

Luke Ong
Ruy de Queiroz (Eds.)

LNCS 7456

Logic, Language, Information and Computation

19th International Workshop, WoLLIC 2012
Buenos Aires, Argentina, September 2012
Proceedings

 Springer



Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison, UK

Josef Kittler, UK

Alfred Kobsa, USA

John C. Mitchell, USA

Oscar Nierstrasz, Switzerland

Bernhard Steffen, Germany

Demetri Terzopoulos, USA

Gerhard Weikum, Germany

Takeo Kanade, USA

Jon M. Kleinberg, USA

Friedemann Mattern, Switzerland

Moni Naor, Israel

C. Pandu Rangan, India

Madhu Sudan, USA

Doug Tygar, USA

FoLLI Publications on Logic, Language and Information

Subline of Lectures Notes in Computer Science

Subline Editors-in-Chief

Valentin Goranko, *Technical University, Lyngby, Denmark*

Erich Grädel, *RWTH Aachen University, Germany*

Michael Moortgat, *Utrecht University, The Netherlands*

Subline Area Editors

Nick Bezhanishvili, *Imperial College London, UK*

Anuj Dawar, *University of Cambridge, UK*

Philippe de Groote, *Inria-Lorraine, Nancy, France*

Gerhard Jäger, *University of Tübingen, Germany*

Fenrong Liu, *Tsinghua University, Beijing, China*

Eric Pacuit, *Tilburg University, The Netherlands*

Ruy de Queiroz, *Universidade Federal de Pernambuco, Brazil*

Ram Ramanujam, *Institute of Mathematical Sciences, Chennai, India*

Luke Ong Ruy de Queiroz (Eds.)

Logic, Language, Information and Computation

19th International Workshop, WoLLIC 2012
Buenos Aires, Argentina, September 3-6, 2012
Proceedings



Springer

Volume Editors

Luke Ong
University of Oxford
Department of Computer Science
Wolfson Building, Parks Road
Oxford OX1 3QD, UK
E-mail: lo@cs.ox.ac.uk

Ruy de Queiroz
Universidade Federal de Pernambuco
Centro de Informática
Av. Jornalista Anibal Fernandes, s/n
50740-560 Recife, PE, Brazil
E-mail: ruy@cin.ufpe.br

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-32620-2 e-ISBN 978-3-642-32621-9
DOI 10.1007/978-3-642-32621-9
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012943843

CR Subject Classification (1998): F.4.1, F.2-4, I.2.3, G.2, E.1, I.1

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

© Springer-Verlag Berlin Heidelberg 2012

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, 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.

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume contains the papers of the 19th Workshop on Logic, Language, Information and Computation (WoLLIC 2012), which was held during September 3–6, 2012 at the University of Buenos Aires.

The WoLLIC series of workshops series 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 for dialogues between logic and the sciences relating to information and computation, and yet small enough for interactions among participants.

A total of 16 papers were accepted out of 46 submissions for presentation at WoLLIC 2012 and for inclusion in the proceedings. Each submitted paper was reviewed by at least three members of the Program Committee, who were assisted in their work by 62 external reviewers. We would like to thank the members of the Program Committee and the external reviewers for their review work, as well as Andrei Voronkov for providing the EasyChair system that proved invaluable throughout the review process and the preparation of this volume. In addition to the contributed papers, the WoLLIC program contained invited lectures by Andrea Asperti (Bologna), Hans van Ditmarsch (Sevilla), Laura Kallmeyer (Düsseldorf), George Metcalfe (Nashville), Anca Muscholl (Bordeaux), Andre Nies (Auckland), Peter Selinger (Halifax), and Nicole Schweikardt (Frankfurt).

Many people helped to make WoLLIC 2012 a success. We would like to thank Carlos Areces (Local Co-chair), Santiago Figueira (Local Co-chair), Javier Legris, and Anjolina G. de Oliveira; our sponsors: Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires and Facultad de Ciencias Económicas, Universidad de Buenos Aires, and Centro de Informática, Universidade Federal de Pernambuco, Brazil. Last, but not least, we gratefully acknowledge the sponsorship of the following organizations: Interest Group in Pure and Applied Logics (IGPL), Association for Logic, Language and Information (FoLLI), Association for Symbolic Logic (ASL), European Association for Theoretical Computer Science (EATCS), European Association for Computer Science Logic (EACSL), Sociedade Brasileira de Computação (SBC), and Sociedade Brasileira de Lógica (SBL).

June 2012

Luke Ong
Ruy de Queiroz

WoLLIC 2012 Conference Organization

Program Chair

Luke Ong University of Oxford, UK

Program Committee

Carlos Areces	Universidad Nacional de Córdoba, Argentina
Marcelo Arenas	Pontificia Universidad Católica, Chile
Steve Awodey	Carnegie Mellon University, USA
Verónica Becher	Universidad de Buenos Aires, Argentina
Patrick Blackburn	Roskilde University, Denmark
Maribel Fernandez	King's College London, UK
Santiago Figueira	Universidad Nacional de Córdoba, Argentina
Marcelo Finger	Universidade de Sao Paulo, Brazil
Marcelo Fiore	University of Cambridge, UK
Yuxi Fu	Shanghai Jiaotong University, China
Rosalie Iemhoff	Utrecht University, The Netherlands
Neil Immerman	University of Massachusetts, USA
Jean-Pierre Jouannaud	INRIA, France and Tsinghua University, China
Makoto Kanazawa	National Institute of Informatics, Japan
Delia Kesner	CNRS and Université Paris 7, France
Dexter Kozen	Cornell University, USA
Martin Lange	University of Kassel, Germany
Benedikt Löwe	Universiteit van Amsterdam, The Netherlands
Dag Normann	University of Oslo, Norway
Luke Ong	University of Oxford, UK
Erik Palmgren	Stockholm University, Sweden
Sylvain Salvati	INRIA, France
Philippe Schnoebelen	CNRS and ENS Cachan, France
Fernando Souza	Universidade Federal de Pernambuco, Brazil
Kazushige Terui	Kyoto University, Japan

Conference Co-chairs

Carlos Areces	Universidad Nacional de Córdoba, Argentina
Santiago Figueira	Universidad de Buenos Aires, Argentina

Organizing Committee

Carlos Areces	Universidad Nacional de Córdoba, Argentina
Santiago Figueira	Universidad de Buenos Aires, Argentina
Javier Legris	Universidad de Buenos Aires, Argentina
Anjolina G. de Oliveira	Universidade Federal de Pernambuco, Brazil
Ruy de Queiroz	Universidade Federal de Pernambuco, Brazil

WoLLIC Steering Committee Chair

Ruy de Queiroz	Universidade Federal de Pernambuco, Brazil
----------------	--

WoLLIC Steering Committee

Samson Abramsky	University of Oxford, UK
Johan van Benthem	University of Amsterdam, The Netherlands
Anuj Dawar	University of Cambridge, UK
Joe Halpern	Cornell University, USA
Wilfrid Hodges	Queen Mary, University of London, UK
Daniel Leivant	Indiana University, USA
Angus Macintyre	Queen Mary, University of London, UK
Grigori Mints	Stanford University, USA
Hiroakira Ono	Japan Advanced Institute of Science and Technology, Japan
Ruy de Queiroz	Universidade Federal de Pernambuco, Brazil

Additional Reviewers

Arisaka, Ryuta	Delaune, Stephanie
Arratia, Argimiro	Dung, Phan Minh
Atserias, Albert	Forssell, Henrik
Balat, Vincent	Giese, Martin
Barpalias, George	Godo, Lluís
Berwanger, Dietmar	Hansen, Jens Ulrik
Bodirsky, Manuel	He, Chaodong
Boldo, Sylvie	Holz, Rupert
Broersen, Jan	Kapulkin, Chris
Bursuc, Sergiu	Kishida, Kohei
Busaniche, Manuela	Koller, Alexander
Caicedo Ferrer, Xavier	Lazic, Ranko
Carnielli, Walter	Leroux, Jerome
Cignoli, Roberto	Loeding, Christof
Cruz-Filipe, Luis	Long, Huan
D'Agostino, Marcello	Longin, Dominique

Lozes, Etienne
Lu, Pinyan
Manya, Felip
Monin, Jean-Francois
Mostrous, Dimitris
Moyen, Jean-Yves
Nguyen Van Thé, Lionel
Nies, André
Oitavem, Isabel
Pacuit, Eric
Pedersen, Arthur Paul
Reimann, Jan
Romeijn, Jan-Willem
Rossman, Benjamin

Slaman, Theodore
Sojakova, Kristina
Terrell, Jeffrey
Terwijn, Sebastiaan
Uckelman, Sara L.
Valiron, Benoît
Van Eijck, Jan
Westera, Matthijs
Winter, Yoad
Wu, Yining
Zeilberger, Noam
Zhao, Jianjun
Zhou, Xiaocong
Zivny, Stanislav

