-consequence as a simple instance of the notion of dynamic consequence from [3, 6].

We introduce some notation to help us deal with pivotal consequence in PML.

Definition 14. *Let (π/χ) be a pivotal rule, and ψ be a pivotal formula; we use $(\pi/\chi)\psi$ as an abbreviation for $\pi \wedge [\chi]\psi$.*

Intuitively, $(\pi/\chi)\psi$ captures the effect of a single P -detachment step in Definition 10 in semantic terms². Proposition 7 gives a more concrete view of how this is done.

Proposition 7. *Let (π/χ) be a pivotal rule, and φ and ψ be CPL-formulas:*

$$\models [\varphi](\pi/\chi)\psi \text{ iff } \{\varphi\} \models_{\text{CPL}} \pi \text{ and } \{\varphi, \chi\} \models_{\text{CPL}} \psi.$$

Proof. Left to right: Suppose that $\models [\varphi](\pi/\chi)\psi$. By definition, $\models [\varphi](\pi \wedge [\chi]\psi)$. It follows that, $\models [\varphi]\pi \wedge [\varphi][\chi]\psi$, and so (a) $\models [\varphi]\pi$ and (b) $\models [\varphi][\chi]\psi$. We obtain $\{\varphi\} \models_{\text{CPL}} \pi$ using (a) and Item 2 in Proposition 6. We obtain $\{\varphi, \chi\} \models_{\text{CPL}} \psi$ using (b) and Items 1 and 3 in Proposition 6.

¹ $\mathfrak{M}|_\varphi$ might be empty. If so, $\mathfrak{M}, w \Vdash \varphi$ doesn't hold and $\mathfrak{M}, w \models [\varphi]\psi$ is trivially true.

² Definition 14 allows us to refer indistinctly to a pivotal rule, and to the corresponding formula capturing it. Context will always disambiguate.

The result in Proposition 7 can be generalized to capture the effect of successive P -detachment steps at the level of pivotal models. This is done in Proposition 8.

Proposition 8. *Let $\bar{\rho} = \rho_1 \dots \rho_n$ be a sequence of pivotal rules s.t. $\rho_i = (\pi_i / \chi_i)$; and φ and ψ be CPL-formulas. It follows that $\models [\varphi] \bar{\rho} \psi$ is equivalent to*

$$\text{for all } 1 \leq i \leq n, \{\varphi, \chi_1, \dots, \chi_{(i-1)}\} \models_{\text{CPL}} \pi_i.$$

Proof. By definition, $[\varphi] \bar{\rho} \psi = [\varphi](\pi_1 \wedge ([\chi_1](\dots (\pi_n \wedge [\chi_n] \psi))))$. The cases where $\bar{\rho}$ is empty or has just one pivotal rule correspond exactly to Item 1 in Proposition 6 and to Proposition 7. The general case makes repeated use of the fact that $\models [\gamma](\xi \wedge \eta)$ iff $\models [\gamma] \xi \wedge [\gamma] \eta$ and Item 3 in Proposition 7.

It is easy to see that Proposition 8 is the semantic counterpart to Proposition 4. This proposition enables us to *internalize* the effect of P -detachment steps in the language of PML. Moreover, it leads in a natural way to the following semantic formulation of pivotal consequence as a tautology of PML.

Definition 15. *Let P be a set of pivotal rules, and φ and ψ be CPL-formulas; define $\{\varphi\} \models_P \psi$ iff exists a sequence $\bar{\rho}$ of pivotal rules in P s.t. $\models [\varphi] \bar{\rho} \psi$.*

Theorem 2 establishes the parallel between $\vdash_P$ and $\models_P$. Its proof is direct from Propositions 4 and 8, and Theorem 1.