Table of Contents

Invited Papers and Abstracts of Invited Lectures

Formalizing Turing Machines	1
<i>Andrea Asperti and Wilmer Ricciotti</i>	
Equivalence Relations That Are Σ_3^0 Complete for Computable Reducibility (Extended Abstract)	26
<i>Ekaterina Fokina, Sy Friedman, and André Nies</i>	
An Analysis of Directed Motion Expressions with Lexicalized Tree Adjoining Grammars and Frame Semantics	34
<i>Laura Kallmeyer and Rainer Osswald</i>	
Admissible Rules: From Characterizations to Applications	56
<i>George Metcalfe</i>	
On Distributed Monitoring of Asynchronous Systems	70
<i>Volker Diekert and Anca Muscholl</i>	
On the Expressive Power of Logics with Invariant Uses of Arithmetic Predicates	85
<i>Nicole Schweikardt</i>	
Logical Methods in Quantum Information Theory	88
<i>Peter Selinger</i>	
Quantifying Notes	89
<i>Hans van Ditmarsch</i>	

Contributed Papers

Linearizing Bad Sequences: Upper Bounds for the Product and Majoring Well Quasi-orders	110
<i>Sergio Abriola, Santiago Figueira, and Gabriel Senno</i>	
Initiality for Typed Syntax and Semantics	127
<i>Benedikt Ahrens</i>	
Moving Arrows and Four Model Checking Results	142
<i>Carlos Areces, Raul Fervari, and Guillaume Hoffmann</i>	
Standard Completeness for Extensions of MTL: An Automated Approach	154
<i>Paolo Baldi, Agata Ciabattoni, and Lara Spendier</i>	

The Logic of Justified Belief Change, Soft Evidence and Defeasible Knowledge	168
<i>Alexandru Baltag, Bryan Renne, and Sonja Smets</i>	
Minimization via Duality	191
<i>Nick Bezhanishvili, Clemens Kupke, and Prakash Panangaden</i>	
On Some Subclasses of the Fodor-Roubens Fuzzy Bi-implication	206
<i>Claudio Callejas, João Marcos, and Benjamín René Callejas Bedregal</i>	
Linearity in the Non-deterministic Call-by-Value Setting	216
<i>Alejandro Díaz-Caro and Barbara Petit</i>	
Polynomial-Time Solution of Initial Value Problems Using Polynomial Enclosures	232
<i>Amin Farjudian</i>	
Algorithmic Randomness and Ramsey Properties of Countable Homogeneous Structures	246
<i>Willem L. Fouché</i>	
Propositional Reasoning about Saturated Conditional Probabilistic Independence	257
<i>Sebastian Link</i>	
Contracting Logics	268
<i>Márcio M. Ribeiro and Marcelo E. Coniglio</i>	
A Tight Upper Bound on the Number of Variables for Average-Case k -Clique on Ordered Graphs	282
<i>Benjamin Rossman</i>	
Preservation under Substructures modulo Bounded Cores	291
<i>Abhisekh Sankaran, Bharat Adsul, Vivek Madan, Pritish Kamath, and Supratik Chakraborty</i>	
A Logic of Plausible Justifications	306
<i>L. Menasché Schechter</i>	
Classic-Like Cut-Based Tableau Systems for Finite-Valued Logics	321
<i>Marco Volpe, João Marcos, and Carlos Caleiro</i>	
Author Index	337

Formalizing Turing Machines

Andrea Asperti and Wilmer Ricciotti

Department of Computer Science, University of Bologna
{[asperti](mailto:asperti@cs.unibo.it),[ricciott](mailto:ricciott@cs.unibo.it)}@cs.unibo.it

Abstract. We discuss the formalization, in the Matita Theorem Prover, of a few, basic results on Turing Machines, up to the existence of a (certified) Universal Machine. The work is meant to be a preliminary step towards the creation of a formal repository in Complexity Theory, and is a small piece in our Reverse Complexity program, aiming to a comfortable, machine independent axiomatization of the field.

1 Introduction

We have assisted, in recent years, to remarkable achievements obtained by means of interactive theorem provers for the formalization and automatic checking of complex results in many different domains, spanning from pure mathematics [\[9,14,5\]](#) to software verification [\[19,18,25,1\]](#), passing through the metatheory and semantics of programming languages [\[10,24\]](#).

Surprisingly, however, very little work has been done so far in major fields of theoretical computer science, such as computability theory and, especially, complexity theory. The only work we are aware of is [\[20\]](#), containing basic results in computability theory relying on λ -calculus and recursive functions as computational models. The computational constructs of both these models are not finitistic and are not very suitable for complexity purposes: Turing Machines still provide the standard foundation for this discipline.

Our work is an initial, preliminary contribution in this direction. In particular, we present a formalization of basic definitions and results on Turing Machines, up to the existence of a universal machine and the proof of its correctness. In particular, in Section 2 we discuss the notion of Turing Machine and its semantics; Section 3 provides means for composing machines (sequential composition, conditionals and iteration); Section 4 contains the definition of basic, atomic machines for textual manipulation of the tape; Section 5 introduces the notion of Normal Turing Machine and its standard representation as a list of tuples; Section 6 gives an outline of the universal machine; Section 7 and 8 are respectively devoted to the two main routines of the universal machine, namely *finding* the right tuple to apply, and *executing* the corresponding action; in Section 10, we summarize the main results which have been proved about the universal machine. In the conclusion we provide overall information about the size of the contribution and the resources required for its development as well as more motivations for pursuing formalization in computability and complexity theory: in particular we shall briefly outline our long term *Reverse Complexity* program,

aiming to a trusted, comfortable, machine independent axiomatization of the field suitable for mechanization.

In our development, we have been inspired by several traditional articles and textbooks, comprising e.g. [13,17,11,23,21]; however, it is worth to remark that none of them provides a description of the topic, and especially of universal machines, sufficiently accurate to be directly used as a guideline for formalization.

The formalization work described in this paper has been performed by means of the Matita Interactive Theorem Prover [8]. For lack of space we cannot provide details about proofs; the development will be part of the standard library of Matita since the next public release, and in the next few months will be made accessible on-line through the new Web interface of the system [6].

2 The Notion of Turing Machine

Turing Machines were defined by Alan M. Turing in [22]. To Computer Scientists, they are a very familiar notion, so we shall address straight away their formal definition. Let us just say that, for the purposes of this paper, we shall stick to deterministic, single tape Turing Machines. The generalization to multi-tape/non deterministic machines does not look problematic.¹

2.1 The Tape

The first problem is the definition of the tape. The natural idea is to formalize it as a zipper, that is a pair of lists l and r , respectively representing the portions of the tape at the left and the right of the tape head; by convention, we may assume the head is reading the first symbol on the right. Of course, the machine must be aware this list can be empty, that means that the transition function should accept an *optional* tape symbol as input. Unfortunately, in this way, the machine is only able to properly react to a right overflow; the problem arises when the left tape is empty and the head is moved to the left: a new “blank” symbol should be added to the right tape. A common solution in textbooks is to reserve a special blank character \sqcup of the tape alphabet for this purpose: the annoying consequence is that tape equality should be defined only up to a suitable equivalence relation ignoring blanks. To make an example, suppose we move the head to the left and then back to the right: we expect the tape to end up in the same situation we started with. However, if the tape was in the configuration $([], r)$ we would end up in $([\sqcup], r)$. As anybody with some experience in interactive proving knows very well, reasoning up to equivalence relations is *extremely* annoying, that prompts us to look for a different representation of the tape.

¹ It is worth to recall that the choice about the number of tapes, while irrelevant for computability issues, it is not from the point of view of complexity. Hartmanis and Stearns [15] have shown that any k -tape machine can be simulated by a one-tape machine with at most a quadratic slow-down, and Hennie [16] proved that in some cases this is the best we can expect; Hennie and Stearns provided an efficient simulation of multi-tape machines on a two-tape machine with just a logarithmic slow-down [12].

The main source of our problem was the asymmetric management of the left and right tape, with the arbitrary assumption that the head symbol was part of the right tape. If we try to have a more symmetric representation we must clearly separate the head symbol from the left and right tape, leading to a configuration of the kind (l, c, r) (mid-tape); if we have no c , this may happen for three different reasons: we are on the left end of a non-empty tape (left overflow), we are on the right end of a non-empty tape (right overflow), or the tape is completely empty.

This definition of the tape may seem conspicuous at first glance, but it resulted to be quite convenient.

```
inductive tape (sig:FinSet) : Type :=
| niltape : tape sig
| leftof   : sig → list sig → tape sig
| rightof  : sig → list sig → tape sig
| midtape  : list sig → sig → list sig → tape sig.
```

For instance, suppose to be in a configuration with an empty left tape, that is $(midtape [] a l)$; moving to the left will result in $(leftof a l)$; further moves to the left are forbidden (unless we write a character to the uninitialized cell, therefore turning the overflow into a mid-tape), and moving back to the right restores the original situation.

Given a tape, we may easily define the left and right portions of the tape and the optional current symbol (question marks and dots appearing in the code are implicit parameters that the type checker is able to infer by itself):

```
definition left := λsig.λt:tape sig.match t with
[ niltape ⇒ [] | leftof _ _ ⇒ [] | rightof s l ⇒ s::l | midtape l _ _ ⇒ l ].

definition right := λsig.λt:tape sig.match t with
[ niltape ⇒ [] | leftof s r ⇒ s::r | rightof _ _ ⇒ [] | midtape _ _ r ⇒ r ].

definition current := λsig.λt:tape sig.match t with
[ midtape _ c _ ⇒ Some ? c | _ ⇒ None ? ].
```

Note that if $(current\ t) = None$ than either $(left\ t)$ or $(right\ t)$ is empty.

2.2 The Machine

We shall consider machines with three possible moves for the head: L (left) R (right) and N (None).

```
inductive move : Type :=| L : move | R : move | N : move.
```

The machine, parametric over a tape alphabet sig , is a record composed of a finite set of *states*, a transition function *trans*, a *start* state, and a set of halting states identified by a boolean function. To encode the alphabet and the states, we exploit the FinSet library of Matita, making extensive use of *unification hints* [7].

```

record TM (sig:FinSet): Type :=
{ states : FinSet;
  trans : states × (option sig) → states × (option (sig × move));
  start : states;
  halt : states → bool}.

```

The transition function takes in input a pair $\langle q, a \rangle$ where q is the current internal state and a is the current symbol of the tape (hence, an optional character); it returns a pair $\langle q, p \rangle$ where p is an *optional* pair $\langle b, m \rangle$ composed of a new character and a move. The rationale is that if we write a new character we will always be allowed to move, also in case the current head symbol was *None*. However, we also want to give the option of not touching the tape (*NOP*), that is the intended meaning of returning *None* as output.

Executing p on the tape has the following effect:

```

definition tape_move := λsig.λt:tape sig.λp:option (sig × move).
match p with
| None ⇒ t
| Some p1 ⇒
  let ⟨s,m⟩ :=p1 in
  match m with
  | R ⇒ tape_move_right ? (left ? t) s (right ? t)
  | L ⇒ tape_move_left ? (left ? t) s (right ? t)
  | N ⇒ midtape ? (left ? t) s (right ? t) ] ].

```

where

```

definition tape_move_left := λsig:FinSet.λlt: list sig .λc:sig .λrt: list sig .
match lt with
| nil ⇒ leftof sig c rt
| cons c0 lt0 ⇒ midtape sig lt0 c0 (c::rt) ].

definition tape_move_right := λsig:FinSet.λlt: list sig .λc:sig .λrt: list sig .
match rt with
| nil ⇒ rightof sig c lt
| cons c0 rt0 ⇒ midtape sig (c::lt) c0 rt0 ].

```

A *configuration* relative to a given set of *states* and an alphabet *sig* is a record composed of a current internal state *cstate* and a *sig* tape.

```

record config (sig , states :FinSet): Type :=
{ cstate : states ;
  ctape: tape sig }.

```

A transition *step* between two configurations is defined as follows:

```

definition step := λsig.λM:TM sig.λc:config sig (states sig M).
let current_char :=current ? (ctape ?? c) in
let ⟨news,mv⟩ :=trans sig M ⟨cstate ?? c,current_char⟩ in
mk_config ?? news (tape_move sig (ctape ?? c) mv).

```

2.3 Computations

A computation is an iteration of the step function until a final internal state is met. In Matita, we may only define total functions, hence we provide an upper bound to the number of iterations, and return an optional configuration depending on the fact that the halting condition has been reached or not.

```

let rec loop (A:Type) n (f:A→A) p a on n :=
  match n with
  | 0 ⇒ None ?
  | S m ⇒ if p a then (Some ? a) else loop A m f p (f a) ].

```

The transformation between configurations induced by a Turing machine M is hence:

```

definition loopM := λsig,M,i,inc.
  loop ? i (step sig M) (λc.halt sig M (cstate ?? c)) inc.

```

The usual notion of computation for Turing Machines is defined according to given input and output functions, providing the initial tape encoding and the final read-back function. As we know from Kleene's normal form, the output function is particularly important: the point is that our notion of Turing Machine is monotonically increasing w.r.t. tape consumption, with the consequence that the transformation relation between configurations is decidable. However, input and output functions are extremely annoying when composing machines and we would like to get rid of them as far as possible.

Our solution is to define the semantics of a Turing Machine by means of a relation between the input tape and the final tape (possibly embedding the input and output functions): in particular, we say that a machine M *realizes* a relation R between tapes ($M \models R$), if for all t_1 and t_2 there exists a computation leading from $\langle q_0, t_1 \rangle$, to $\langle q_f, t_2 \rangle$ and $t_1 R t_2$, where q_0 is the initial state and q_f is some halting state of M .

```

definition initc := λsig.λM:TM sig.λt.
  mk_config sig (states sig M) (start sig M) t.

```

```

definition Realize := λsig.λM:TM sig.λR:relation (tape sig).

```

```

  ∀t.∃i.∃outc.

```

```

  loopM sig M i (initc sig M t) = Some ? outc ∧ R t (ctape ?? outc).

```

It is natural to wonder why we use relations on tapes, and not on configurations. The point is that different machines may easily *share* tapes, but they can hardly share their internal states. Working with configurations would force us to an input/output recoding between different machines that is precisely what we meant to avoid.

Our notion of realizability implies termination. It is natural to define a weaker notion (weak realizability, denoted $M \models\!\!\! \models R$), asking that $t_1 R t_2$ *provided* there is a computation between t_1 and t_2 . It is easy to prove that termination together with weak realizability imply realizability (we shall use the notation $M \downarrow t$ to express the fact that M terminates on input tape t).

definition $WRealize := \lambda \text{sig} . \lambda M : TM \text{ sig} . \lambda R : \text{relation} \text{ (tape sig)} .$
 $\forall t, i, \text{outc} .$
 $\text{loopM sig M } i \text{ (initc sig M } t) = \text{Some ? outc} \rightarrow R \ t \text{ (ctape ?? outc)} .$

definition $Terminate := \lambda \text{sig} . \lambda M : TM \text{ sig} . \lambda t . \exists i, \text{outc} .$
 $\text{loopM sig M } i \text{ (initc sig M } t) = \text{Some ? outc} .$

lemma $WRealize_to_Realize : \forall \text{sig} . \forall M : TM \text{ sig} . \forall R .$
 $(\forall t . M \downarrow t) \rightarrow M \Vdash R \rightarrow M \models R .$

2.4 A Canonical Relation

For every machine M we may define a canonical relation, that is the smallest relation weakly realized by M

definition $R_TM := \lambda \text{sig} . \lambda M : TM \text{ sig} . \lambda q . \lambda t1, t2 .$
 $\exists i, \text{outc} . \text{loopM ? M } i \text{ (mk_config ?? q } t1) = \text{Some ? outc} \wedge t2 = \text{ctape ?? outc} .$

lemma $Wrealize_R_TM : \forall \text{sig} , M .$
 $M \Vdash R_TM \text{ sig } M \text{ (start sig } M) .$

lemma $R_TM_to_R : \forall \text{sig} , M, R . \forall t1, t2 .$
 $M \Vdash R \rightarrow R_TM \ ? \ M \text{ (start sig } M) \ t1 \ t2 \rightarrow R \ t1 \ t2 .$

2.5 The Nop Machine

As a first, simple example, we define a Turing machine performing no operation (we shall also use it in the sequel to force, by sequential composition, the existence of a unique final state).

The machine has a single state that is both initial and final; the transition function is irrelevant, since it will never be executed.

The semantic relation R_nop characterizing the machine is just the identity and the proof that the machine realizes it is entirely straightforward.

in this case, states are defined as $initN \ 1$, that is the interval of natural numbers less than 1. This is actually a sigma type containing a natural number m and an (irrelevant) proof that it is smaller than n .

definition $\text{nop_states} := \text{initN } 1 .$
definition $\text{start_nop} : \text{initN } 1 := \text{mk_Sig} \ ? \ ? \ 0 \text{ (le_n ... 1)} .$

definition $\text{nop} := \lambda \alpha : \text{FinSet} .$
 $\text{mk_TM } \alpha \ \text{nop_states}$
 $(\lambda p . \text{let } \langle q, a \rangle := p \text{ in } \langle q, \text{None} \ ? \rangle)$
 $\text{start_nop } (\lambda _ . \text{true}) .$


```
definition R_nop := λalpha.λt1,t2:tape alpha.t2 = t1.
```

```
lemma sem_nop : ∀alpha.nop alpha ⊨ R_nop alpha.
```

3 Composing Machines

Turing Machines are usually reputed to suffer for a lack of compositionality. Our semantic approach, however, allows us to compose them in relatively easy ways. This will give us the opportunity to reason at a higher level of abstraction, rapidly forgetting their low level architecture.