Theorem 2. *Let P be a set of pivotal rules, and φ and ψ be CPL-formulas:*

$$\{\varphi\} \vdash_P \psi \text{ iff } \{\varphi\} \models_P \psi.$$

Theorem 2 offers an account of how to capture pivotal consequence semantically, and from a dynamic perspective, as certain tautologies of PML. It may be noted that the definitions and results presented above restrict their attention to singleton sets of premisses. There is no loss of generality in this restriction. Definition 15 can be extended to arbitrary sets of premisses by defining $\varphi = \varphi_1 \wedge \dots \wedge \varphi_n$ for some finite subset $\{\varphi_1 \dots \varphi_n\} \subseteq \Phi$. This reformulation of Definition 15 has no impact on Theorem 2 because of compactness of CPL. We have chosen to deal with singleton sets of premisses to simplify the exposition of the main concepts and results.

Transfer Results. We conclude this section by showing some transfer results from PML to the setting of pivotal consequence. In particular, we tackle *complete axiomatizations*, *expressivity*, and *complexity* of pivotal consequence. These results take advantage of the fact that PML is a modal logic.

Axiom system. Working a logic from a semantic perspective has its advantages. In our case, it enabled us to recast the notion of pivotal rule consequence over CPL formulas into a notion of tautology in PML. At the same time, one may ask whether it is possible to use a syntactic deduction system in PML to determine pivotal rule consequence.

Table 1 introduces an axiomatization of PML. This is a standard complete axiomatization for PAL. It includes the S5 axioms and rules for the A modality, and the reduction axioms for eliminating $[\_]$ (see, e.g., [8, 11] for details).

Table 1. Axiom system for PML.

Axioms:			
(Taut) CPL-tautologies	(K)	$A(\psi \rightarrow \varphi) \wedge A\psi \rightarrow A\varphi$	(T) $A\varphi \rightarrow \varphi$
	(4)	$A\varphi \rightarrow AA\varphi$	(5) $\neg A\varphi \rightarrow A\neg A\varphi$
Rules:	(mp)	from $\varphi \rightarrow \psi$ and φ infer ψ	(nec) from φ infer $A\varphi$
Reduction Axioms:	(univ)	$[\varphi]A\psi \leftrightarrow (\varphi \rightarrow A[\varphi]\psi)$	(bot) $[\varphi]\perp \leftrightarrow (\varphi \rightarrow \perp)$
	(imp)	$[\varphi](\psi \rightarrow \chi) \rightarrow ([\varphi]\psi \rightarrow [\varphi]\chi)$	(prop) $[\varphi]p_i \leftrightarrow (\varphi \rightarrow p_i)$

Theorem 3. *Table 1 is a sound and strongly complete axiom system for PML.*

The axiom system for PML is the exact machinery behind single-agent PAL. The global modality A can be seen as a total relation in a Kripke model. Theorem 3 implies that P -deductions can be captured as deductions in single-agent PAL.

Expressive Power. Turning to expressivity, it is important to determine whether two models are structurally equivalent for a particular logic; and to provide characterization results which make coincide the notion of model equivalence and of logical equivalence. In Modal Logic this is related to the notion of bisimulation [2]. Model equivalence in PML is fairly straightforward. It is easily seen that finite models form a Hennessy-Milner class for PML (see, e.g., [10] for details).

Definition 16. *Let $\mathfrak{M}_1, w_1$ and $\mathfrak{M}_2, w_2$ be two pointed pivotal models; we write $\mathfrak{M}_1, w_1 \equiv \mathfrak{M}_2, w_2$ iff $\mathfrak{M}_1, w_1 \Vdash \varphi$ iff $\mathfrak{M}_2, w_2 \Vdash \varphi$ (for all PML-formulas φ).*

Proposition 9 (Hennessy-Milner Property). *The following are equivalent:*
1. $\mathfrak{M}_1, w_1 = \mathfrak{M}_2, w_2$; and 2. $\mathfrak{M}_1, w_1 \equiv \mathfrak{M}_2, w_2$.

Proof. Left to right: trivial. Right to left: we prove $\mathfrak{M}_1, w_1 \neq \mathfrak{M}_2, w_2$ implies $\mathfrak{M}_1, w_1 \not\equiv \mathfrak{M}_2, w_2$. Suppose that $\mathfrak{M}_1, w_1 \neq \mathfrak{M}_2, w_2$; either (1) $w_1 \neq w_2$ or (2) $\mathfrak{M}_1 \neq \mathfrak{M}_2$. From (1), exists $p \in \text{Prop}$ s.t. $w_1(p) \neq w_2(p)$. But then, (w.l.o.g.) $\mathfrak{M}_1, w_1 \Vdash p$ and $\mathfrak{M}_2, w_2 \not\Vdash p$. Thus, $\mathfrak{M}_1, w_1 \not\equiv \mathfrak{M}_2, w_2$. From (2), (w.l.o.g.) there is $w^* \in \mathfrak{M}_1$ s.t. $w^* \notin \mathfrak{M}_2$. That is, for all $w_i \in \mathfrak{M}_2$, there is p_j s.t. $w^*(p_j) \neq w_i(p_j)$. Moreover, because $\mathfrak{M}_1$ and $\mathfrak{M}_2$ are finite, there is a maximal j such that p_j has this property. Let $\varphi = (l_0 \wedge \dots \wedge l_j)$ where $l_i = p_i$ if $w^*(p_i) = 1$ and $l_i = \neg p_i$ otherwise. Clearly, $\mathfrak{M}_1, w_1 \Vdash E\varphi$ (because of w^*), but $\mathfrak{M}_2, w_2 \not\Vdash E\varphi$. Thus, $\mathfrak{M}_1, w_1 \not\equiv \mathfrak{M}_2, w_2$.

Proposition 9 shows that finite pivotal models form a Hennessy-Milner class for PML [10]. Intuitively, this means that over this class, model equivalence implies bisimulation – which, for the case of PML, coincides with equality. However, model equivalence is not enough to ensure equality in infinite models. To see why, consider the model $\mathfrak{M}_1 = \{v_i \mid i \in \mathbb{N}, v_i(p_j) = 1 \text{ iff } 0 \leq j \leq i\}$, and $\mathfrak{M}_2 = \mathfrak{M}_1 \cup \{v\}$ with $v(p_j) = 1$ for all $j \in \mathbb{N}$. Clearly $\mathfrak{M}_1 \neq \mathfrak{M}_2$ but when restricted to a finite number of propositional symbols they coincide (i.e., they satisfy the same formulas of PML). Hence $\mathfrak{M}_1, v_1 \equiv \mathfrak{M}_2, v_1$.

Complexity. Finally, we characterize the complexity of checking pivotal consequence by using known complexity results for PAL [13]. To the best of our knowledge, Theorem 4 establishes the exact complexity of pivotal consequence for the first time.

Theorem 4. *Deciding $\{\varphi\} \vdash_P \psi$ is coNP-complete.*

Proof. coNP-hardness is clear by the coNP-hardness of checking consequence in CPL. Theorem 2 shows that $\{\varphi\} \vdash_P \psi$ is equivalent to $\{\varphi\} \models_P \psi$, which (by Definition 15) is the case iff there is a sequence $\bar{\rho}$ of pivotal rules in P s.t. $[\varphi]\bar{\rho}\psi$ is a validity in PML. We know that PML is just a notational variant of single agent PAL. This means that the results of [13] apply to PML. Then, there is a polynomial satisfiability preserving translation (for S5-models) T of formulas from PML into the basic modal language (i.e., formulas without the $[\_]_{\text{modality}}$). As a consequence of the results in [13], checking satisfiability of a PML-formula is NP-complete. This means that, given $\{\varphi\} \vdash_P \psi$, we can guess a witness $(\bar{\rho}, \mathfrak{M}, w)$ polynomial in (φ, ψ, P) such that $\mathfrak{M}, w \not\models T([\varphi]\bar{\rho}\psi)$. It is easy to check, in polynomial time, that $(\bar{\rho}, \mathfrak{M}, w)$ is a proper witness:

1. $\bar{\rho}$ should be a sequence, without repetitions, in P^* ;
2. $\mathfrak{M}, w$ should be an S5-model that satisfies $\neg T([\varphi]\bar{\rho}\psi)$.