3.1 Sequential Composition

The sequential composition $M_1 \cdot M_2$ of two Turing Machines M_1 and M_2 is a new machine having as states the disjoint union of the states of M_1 and M_2 . The initial state is the (injection of the) initial state of M_1 , and similarly the halting condition is inherited from M_2 ; the transition function is essentially the disjoint sum of the transition functions of M_1 and M_2 , plus a transition leading from the final states of M_1 to the (old) initial state of M_2 (here it is useful to have the possibility of not moving the tape).

```
definition seq_trans := λsig. λM1,M2 : TM sig.
```

```
λp. let ⟨s,a⟩ := p in
```

```
  match s with
```

```
  [ inl s1 ⇒
```

```
    if halt sig M1 s1 then ⟨inr ... (start sig M2), None ?⟩
```

```
    else let ⟨news1,m⟩ := trans sig M1 ⟨s1,a⟩ in ⟨inl ... news1,m⟩
```

```
  | inr s2 ⇒
```

```
    let ⟨news2,m⟩ := trans sig M2 ⟨s2,a⟩ in ⟨inr ... news2,m⟩
```

```
  ].
```

```
definition seq := λsig. λM1,M2 : TM sig.
```

```
  mk_TM sig
```

```
    (FinSum (states sig M1) (states sig M2))
```

```
    (seq_trans sig M1 M2)
```

```
    (inl ... (start sig M1))
```

```
    (λs.match s with [inl _ ⇒ false |inr s2 ⇒ halt sig M2 s2]).
```

If $M_1 \models R_1$ and $M_2 \models R_2$ then $M_1 \cdot M_2 \models R_1 \circ R_2$, that is a very elegant way to express the semantics of sequential composition. The proof of this fact, however, is not as straightforward as one could expect. The point is that M_1 works with its own internal states, and we should “lift” its computation to the states of the sequential machine.

To have an idea of the kind of results we need, here is one of the the key lemmas:

lemma `loop_lift` : $\forall A, B, k, \text{lift}, f, g, h, \text{hlift}, c, c1.$
 $(\forall x. \text{hlift} (\text{lift } x) = h x) \rightarrow$
 $(\forall x. h x = \text{false} \rightarrow \text{lift} (f x) = g (\text{lift } x)) \rightarrow$
 $\text{loop } A \ k \ f \ h \ c = \text{Some } ? \ c1 \rightarrow$
 $\text{loop } B \ k \ g \ \text{hlift} (\text{lift } c) = \text{Some } ? (\text{lift } \dots c1).$

It says that the result of iterating a function g starting from a lifted configuration $\text{lift } c$ is the same (up to lifting) as iterating a function f from c provided that

1. a base configuration is halting if and only if its lifted counterpart is halting as well;
2. f and g commute w.r.t. lifting on every non-halting configuration.

3.2 If Then Else

The next machine we define is an if-then-else composition of three machines M_1, M_2 and M_3 respectively implementing a boolean test, and the two conditional branches. One typical problem of working with single tape machines is the storage of intermediate results: using the tape is particularly annoying, since it requires moving the whole tape back and forward to avoid overwriting relevant information. Since in the case of the if-then-else the result of the test is just a boolean, it makes sense to store it in a state of the machine; in particular we expect to end up in a distinguished final state $qacc$ if the test is successful, and in a different state otherwise. This special state $qacc$ must be explicitly mentioned when composing the machines. The definition of the if-then-else machine is then straightforward: the states of the new machine are the disjoint union of the states of the three composing machines; the initial state is the initial state of M_1 ; the final states are the final states of M_2 and M_3 ; the transition function is the union of the transition functions of the composing machines, where we add new transitions leading from $qacc$ to the initial state of M_2 and from all other final states of M_1 to the initial state of M_2 .

definition `if_trans` := $\lambda \text{sig}. \lambda M1, M2, M3: \text{TM } \text{sig}. \lambda q: \text{states } \text{sig } M1. \lambda p.$
let $\langle s, a \rangle := p$ **in**
match s **with**
 [$\text{inl } s1 \Rightarrow$
 if $\text{halt } \text{sig } M1 \ s1$ **then**
 if $s1 = q$ **then** $\langle \text{inr } \dots (\text{inl } \dots (\text{start } \text{sig } M2)), \text{None } ? \rangle$
 else $\langle \text{inr } \dots (\text{inr } \dots (\text{start } \text{sig } M3)), \text{None } ? \rangle$
 else let $\langle \text{news1}, m \rangle := \text{trans } \text{sig } M1 \ \langle s1, a \rangle$ **in**
 $\langle \text{inl } \dots \text{news1}, m \rangle$
 | $\text{inr } s' \Rightarrow$
 match s' **with**
 [$\text{inl } s2 \Rightarrow \text{let } \langle \text{news2}, m \rangle := \text{trans } \text{sig } M2 \ \langle s2, a \rangle$ **in**
 $\langle \text{inr } \dots (\text{inl } \dots \text{news2}), m \rangle$
 | $\text{inr } s3 \Rightarrow \text{let } \langle \text{news3}, m \rangle := \text{trans } \text{sig } M3 \ \langle s3, a \rangle$ **in**
 $\langle \text{inr } \dots (\text{inr } \dots \text{news3}), m \rangle$]].

```

definition ifTM := λsig. λcondM,thenM,elseM:TM sig.λqacc: states sig condM.
mk_TM sig
  (FinSum (states sig condM) (FinSum (states sig thenM) (states sig elseM)))
  (if_trans sig condM thenM elseM qacc)
  (inl ... (start sig condM))
  (λs.match s with
    [ inl _ ⇒ false
    | inr s' ⇒ match s' with
      [ inl s2 ⇒ halt sig thenM s2
      | inr s3 ⇒ halt sig elseM s3 ] ] ).

```

Our realizability semantics is defined on tapes, and not configurations. In order to observe the accepting state we need to define a suitable variant that we call *conditional realizability*, denoted by $M \models [q : R_1, R_2]$. The idea is that M realizes R_1 if it terminates the computation on q , and R_2 otherwise.

```

definition accRealize := λsig.λM:TM sig.λacc:states sig M.λRtrue,Rfalse.
∀t.∃i.∃outc.
loopM sig M i (initc sig M t) = Some ? outc ∧
  (cstate ?? outc = acc → Rtrue t (ctape ?? outc)) ∧
  (cstate ?? outc ≠ acc → Rfalse t (ctape ?? outc)).

```

The semantics of the if-then-else machine can be now elegantly expressed in the following way:

```

lemma sem_if: ∀sig.∀M1,M2,M3:TM sig.∀Rtrue,Rfalse,R2,R3,acc.
M1 ⊨ [acc: Rtrue,Rfalse] → M2 ⊨ R2 → M3 ⊨ R3 →
  ifTM sig M1 M2 M3 acc ⊨ (Rtrue ∘ R2) ∪ (Rfalse ∘ R3).

```

It is also possible to state the semantics in a slightly stronger form: in fact, we know that if the test is successful we shall end up in a final state of M_2 and otherwise in a final state of M_3 . If M_2 has a single final state, we may express the semantics by a conditional realizability over this state. As we already observed, a simple way to force a machine to have a unique final state is to sequentially compose it with the nop machine. Then, it is possible to prove the following result (the conditional state is a suitable injection of the unique state of the nop machine):

```

lemma acc_sem_if: ∀sig,M1,M2,M3,Rtrue,Rfalse,R2,R3,acc.
M1 ⊨ [acc: Rtrue, Rfalse] → M2 ⊨ R2 → M3 ⊨ R3 →
  ifTM sig M1 (single_finalTM ... M2) M3 acc ⊨
  [inr ... (inl ... (inr ... start_nop)): Rtrue ∘ R2, Rfalse ∘ R3].

```

3.3 While

The last machine we are interested in, implements a while-loop over a body machine M . Its definition is really simple, since we have just to add to M a single transition leading from a distinguished final state q back to the initial state.

definition `while_trans` := $\lambda \text{sig}. \lambda M : \text{TM sig}. \lambda q : \text{states sig } M. \lambda p.$
let `(s, a)` := `p` **in**
if `s == q` **then** `(start ? M, None ?)`
else `trans ? M p`.

definition `whileTM` := $\lambda \text{sig}. \lambda M : \text{TM sig}. \lambda \text{qacc} : \text{states ? } M.$
`mk_TM sig`
`(states ? M)`
`(while_trans sig M qacc)`
`(start sig M)`
 $(\lambda s. \text{halt sig } M s \wedge \neg s == \text{qacc}).$

More interesting is the way we can express the semantics of the while machine: provided that $M \models [q : R_1, R_2]$, the while machine (relative to q) weakly realizes $R_1^* \circ R_2$:

theorem `sem_while`: $\forall \text{sig}, M, \text{qacc}, R_{\text{true}}, R_{\text{false}}.$
`halt sig M qacc = true` \rightarrow
 $M \models [\text{qacc} : R_{\text{true}}, R_{\text{false}}] \rightarrow$
`whileTM sig M qacc` $\models (\text{star ? } R_{\text{true}}) \circ R_{\text{false}}.$

In this case, the use of weak realizability is essential, since we are not guaranteed to exit the while loop, and the computation can actually diverge. Interestingly, we can reduce the termination of the while machine to the well foundedness of *Rtrue*:

theorem `terminate_while`: $\forall \text{sig}, M, \text{qacc}, R_{\text{true}}, R_{\text{false}}, t.$
`halt sig M qacc = true` \rightarrow
 $M \models [\text{qacc} : R_{\text{true}}, R_{\text{false}}] \rightarrow$
 $\text{WF ? (inv ... } R_{\text{true}}) t \rightarrow \text{whileTM sig } M \text{ qacc} \downarrow t.$

4 Basic Machines

A major mistake we made when we started implementing the universal machine consisted in modelling relatively complex behaviors by directly writing a corresponding Turing Machine. While writing the code is usually not very complex, proving its correctness is often a nightmare, due to the complexity of specifying and reasoning about internal states of the machines and all intermediate configurations. A much better approach consists in specifying a small set of basic machines, and define all other machines by means of the compositional constructs of the previous section. In this way, we may immediately forget about Turing Machines' internals, since the behavior of the whole program only depends on the behavior of its components.