Since satisfiability is NP-complete, deciding $\{\varphi\} \vdash_P \psi$ is coNP-complete.

4.2 Credulous Consequence in PAL

Having dealt with pivotal consequence, we turn our attention to dealing with credulous consequence from a semantic perspective. At this point we fully exploit and take advantage of the expressive power of the modalities of PML.

The key idea is to use the existential modality E to capture the consistency constraint on the use of pivotal rules. We make this precise in Proposition 10 below.

Proposition 10. *For all $\{\varphi, \psi\} \subseteq \text{Form}_0$, $\models [\varphi]E\psi$ implies $\{\varphi\} \not\models_{\text{CPL}} \neg\psi$.*

It is worth mentioning that the converse of Proposition 10 fails to hold in general. To see why, notice that, e.g., $\not\models_{\text{CPL}} \neg p$ but $\not\models [\top]Ep$. That $\not\models_{\text{CPL}} \neg p$ is obvious. We build a witness for $\not\models [\top]Ep$ by taking $\mathfrak{M}$ to be a pivotal model comprised of a single $v \in \{0, 1\}^{\text{Prop}}$ s.t. $v(p) = 0$. Clearly, $\mathfrak{M}, v \not\models [\top]Ep$. So, $\not\models [\top]Ep$. This example brings to the surface why the converse of Proposition 10 may fail to hold in general: pivotal models, when seen as sets of CPL-assignments, do not necessarily include all CPL-counter-models. In the particular case of the example, $\mathfrak{M}$ does not contain at least one CPL-assignment v' s.t. $v'(p) = 1$. The point to be made is: the consistency check that we are after requires us not to miss relevant CPL-assignments. We overcome this issue by restricting our attention to a very particular pivotal model –introduced below.

Definition 17. Let $\mathfrak{M}^{\text{CPL}} = \{0, 1\}^{\text{Prop}}$, i.e., $\mathfrak{M}^{\text{CPL}}$ contains all CPL-assignments.

The converse of Proposition 10 holds immediately in $\mathfrak{M}^{\text{CPL}}$.

Proposition 11. *For all $\{\varphi, \psi\} \subseteq \text{Form}_0$, $\{\varphi\} \not\models_{\text{CPL}} \neg\psi$ implies $\mathfrak{M}^{\text{CPL}} \Vdash [\varphi]E\psi$.*

Let us now turn our attention onto how to use Definition 14 to mimic the effect of a single credulous P -detachment step.

Proposition 12. *Let (π/χ) be a pivotal rule, φ and ψ be CPL-formulas; then:
 $\models [\varphi](\pi \wedge E\chi/\chi)\psi$ implies $\{\varphi\} \models_{\text{CPL}} \pi$, $\{\varphi\} \not\models_{\text{CPL}} \neg\chi$, and $\{\varphi, \chi\} \models_{\text{CPL}} \psi$.*

Proof. Suppose that $\models [\varphi](\pi \wedge E\chi/\chi)\psi$. By definition, $\models [\varphi](\pi \wedge E\chi \wedge [\chi]\psi)$. It follows that: (a) $\models [\varphi]\pi$; (b) $\models [\varphi]E\chi$; and (c) $\models [\varphi][\chi]\psi$. We obtain $\{\varphi\} \models_{\text{CPL}} \pi$ using (a) and Item 2 in Proposition 6. We obtain $\{\varphi\} \not\models_{\text{CPL}} \neg\chi$ using (b) and Proposition 10. We obtain $\{\varphi, \chi\} \models_{\text{CPL}} \psi$ using (c) and Items 1 and 3 in Proposition 6.

Notice that $\models [\varphi](\pi \wedge E\chi/\chi)\psi$ does not imply $\{\varphi\} \models \chi$. The condition $E\chi$ is sufficiently strong to guarantee a consistency check in CPL, but not strong enough to warrant consequence in CPL. Unsurprisingly, as with Proposition 11, the converse of Proposition 12 does not hold in general, but it does over $\mathfrak{M}^{\text{CPL}}$.

Proposition 13. *Let (π/χ) be a pivotal rule, φ and ψ be CPL-formulas; then:*

$$\{\varphi\} \models_{\text{CPL}} \pi, \{\varphi\} \not\models_{\text{CPL}} \neg\chi, \text{ and } \{\varphi, \chi\} \models_{\text{CPL}} \psi \text{ iff } \mathfrak{M}^{\text{CPL}} \Vdash [\varphi](\pi \wedge E\chi/\chi)\psi.$$

Proof. Right to left: Immediate from Proposition 12. Left to right: We prove the contrapositive, i.e., $\mathfrak{M}^{\text{CPL}} \not\models [\varphi](\pi \wedge E\chi/\chi)\psi$ implies (a) $\{\varphi\} \not\models_{\text{CPL}} \pi$ or (b) $\{\varphi\} \models_{\text{CPL}} \neg\chi$ or (c) $\{\varphi, \chi\} \not\models_{\text{CPL}} \psi$. Suppose $\mathfrak{M}^{\text{CPL}} \not\models [\varphi](\pi \wedge E\chi/\chi)\psi$; by definition $\mathfrak{M}^{\text{CPL}} \not\models [\varphi](\pi \wedge E\chi \wedge [\chi]\psi)$. It follows that either (a') $\mathfrak{M}^{\text{CPL}} \not\models [\varphi]\pi$; or (b') $\mathfrak{M}^{\text{CPL}} \not\models [\varphi]E\chi$; or (c') $\mathfrak{M}^{\text{CPL}} \not\models [\varphi][\chi]\psi$. We obtain (a) and (c) using (a') and (c'), and Proposition 6. We obtain (b) using the contrapositive of Proposition 11.

We have used $(\pi \wedge E\chi/\chi)\psi$ to capture the effect of a single *credulous* P -detachment step in semantic terms. To be able to compose the effect of successive credulous P -detachment steps at the level of models we revise Proposition 8.

Definition 18. *Let $\bar{\rho} = \rho_1 \dots \rho_n$ be a sequence of pivotal rules, where each ρ_i is of the form (π_i/χ_i) ; define $\bar{\rho}_c = (\pi_1 \wedge E\chi_1/\chi_1) \dots (\pi_n \wedge E\chi_n/\chi_n)$.*

Proposition 14. *Let $\bar{\rho} = \rho_1 \dots \rho_n$ be a sequence of pivotal rules; and φ, ψ be CPL-formulas. It follows that $\mathfrak{M}^{\text{CPL}} \Vdash [\varphi]\bar{\rho}_c\psi$ is equivalent to*

$$\{\varphi\} \cup X \models_{\text{CPL}} \psi, \text{ and} \\ \text{for all } 1 \leq i \leq n, \{\varphi\} \cup X_{(i-1)} \models_{\text{CPL}} \pi_i \text{ and } \{\varphi\} \cup X_{(i-1)} \not\models_{\text{CPL}} \neg\chi_i,$$

where: $\rho_i = (\pi_i/\chi_i)$; $X_j = \{\chi_1, \dots, \chi_j\}$; and $X = X_n$.

Definition 19 allows us to formulate credulous consequence using announcements.

Definition 19. Let P be a set of pivotal rules, and φ and ψ be CPL-formulas; define $\{\varphi\} \approx_P \psi$ iff exists a sequence $\bar{\rho}$ of pivotal rules in P s.t. $\mathfrak{M}^{\text{CPL}} \models [\varphi]\bar{\rho}_c\psi$.

Theorem 5 establishes the parallel between $\vdash_P$ and $\approx_P$. Its proof follows from Propositions 4 and 14 and Theorem 1.