A very small set of primitive programs turned out to be sufficient for our purposes (most of them are actually families of machines, parametrized over some input arguments).

write c write the character c on the tape at the current head position
move_r move the head one step to the right
move_l move the head one step to the left
test_char f perform a boolean test f on the current character and enter state tc_true or tc_false according to the result of the test
swap_r swap the current character with its right neighbor (if any)
swap_l swap the current character with its left neighbor (if any)

The specification of these machines is straightforward. Let us have a glance at the *swap_r* machine. In order to swap characters we need an auxiliary memory cell; since tape characters are finite, we may use an internal state (register) of the machine to this purpose. The machine will sequentially enter in the following four states:

- swap0: read the current symbol, save it in a register and move right
- swap1: swap the current symbol with the register content, and move back to the left
- swap2: write the register content at the current position
- swap3: stop

Here is the machine implementation:

```

definition swap_r :=
  λalpha:FinSet.λfoo:alpha.
  mk_TM alpha (swap_states alpha)
  (λp.let ⟨q,a⟩ :=p in
    let ⟨q',b⟩ :=q in
    let q' :=\fst q' in (* extract the witness *)
    match a with
    [ None ⇒ ⟨⟨swap3,foo⟩,None ?⟩ (* if tape is empty then stop *)
    | Some a' ⇒
    match q' with
    [ O ⇒ (* q0 *) ⟨⟨swap1,a'⟩,Some ? ⟨a',R⟩⟩ (* save in register and move R *)
    | S q' ⇒ match q' with
    [ O ⇒ (* q1 *) ⟨⟨swap2,a'⟩,Some ? ⟨b,L⟩⟩ (* swap with register and move L *)
    | S q' ⇒ match q' with
    [ O ⇒ (* q2 *) ⟨⟨swap3,foo⟩,Some ? ⟨b,N⟩⟩ (* copy from register and stay *)
    | S q' ⇒ (* q3 *) ⟨⟨swap3,foo⟩,None ?⟩ (* final state *)
    ] ] ] )
  ⟨swap0,foo⟩
  (λq.\fst q == swap3).

```

and this is its specification.

```

definition Rswap_r :=λalpha,t1,t2.
  ∀a,b,ls,rs. t1 = midtape alpha ls b (a::rs) → t2 = midtape alpha ls a (b::rs).

```

It is possibly worth to remark that an advantage of using relations is the possibility of under-specifying the behavior of the program, restricting the attention

to what we expect to be the structure of the input (e.g., in the previous case, the fact of receiving a mid-tape as the input tape).

The proof that *swap_r* realizes its specification is by cases on the structure of the tape: three cases are vacuous; the case when the tape is actually a mid-tape is essentially solved by direct computation.

4.1 Composing Machines

Let us see an example of how we can use the previous bricks to build more complex functions. When working with Turing Machines, moving characters around the tape is a very frequent and essential operation. In particular, we would like to write a program that moves a character to the left until we reach a special character taken as a parameter (*move_char_l*). A step of the machine essentially consists of a swap operation, but guarded by a conditional test; then we shall simply wrap a while machine around this step.

definition `mcl_step` := $\lambda\alpha:\text{FinSet}.\lambda\text{sep}:\alpha.$
`ifTM` α (`test_char` ? ($\lambda c.\neg c==\text{sep}$))
`(single_finalTM` ... (`swap_r` α `sep` · `move_l` ?)) (`nop` ?) `tc_true` .

definition `Rmcl_step_true` := $\lambda\alpha,\text{sep},t1,t2.$
 $\forall a,b,ls,rs.$
`t1 = midtape` α `ls b (a::rs)` \rightarrow
`b` \neq `sep` \wedge `t2 = mk_tape` α (`tail` ? `ls`) (`option_hd` ? `ls`) (`a::b::rs`) .

definition `Rmcl_step_false` := $\lambda\alpha,\text{sep},t1,t2.$
`right` ? `t1` \neq [] \rightarrow `current` α `t1` \neq `None` α \rightarrow
`current` α `t1` = `Some` α `sep` \wedge `t2` = `t1` .

definition `mcls_acc`: $\forall\alpha:\text{FinSet}.\forall\text{sep}:\alpha.$ `states` ? (`mcl_step` α `sep`)
:= $\lambda\alpha,\text{sep}.$ `inr` ... (`inl` ... (`inr` ... `start_nop`)).

lemma `sem_mcl_step` :
 $\forall\alpha,\text{sep}.$
`mcl_step` α `sep` \models
[`mcls_acc` α `sep`: `Rmcl_step_true` α `sep`, `Rmcl_step_false` α `sep`]

Here is the full *move_char_l* program:

definition `move_char_l` := $\lambda\alpha,\text{sep}.$
`whileTM` α (`mcl_step` α `sep`) (`mcls_acc` α `sep`) .

definition `R_move_char_l` := $\lambda\alpha,\text{sep},t1,t2.$
 $\forall b,a,ls,rs.$ `t1 = midtape` α `ls b (a::rs)` \rightarrow
`(b = sep` \rightarrow `t2 = t1)` \wedge
 $(\forall ls1,ls2. ls = ls1@\text{sep}::ls2 \rightarrow$
`b` \neq `sep` \rightarrow `memb` ? `sep` `ls1` = `false` \rightarrow
`t2 = midtape` α `ls2 sep (a::reverse` ? `ls1@b::rs`)).

```
lemma sem_move_char_l : ∀alpha,sep.
  WRealize alpha (move_char_l alpha sep) (R_move_char_l alpha sep).
```

In a very similar way, we may define two machines *move_left_to* and *move_right_to* that move the head left or right until they meet a character that satisfies a given condition.

5 Normal Turing Machines

A normal Turing machine is just an ordinary machine where:

1. the tape alphabet is $\{0, 1\}$;
2. the finite states are supposed to be an initial interval of the natural numbers.

By convention, we assume the starting state is 0.

```
record normalTM : Type :=
{ no_states : nat;
  pos_no_states : (0 < no_states);
  ntrans : (initN no_states) × Option bool → (initN no_states) × Option (bool × Move);
  nhalt : initN no_states → bool}.
```

We may easily define a transformation from a normal TM into a traditional Machine; declaring it as a coercion we allow the type system to freely convert the former into the latter:

```
definition normalTM_to_TM := λM:normalTM.
  mk_TM FinBool (initN (no_states M))
    (ntrans M) (mk_Sig ?? 0 (pos_no_states M)) (nhalt M).

coercion normalTM_to_TM.
```

A normal configuration is a configuration for a normal machine: it only depends on the number of states of the normal Machine:

```
definition nconfig := λn. config FinBool (initN n).
```

5.1 Tuples

By general results on FinSets (the Matita library about finite sets) we know that every function f between two finite sets A and B can be described by means of a finite graph of pairs $\langle a, fa \rangle$. Hence, the transition function of a normal Turing machine can be described by a finite set of tuples $\langle \langle i, c \rangle, \langle j, action \rangle \rangle$ of the following type:

$$(initN\ n \times option\ bool) \times (initN\ n \times option\ bool \times move)$$

Unfortunately, this description is not suitable for a Universal Machine, since such a machine must work with a fixed set of states, while the size on n is unknown. Hence, we must pass from natural numbers to a representation for them on a finitary, e.g. binary, alphabet. In general, we shall associate to a pair $\langle\langle i, c \rangle, \langle j, action \rangle\rangle$ a tuple with the following syntactical structure

$$|w_i x, w_j y, z$$

where

1. " | " and " , " are special characters used as delimiters;
2. w_i and w_j are list of booleans representing the states i and j ;
3. x is special symbol *null* if $c = None$ and is the boolean a if $c = Some\ a$
4. y and z are both *null* if $action = None$, and are respectively equal to b and m' if $action = Some(b, m)$
5. finally, $m' = 0$ if $m = L$, $m' = 1$ if $m = R$ and $m' = null$ if $m = N$

As a minor, additional complication, we shall suppose that every character is decorated by an additional bit, normally set to false, to be used as a marker.

definition `mk_tuple := λqin,cin,qout,cout,mv.
 ⟨bar,false⟩ :: qin @ cin :: ⟨comma,false⟩ :: qout @ cout :: ⟨comma,false⟩ :: [mv].`

The actual encoding of states is not very important, and we shall skip it: the only relevant points are that (a) it is convenient to assume that all states (and hence all tuples for a given machine) have a fixed, uniform length; (b) the first bit of the representation of the state tells us if the state is final or not.

5.2 The Table of Tuples

The list of all tuples, concatenated together, provides the low level description of the normal Turing Machine to be interpreted by the Universal Machine: we call it a *table*.

The main lemma relating a table to the corresponding transition function is the following one, stating that for a pair $\langle s, t \rangle$ belonging to the graph of *trans*, and supposing that l is its encoding, then l occurs as a sublist (can be matched) inside the table associated with *trans*.

lemma `trans_to_match:`
`∀n.∀h.∀trans: trans_source n → trans_target n.
 ∀inp,outp,qin,cin,qout,cout,mv. trans inp = outp →
 tuple_encoding n h ⟨inp,outp⟩ = mk_tuple qin cin qout cout mv →
 match_in_table (S n) qin cin qout cout mv
 (flatten ? (tuples_list n h (graph_enum ?? trans))).`

5.3 The Use of Marks

We shall use a special alphabet where every character can be marked with an additional boolean. Marks are typically used in pairs and are meant to identify (and recall) a source and a target position where some joint operation must be performed: typically, a comparison or a copy between strings. The main generic operations involving marks are the following:

mark mark the current cell

clear_mark clear the mark (if any) from the current cell

adv_mark_r shift the mark one position to the right

adv_mark_l shift the mark one position to the left

adv_both_marks shift the marks at the right and left of the head one position to the right

match_and_advance f if the current character satisfies the boolean test f then advance both marks and otherwise remove them

adv_to_mark_r move the head to the next mark on the right

adv_to_mark_l move the head to the next mark on the left.

5.4 String Comparison

Apart from markings, there is an additional small problem in comparing and copying strings. The natural idea would be to store the character to be compared/copied into a register (i.e. as part of the state); unfortunately, our semantics is not state-aware. The alternative solution we have exploited is to have a family of machines, each specialized on a given character. So, comparing a character will consist of testing a character and calling the suitable machine in charge of checking/writing that particular character at the target position. This behavior is summarized in the following functions. The *comp_step_subcase* takes as input a character c , and a continuation machine *elseM* and compares the current character with c ; if the test succeeds it moves to the next mark to the right, repeats the comparison, and if successful advances both marks; if the current character is not c , it passes the control to *elseM*.

definition `comp_step_subcase := λalpha,c,elseM.
 ifFM ? (test_char ? (λx.x == c))
 (move_r ... adv_to_mark_r ? (is_marked alpha) · match_and_adv ? (λx.x == c))
 elseM tc_true.`

A step of the *compare* machine consists in using the previous function to build a chain of specialized testing functions on all characters we are interested in (in this case, *true*, *false*, or *null*), each one passing control to the next one in cascade:

```

definition comp_step :=
  ifTM ? (test_char ? (is_marked ?))
  (single_finalTM ... (comp_step_subcase FSUalpha ⟨bit false,true⟩
    (comp_step_subcase FSUalpha ⟨bit true,true⟩
      (comp_step_subcase FSUalpha ⟨null,true⟩
        (clear_mark ...))))))
  (nop ?)
  tc_true.

```

String comparison is then simply a while over *comp_step*

```

definition compare :=
  whileTM ? comp_step (inr ... (inl ... (inr ... start_nop))).

```

6 The Universal Machine

Working with a single tape, the most efficient way to simulate a given machine M is by keeping its code always close to the head of the tape, in such a way that the cost of fetching the next move is independent of the current size of the tape and only bounded by the dimension of M . The drawback is that simulating a tape move can require to shift the whole code of M ; assuming however that this is fixed, we have no further complexity slow-down in the interpretation. The Universal Machine is hence *fair* in the sense of [3].

Our universal machine will work with an alphabet comprising booleans and four additional symbols: “*null*”, “#” (grid), “|” (bar) and “,” (comma). In addition, in order to compare cells and to move their content around, it is convenient to assume the possibility of marking individual cells: so our tape symbols will actually be pairs of an alphabet symbol and a boolean mark (usually set to false).

The universal machine must be ready to simulate machines with an arbitrary number of states. This means that the current state of the simulated machine cannot be kept in a register (state) of the universal machine, but must be memorized on the tape. We keep it together with the current symbol of the simulated tape

The general structure of the tape is the following:

$$\alpha \# \overset{\downarrow}{q_{i0}} \dots q_{in} c \# table \# \beta$$

where α, β and c are respectively the left tape, right tape, and current character of the simulated machine. If there is no current character (i.e. the tape is empty or we are in a left or right overflow) then c is the special “*null*” character. The string $w_i = q_{i0} \dots q_{in}$ is the encoding of the current state q_i of M , and *table* is the set of tuples encoding the transition function of M , according to the definition of the previous section. In a well formed configuration we always have three occurrences of #: a leftmost, a middle and rightmost one; they are basic milestones to help the machine locating the information on the tape. At each iteration of a single step

of M the universal machine will start with its head (depicted with \Downarrow in the above representation) on the first symbol q_{i0} of the state.

Each step is simulated by performing two basic operations: fetching in the table a tuple matching $w_i c$ (*match_tuple*), and executing the corresponding action (*exec_action*). The *exec_action* function is also responsible for updating $w_i c$ according to the new state-symbol pair $w_j d$ provided by the matched tuple.

If matching succeeds, *match_tuple* is supposed to terminate in the following configuration, with the head on the middle $\#$

$$\alpha \# w_i c \# \underbrace{\dots | w_i c^* ; w_j d, m | \dots}_{table} \# \beta$$

where moreover the comma preceding the action to be executed will be marked (marking will be depicted with a $*$ on top of the character). If matching fails, the head will be on the $\#$ at the end of table (marked to discriminate easily this case from the former one):

$$\alpha \# w_i c \# \underbrace{table}_{*} \# \beta$$

The body of the universal machine is hence the following *uni_step* function, where *tc_true* is the accepting state of the *test_char* machine (in the next section we shall dwell into the details of the *match_tuple* and *exec_action* functions).

```

definition uni_step :=
  ifTM ? (test_char STape (λc.\fst c == bit false))
    (single_finalTM ?
      (init_match · match_tuple ·
        (ifTM ? (test_char ? (λc.¬ is_marked ? c))
          (exec_action · move_r ...))
        (nop ?) tc_true)))
    (nop ?) tc_true.
    
```

At the end of *exec_action* we must perform a small step to the right to reenter the expected initial configuration of *uni_step*.

The universal machine is simply a while over *uni_step*:

```

definition universalTM := whileTM ? uni_step us.acc.
    
```

The main semantic properties of *uni_step* and *universalTM* will be discussed in Section [9](#).

7 Matching

Comparing strings on a single tape machine requires moving back and forth between the two strings, suitably marking the corresponding positions on the tape. The following *initialize_match* function initializes marks, adding a mark at the

beginning of the source string (the character following the leftmost #, where the current *state, character* pair begins), and another one at the beginning of the table (the character following the middle #):

```
definition init_match :=
  mark ? · adv_to_mark_r ? (λc:STape.is_grid (\fst c)) · move_r ? ·
  move_r ? · mark ? · move_l ? · adv_to_mark_l ? (is_marked ?).
```

The *match_tuple* machine scrolls through the tuples in the transition table until one matching the source string is found. It just repeats, in a while loop, the operation of trying to match a single tuple discussed in the next section:

```
definition match_tuple :=
  whileTM ? match_tuple_step (inr ... (inl ... (inr ... start_nop))).
```

7.1 Match_Tuple_Step

The *match_tuple_step* starts checking the halting condition, that is when we have reached a (rightmost) #. If this is not the case, we execute the “then” branch, where we compare the two strings starting from the marked characters. If the two strings are equal, we mark the comma following the matched string in the table and then we stop on the middle #; otherwise, we mark the next tuple (if any) and reinitialize the mark at the beginning of the current state-character pair. If there is no next tuple, we stop on the rightmost grid after marking it.

If on the contrary the *match_tuple_step* is executed when the current character is a #, we execute the “else” branch, which does nothing.

```
definition match_tuple_step :=
  ifTM ? (test_char ? (λc:STape.¬ is_grid (\fst c)))
  (single_finalTM ?
    (compare ·
      (ifTM ? (test_char ? (λc:STape.is_grid (\fst c)))
        (nop ?)
        (mark_next_tuple ·
          (ifTM ? (test_char ? (λc:STape.is_grid (\fst c)))
            (mark ?) (move_l ? · init_current) tc_true)) tc_true)))
    (nop ?) tc_true.
```

The *match_tuple_step* is iterated until we end up in the “else” branch, meaning the head is reading a #. The calling machine can distinguish whether we ended up in a failure or success state depending on whether the # is marked or not.