Theorem 5. Let P be a set of pivotal rules, φ and ψ be CPL-formulas; then:

$$\{\varphi\} \vdash_P \psi \text{ iff } \{\varphi\} \approx_P \psi.$$

Theorem 5 tells us that ψ is a credulous consequence of φ given P iff $[\varphi]\bar{\rho}_c\psi$ holds everywhere in $\mathfrak{M}^{\text{CPL}}$. In this way, Theorem 5 reduces witnessing credulous consequence to a model checking problem in a particular pivotal model.

Transfer Results. We finish this section discussing some transference results that can be obtained by virtue of Theorem 5.

Axiom System. We begin with a discussion on the axiomatization of $\vdash_P$. Given Theorem 5, we would need for PML formulas to be evaluated in $\mathfrak{M}^{\text{CPL}}$. This makes the axiom system in Table 1 insufficient. However, it is not a major obstacle. Let us make an abuse of notation and write $\mathfrak{M}^{\text{CPL}}$ for the class of models containing solely $\mathfrak{M}^{\text{CPL}}$. The sole model in $\mathfrak{M}^{\text{CPL}}$ is needed to capture the “consistency check” required in the credulous use of pivotal rules. $\mathfrak{M}^{\text{CPL}}$ is characterized in Definition 20.

Definition 20. Let $P = \{p_1, \dots, p_n\}$ and $Q = \{q_1, \dots, q_m\}$ be finite subsets of Prop s.t. $P \cap Q = \emptyset$; define $\text{Val}(P, Q) = E((p_1 \wedge \dots \wedge p_n) \wedge (\neg q_1 \wedge \dots \wedge \neg q_m))$. Moreover, define VAL as the set of all $\text{Val}(P, Q)$.

Theorem 6. The axioms and rules in Table 1 plus VAL yield a sound and complete axiom system for PML over the class $\mathfrak{M}^{\text{CPL}}$.

Complexity. Our last result concerns complexity of deciding default consequence. We provide an upper bound, whose proof relies on the model checking problem for modal logic with the particular model $\mathfrak{M}^{\text{CPL}}$.

Similarly to the case of pivotal consequence, we use Theorem 5, i.e., $\{\varphi\} \vdash_P \psi$ is equivalent to $\{\varphi\} \approx_P \psi$, which is the case (from Definition 19) iff there is a sequence $\bar{\rho}$ of pivotal rules in P s.t. $\mathfrak{M}^{\text{CPL}} \models [\varphi]\bar{\rho}_c\psi$. On a first attempt, we could try to use the same strategy employed in Theorem 4, i.e., guess a polynomial witness $(\bar{\rho}, \mathfrak{M}, w)$. However, notice that, for credulous consequence, $\mathfrak{M}$ cannot be an arbitrary S5-model –since we need to check the satisfiability of $[\varphi]\bar{\rho}_c\psi$ in $\mathfrak{M}^{\text{CPL}}$ (of course, we can use $\mathfrak{M}^{\text{CPL}}$ restricted to the signature of $[\varphi]\bar{\rho}_c\psi$, which is finite, but not polynomial in the input). If we guess an arbitrary polynomial S5-model $\mathfrak{M}$, it is unclear how we can check in polynomial time that it is a proper witness (and the results in [12], see below, imply that being able to do so would result in a collapse of the polynomial hierarchy). In spite of these issues, we can directly establish a PSpace upper bound.

Theorem 7. *Deciding $\{\varphi\} \vdash_P \psi$ is in PSpace.*

Proof. We only need to show that it is possible to model check $[\varphi]\bar{\rho}_c\psi$ in $\mathfrak{M}^{\text{CPL}}$, on the fly, using only polynomial space.

Because $\text{PSpace} = \text{NPSpace}$ we can, non-deterministically, model check $[\varphi]\bar{\rho}_c\psi$ for a particular, arbitrary, $\bar{\rho}$. The following algorithm $A(\varphi)$ checks that an arbitrary PML-formula φ holds in the restriction of $\mathfrak{M}^{\text{CPL}}$ to the signature of φ in polynomial space.

Let φ be a PML-formula, and i be the largest index s.t. p_i appears in φ . Let $v_1, \dots, v_{2^i}$ be an enumeration of all the possible assignments over $p_1, \dots, p_i$, and assume that numbers are represented in binary. Let

$$A(\varphi) := \text{for } k = 1 \text{ to } 2^i \text{ do } F(\varphi, k, \{1, \dots, 2^i\});$$

where $F(\psi, k, S)$ is defined recursively as follows:

1. if $\psi = \perp$ then return False;
2. if $\psi = p_j$ then return $v_k(p_j) = 1$;
3. if $\psi = \alpha \rightarrow \beta$ then return not $F(\alpha, k, S)$ or $F(\beta, k, S)$;
4. if $\psi = A\alpha$ then
 - for $j \in S$ do
 - if not $F(\alpha, j, S)$ then return False;
 - return True;
5. if $\psi = [\alpha]\beta$ then
 - if not $F(\alpha, k, S)$ then return True;
 - $S' = \{j \mid F(\alpha, j, S)\}$;
 - return $F(\beta, k, S')$;

Remark 2. The complexity of reasoning in different non-monotonic logics has been investigated in [12]. In particular, [12] shows that the problem of deciding credulous consequence is Σ_2^P -complete. Such a result is exact; and, thus, it is stricter than ours. Nonetheless, we believe that Theorem 7 gives a simpler proof of a reasonable small upper bound via a transfer result.

5 Final Remarks

We presented a semantic exploration of the concept of PRC proposed in [14, 15] using the standard semantic machinery of PAL. In doing this, we connected PRC and PAL. In more detail, we obtained characterization results of PRC and credulous consequence in PAL. These characterization results led to natural axiomatizations and completeness results. Moreover, they led to transference results of expressivity and complexity. Interestingly, in our framework, credulous consequence turns out to be a model checking problem.

Our main results are Theorems 2 and 5. These can be seen as completeness results that show that the semantics of (a clearly defined fragment of) PAL captures PRC and credulous consequence. When seen in this light our work

complements the picture presented in [14,15], where these notions are defined as closure operators on sets of formulas.

We consider our work a first step towards studying different notions of non-monotonic consequence from a dynamic logic perspective. The results in [14,15] are not restricted to PRC, they show other ways of building *bridges* from classical to non-monotonic logic. We would also like to explore these other bridges in our setting. Moreover, it would be interesting to generalize the notion of PRC to logics more expressive than CPL, e.g., modal logics, and to study whether or not we need to extend PAL to characterize these logics.

Our work also opens up the door to explore the use of reasoning methods from modal logic, i.e., tableaux systems and model checkers, in the setting of PRC and in Default Logic. Modal logics boast a trove of automated tools and techniques which could now be applied to non-monotonic reasoning.

Acknowledgments. Our work is supported by ANPCyT-PICT-2020-3780, CONICET project PIP 11220200100812CO, and by the LIA SINFIN.