8 Action Execution

Executing an action can be decomposed in two simpler operations, which can be executed sequentially: updating the current state and the character under the (simulated) tape head (*copy*), and moving the (simulated) tape (*move_tape*).

Similarly to matching, copying is done one character at a time, and requires a suitable marking of the tape (and a suitable initialization *init_copy*). As we shall see, the *copy* machine will end up clearing all marks, halting with the head on the comma preceding the move. Since *tape_move* expects to start with the head on the move, we must move the head one step to the right before calling it.

definition `exec_action` :=
`init_copy · copy · move_r ... · move_tape.`

8.1 *Init_Copy*

The *init_copy* machine initializes the tape marking the positions corresponding to the the cell to be copied and its destination (with the head ending up on the former). In our case, the destination is the position on the right of the leftmost #, while the source is the action following the comma in the tuple that has been matched in the table (that is the position to the right of the currently marked cell). In graphical terms, the *init_copy* machine transforms a tape of the form

$$\alpha \# q_{i0} \dots q_{in} c \# \dots \underbrace{|w_k a^*, q_{j0} \dots q_{jn} d, m|}_{table} \dots \# \beta$$

into

$$\alpha \# \overset{*}{q_{i0}} \dots q_{in} c \# \dots \underbrace{|w_k a, \overset{*}{q_{j0}} \dots q_{jn} d, m|}_{table} \dots \# \beta$$

This is the corresponding code:

definition `init_copy` :=
`init_current_on_match · move_r ? ·`
`adv_to_mark_r ? (is_marked ?) · adv_mark_r ?.`

where

definition `init_current_on_match` :=
`move_l ? · adv_to_mark_l ? (\lambda c:STape.is_grid (\fst c)) · move_r ? · mark ?.`

8.2 *Copy*

The copy machine copies the portion of the tape starting on the left mark and ending with a comma to a portion of the tape of the same length starting on the right mark. The machine is implemented as a while machine whose body copies one bit at a time, and advances the marks. In our case, this will allow us to pass from a configuration of the kind

$$\alpha \# \overset{*}{q_{i0}} \dots q_{in} c \# \dots \underbrace{|w_k a, \overset{*}{q_{j0}} \dots q_{jn} d, m|}_{table} \dots \# \beta$$

to a configuration like

$$\alpha \# q_{j_0} \dots q_{j_n} d \# \dots \underbrace{|w_k a, q_{j_0} \dots q_{j_n} d \downarrow m|}_{table} \dots \# \beta$$

As a special case, d can be a *null* rather than a bit: this identifies those actions that do not write a character to the tape. The *copy* machine acts accordingly, ignoring *nulls* and leaving c untouched. The copy machine removes all marks before exiting.

8.3 *Move_Tape*

Finally, the *move_tape* machine mimics the move action on the simulated tape. This is a complex operation, since we must skip the code of the simulated machine and its state. The main function just tests the character encoding the move action and calls three more elementary functions: *move_tape_r*, *move_tape_l*, and *no_move*:

definition *move_tape* :=
 ifTM ? (test_char ? (λc :STape.c == <bit false, false>))
 (adv_to_mark_r ? (λc :STape.is_grid (\fst c)) · move_tape_l)
 (ifTM ? (test_char ? (λc :STape.c == <bit true, false>))
 (adv_to_mark_r ? (λc :STape.is_grid (\fst c)) · move_tape_r)
 (no_move ?) tc_true) tc_true.

The *no_move* machine is pretty simple since it is merely responsible for resetting the head of tape at the expected output position, that is on the leftmost #:

definition *no_move* :=
 adv_to_mark_l ? (λc :STape.is_grid (\fst c)) ·
 move_l ... · adv_to_mark_l ? (λc :STape.is_grid (\fst c))

The other two functions are pretty similar; we shall only discuss the first one.

8.4 *Move_Tape_r*

The move tape right is conceptually composed of three sub-functions, executed sequentially: a *fetch_r* function, that advances the head to the first character of the simulated right tape (that is, the first character after the rightmost #), and initializes it to *null* if the tape is empty; a *set_new_current_r* function that moves it to the “current” position, that is at the position at the left of the middle #; and finally a *move_old_current_r*, that moves the old “current” value (which is now just at the left of the tape head), as first symbol of the left tape (that is, just after the the leftmost #). The last two functions are in fact very similar: they have just to move a character after the first # at their left (*move_after_left_grid*).

This is the evolution of the tape, supposing the right tape is not empty:

$$\begin{array}{ll}
 \alpha\#w_jd\#table\#b\beta & \text{fetch}_r \\
 \downarrow & \\
 \alpha\#w_jd\#table\#b\# \beta & \text{move_after_left_grid} \\
 \downarrow & \\
 \alpha\#w_jd\#b\#table\# \beta & \text{move}_l \\
 \downarrow & \\
 \alpha\#w_j\#d\#b\#table\# \beta & \text{move_after_left_grid} \\
 \downarrow & \\
 \alpha\#d\#w_jb\#table\# \beta & \text{move}_r \\
 \downarrow & \\
 \alpha d\#w_jb\#table\# \beta &
 \end{array}$$

This is the code for the above machines:

```

definition fetch_r :=
  move_r ... · init_cell · move_l ... · swap_r STape ⟨grid,false⟩.

definition move_after_left_grid :=
  move_l ... · move_char_l STape ⟨grid,false⟩ · swap_r STape ⟨grid,false⟩.

definition move_tape_r :=
  fetch_r · move_after_left_grid · move_l ... · move_after_left_grid · move_r ...
    
```

init_cell is an atomic machine defined in the obvious way.

9 Main Results

Given a configuration for a normal machine M , the following function builds the corresponding “low level” representation, that is the actual tape manipulated by the Universal Machine:

```

definition low_config: ∀M:normalTM.nconfig (no_states M) → tape STape :=
  λM:normalTM.λc.
  let n :=no_states M in
  let h :=nhalt M in
  let t :=ntrans M in
  let q :=cstate ... c in
  let q_low := m_bits_of_state n h q in
  let current_low :=
    match current ... (ctape ... c) with
    [ None ⇒ null | Some b ⇒ bit b] in
  let low_left :=map ... (λb.⟨bit b,false⟩) (left ... (ctape ... c)) in
  let low_right :=map ... (λb.⟨bit b,false⟩) (right ... (ctape ... c)) in
  let table :=flatten ? ( tuples_list n h (graph_enum ?? t)) in
  let right :=
    q_low@⟨current_low,false⟩ :: ⟨grid, false⟩ :: table@⟨grid, false⟩ :: low_right in
  mk_tape STape ((grid,false):: low_left) (option_hd ... right) (tail ... right).
    
```

Similarly, every relation over tapes can be reflected into a corresponding relation on their low-level representations:

definition $\text{low_R} := \lambda M, qstart, R, t1, t2.$
 $\forall \text{tape1}. t1 = \text{low_config } M (\text{mk_config } ?? \text{ } qstart \text{ tape1}) \rightarrow$
 $\exists q, \text{tape2}. R \text{ tape1 } \text{tape2} \wedge$
 $\text{halt } ? M \text{ } q = \text{true} \wedge t2 = \text{low_config } M (\text{mk_config } ?? \text{ } q \text{ tape2}).$

We expect the Universal Machine to be able to simulate on its tape each step of the machine M , and to stop leaving the tape unchanged when M stops. The machine must be able to end up in a special accepting state us_acc in the former case, and in a different state in the latter. The input-output relation realized by the machine in the two cases are the following:

definition $\text{low_step_R_true} := \lambda t1, t2.$
 $\forall M: \text{normalTM}. \forall c: \text{nconfig } (\text{no_states } M).$
 $t1 = \text{low_config } M \text{ } c \rightarrow$
 $\text{halt } ? M (\text{cstate } \dots c) = \text{false} \wedge t2 = \text{low_config } M (\text{step } ? M \text{ } c).$

definition $\text{low_step_R_false} := \lambda t1, t2.$
 $\forall M: \text{normalTM}.$
 $\forall c: \text{nconfig } (\text{no_states } M).$
 $t1 = \text{low_config } M \text{ } c \rightarrow \text{halt } ? M (\text{cstate } \dots c) = \text{true} \wedge t1 = t2.$

lemma $\text{sem_uni_step1}:$
 $\text{uni_step} \models [\text{us_acc}: \text{low_step_R_true}, \text{low_step_R_false}].$

For the universal machine we proved that, for any normal machine M , it weakly realizes the low level version of the canonical relation for M

theorem $\text{sem_universal}: \forall M: \text{normalTM}. \forall qstart.$
 $\text{universalTM} \models (\text{low_R } M \text{ } qstart (\text{R_TM } \text{FinBool } M \text{ } qstart)).$