References

1. Antoniou, G., Wang, K.: Default logic. In: Gabbay, D., Woods, J. (eds.) *The Many Valued and Nonmonotonic Turn in Logic*, *Handbook of the History of Logic*, vol. 8, pp. 517–555. North-Holland (2007)
2. van Benthem, J.: Modal correspondence theory. *Handb. Philos. Log.* **2**, 167–247 (1984)
3. van Benthem, J.: Logical dynamics meets logical pluralism? *Australas. J. Log.* **6**, 182–209 (2008)
4. Besnard, P., Schaub, T.: Possible worlds semantics for default logics. *Fund. Inform.* **21**(1/2), 39–66 (1994)
5. Blackburn, P., van Benthem, J., Wolter, F. (eds.): *Handbook of Modal Logic*. Elsevier (2007)
6. Cerdón-Franco, A., van Ditmarsch, H., Nepomuceno-Fernández, Á.: Dynamic consequence and public announcement. *Rev. Symbolic Log.* **6**(4), 659–679 (2013)
7. Denecker, M., Marek, V., Truszczyński, M.: Uniform semantic treatment of default and autoepistemic logics. *Artif. Intell.* **143**(1), 79–122 (2003)
8. Ditmarsch, H.V., van der Hoek, W., Kooi, B.: *Dynamic Epistemic Logic*. Springer, Heidelberg (2007). <https://doi.org/10.1007/978-1-4020-5839-4>
9. Etherington, D.: A semantics for default logic. In: *Proceedings of the 10th International Joint Conference on Artificial Intelligence (IJCAI 1987)*, pp. 495–498. Morgan Kaufmann (1987)
10. Goldblatt, R.: Saturation and the hennessy-milner property. In: Ponse, A., de Rijke, M., Venema, Y. (eds.) *Modal Logic and Process Algebra: A Bisimulation Perspective*, pp. 107–129. Cambridge University Press (1995)
11. Goranko, V., Passy, S.: Using the universal modality: gains and questions. *J. Log. Comput.* **2**(1), 5–30 (1992)
12. Gottlob, G.: Complexity results for nonmonotonic logics. *J. Log. Comput.* **2**(3), 397–425 (1992)
13. Lutz, C.: Complexity and succinctness of public announcement logic. In: *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multi-agent Systems (AAMAS 2006)*, pp. 137–143. ACM (2006)

14. Makinson, D.: Bridges between classical and nonmonotonic logic. *Log. J. IGPL* **11**(1), 69–96 (2003)
15. Makinson, D.: *Bridges from Classical to Nonmonotonic Logic*, Texts in Computing, vol. 5. College Publications (2005)
16. Mendelson, E.: *Introduction to Mathematical Logic*, 5th edn. Chapman & Hall/CRC (2009)
17. Plaza, J.: Logics of public communications. *Synthese* **158**(2), 165–179 (2007). <https://doi.org/10.1007/s11229-007-9168-7>
18. Reiter, R.: A logic for default reasoning. *Artif. Intell.* **13**(1–2), 81–132 (1980)
19. Roy, O., Hjortland, O.T.: Dynamic consequence for soft information. *J. Log. Comput.* **26**(6), 1843–1864 (2016)

Author Index

- Afshari, Bahareh 309
Aguilera, Juan Pablo 18
Akbar Tabatabai, Amirhossein 378
Aman, Bogdan 123
Areces, Carlos 395
- Bednarczyk, Bartosz 295
- Cassano, Valentin 395
Ciobanu, Gabriel 123
- Dalmonte, Tiziano 280
De Domenico, Andrea 326
de Groote, Philippe 105
Diéguez, Martín 18
- Farjami, Ali 326
Fernández-Duque, David 18, 36
Fervari, Raul 395
Fiuk, Oskar 295
- Galimullin, Rustam 190
Galliani, Pietro 263
Gattinger, Malvin 237
Gougeon, Quentin 36
Grellois, Charles 280
- Kirst, Dominik 344
- Liu, Xinghan 158
Lorini, Emiliano 158
Ludmann, Pierre 105
- Maieli, Roberto 1
Manoorkar, Krishna 326
Matsuda, Naosuke 175
McLean, Brett 18
- Olivetti, Nicola 280
Omidvar, Giti 141
- Pacheco, Leonardo 207
Palmigiano, Alessandra 326
Panettiere, Mattia 326
Pedersen, Mina Young 190
Pogodalla, Sylvain 105
- Sanders, Sam 362
Sedlár, Igor 221
Šefl, Vít 250
Slavkovik, Marija 190
Straßburger, Lutz 141
Svoboda, Tomáš 250
- Takagi, Kento 175
Tanaka, Kazuyuki 207
- van Ditmarsch, Hans 237
- Wang, Xiaolong 326
Wang, Xiaoyang 70
Wang, Yanjing 70
Wannenburg, Johann J. 221
Wehr, Dominik 309, 344
- Yamakami, Tomoyuki 88
- Zhao, Zhiguang 53