From this result it is easy to derive that, for any relation weakly realized by M , the universal machine weakly realizes its low level counterpart.

theorem $\text{sem_universal2}: \forall M: \text{normalTM}. \forall R.$
 $M \models R \rightarrow \text{universalTM} \models (\text{low_R } M (\text{start } ? M) R).$

Termination is stated by the following result, whose proof is still in progress.

theorem $\text{terminate_UTM}: \forall M: \text{normalTM}. \forall t.$
 $M \downarrow t \rightarrow \text{universalTM} \downarrow (\text{low_config } M (\text{mk_config } ?? (\text{start } ? M) t)).$

10 Conclusions

We provided in this paper some preliminary results about formal specification and verification of Turing Machines, up to the definition of a universal machine and the proof of its correctness. The work is organized in 15 files (see Figure [1](#)), for a total of 6743 lines (comprising comments). It has been developed by the two authors during 2.5 months of intense joint work, at the good rate of more than

name	dimension	content
mono.ma	475 lines	mono-tape Turing machines
if_machine.ma	335 lines	conditional composition
while_machine	166 lines	while composition
basic_machines.ma	282 lines	basic atomic machines
move_char.ma	310 lines	character copying
alphabet.ma	110 lines	alphabet of the universal machine
marks.ma	901 lines	operations exploiting marks
compare.ma	506 lines	string comparison
copy.ma	579 lines	string copy
normalTM.ma	319 lines	normal Turing machines
tuples.ma	276 lines	normal Turing machines
match_machines.ma	727 lines	machines implementing matching
move_tape.ma	778 lines	machines for moving the simulated tape
uni_step.ma	585 lines	emulation of a high-level step
universal.ma	394 lines	the universal machine
total	6743 lines	

Fig. 1. List of files and their dimension in lines

300 lines per man-week (see [4] for an estimation of the cost of formalization at the current state of the art).

One could possibly wonder what is the actual purpose for performing a similar effort, but the real question is in fact the opposite one: what could be the reason for *not doing* it, since it requires a relatively modest investment in time and resources? The added value of having a complete, executable, and automatically verifiable specification is clear, and it could certainly help to improve confidence (of students, if not of researchers) in a delicate topic that, especially in modern textbooks, is handled in a very superficial way.

The development presented in this paper is still very preliminary, under many respects. In particular, the fact that the universal machine operates with a different alphabet with respect to the machines it simulates is annoying. Of course, any machine can be turned into a normal Turing machine, but this transformation may require a recoding of the alphabet that is not entirely transparent to complexity issues: for example, prefixing every character in a string $x_1 \dots x_n$ with a 0 in order to get the new string $0x_1 \dots 0x_n$ could take, on a single tape Turing Machine, a time quadratic in the length n of the string (this is precisely the kind of problems that raises a legitimate suspicion on the actual complexity of a *true* interpreter).

Complexity Theory, more than Computability, is indeed the real, final target of our research. Any modern textbook in Complexity Theory (see e.g. [2]) starts with introducing Turing Machines just to claim, immediately after, that the computational model *does not matter*. The natural question we are addressing and that we hope to contribute to clarify is: *what matters?*

The way we plan to attack the problem is by *reversing* the usual deductive practice of deriving theorems from axioms, reconstructing from proofs the basic assumptions underlying the major notions and results of Complexity Theory. The

final goal of our Reverse Complexity Program is to obtain a formal, axiomatic treatment of Complexity Theory at a *comfortable* level of abstraction, providing in particular logical characterizations of Complexity Classes that could help to better grasp their essence, identify their distinctive properties, suggest new, possibly non-standard computational models and finally provide new tools for *separating* them.

The axiomatization must obviously be validated with respect to traditional cost models, and in particular w.r.t. Turing Machines that still provide the actual foundation for this discipline. Hence, in conjunction with the “reverse” approach, it is also important to promote a more traditional forward approach, deriving out of concrete models the key ingredients for the study of their complexity aspects. The work in this paper, is meant to be a contribution along this second line of research.

References

1. Amadio, R., Asperti, A., Ayache, N., Campbell, B., Mulligan, D., Pollack, R., Régis-Gianas, Y., Coen, C.S., Stark, I.: Certified complexity. *Procedia CS* 7, 175–177 (2011)
2. Arora, S., Barak, B.: *Computational Complexity: A Modern Approach*. Cambridge Univ. Press (2009)
3. Asperti, A.: The intensional content of rice’s theorem. In: *Proceedings of the 35th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, San Francisco, California, USA, January 7-12, pp. 113–119. ACM (2008)
4. Asperti, A., Sacerdoti Coen, C.: Some Considerations on the Usability of Interactive Provers. In: Autexier, S., Calmet, J., Delahaye, D., Ion, P.D.F., Rideau, L., Rioboo, R., Sexton, A.P. (eds.) *AISC 2010*. LNCS, vol. 6167, pp. 147–156. Springer, Heidelberg (2010)
5. Asperti, A., Avigad, J. (eds.): Special issue on interactive theorem proving and the formalisation of mathematics. *Mathematical Structures in Computer Science*, vol. 21(4) (2011)
6. Asperti, A., Ricciotti, W.: A Web Interface for Matita. In: Jeuring, J. (ed.) *CICM 2012*. LNCS, vol. 7362, pp. 417–421. Springer, Heidelberg (2012)
7. Asperti, A., Ricciotti, W., Sacerdoti Coen, C., Tassi, E.: Hints in Unification. In: Berghofer, S., Nipkow, T., Urban, C., Wenzel, M. (eds.) *TPHOLs 2009*. LNCS, vol. 5674, pp. 84–98. Springer, Heidelberg (2009)
8. Asperti, A., Ricciotti, W., Sacerdoti Coen, C., Tassi, E.: The Matita Interactive Theorem Prover. In: Bjørner, N., Sofronie-Stokkermans, V. (eds.) *CADE 2011*. LNCS, vol. 6803, pp. 64–69. Springer, Heidelberg (2011)
9. Avigad, J., Donnelly, K., Gray, D., Raff, P.: A formally verified proof of the prime number theorem. *ACM Trans. Comput. Log.* 9(1) (2007)
10. Aydemir, B.E., Charguéraud, A., Pierce, B.C., Pollack, R., Weirich, S.: Engineering formal metatheory. In: *Proceedings of the 35th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2008*, San Francisco, California, USA, pp. 3–15. ACM (2008)
11. Davis, M.: *Computability and Unsolvability*. Dover Publications (1985)
12. Stearns, R.E., Hennie, F.C.: Two-tape simulation of multi tape turing machines. *Journal of ACM* 13(4), 533–546 (1966)

13. Fischer, P.C.: On formalisms for turing machines. *J. ACM* 12(4), 570–580 (1965)
14. Hales, T., Gonthier, G., Harrison, J., Wiedijk, F.: A Special Issue on Formal Proof. *Notices of the American Mathematical Society* 55 (2008)
15. Hartmanis, J., Stearns, R.E.: On the computational complexity of algorithms. *Transaction of the American Mathematical Society* 117, 285–306 (1965)
16. Hennie, F.C.: One-tape, off-line turing machine computations. *Information and Control* 8(6), 553–578 (1965)
17. Hopcroft, J.E., Ullman, J.D.: *Introduction to automata theory, languages, and computation*. Addison-Wesley (1979)
18. Klein, G.: Operating system verification – an overview. *Sadhana* 34(1), 27–69 (2009)
19. Leroy, X.: Formal certification of a compiler back-end or: programming a compiler with a proof assistant. In: *Proc. of the 33rd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2006, Charleston, South Carolina, USA*, pp. 42–54 (2006)
20. Norrish, M.: Mechanised Computability Theory. In: van Eekelen, M., Geuvers, H., Schmaltz, J., Wiedijk, F. (eds.) *ITP 2011. LNCS*, vol. 6898, pp. 297–311. Springer, Heidelberg (2011)
21. Sipser, M.: *Introduction to the Theory of Computation*. PWS (1996)
22. Turing, A.M.: On computable numbers, with an application to the entscheidungsproblem. *Proc. of the London Math. Society* 2(42), 230–265 (1936)
23. van Emde Boas, P.: Machine models and simulation. In: *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity (A)*, pp. 1–66 (1990)
24. Weirich, S., Pierce, B. (eds.): Special issue on the poplmark challenge. *Journal of Automated Reasoning* (2011) (published online)
25. Yang, J., Hawblitzel, C.: Safe to the last instruction: automated verification of a type-safe operating system. *Commun. ACM* 54(12), 123–131 (2011)

Equivalence Relations That Are Σ_3^0 Complete for Computable Reducibility^{*}

(Extended Abstract)

Ekaterina Fokina¹, Sy Friedman¹, and André Nies²

¹ Kurt Gödel Research Center, University of Vienna, Austria
{efokina,sdf}@logic.univie.ac.at

² Department of Computer Science, University of Auckland, Auckland, New Zealand
andre@cs.auckland.ac.nz

Abstract. Let E, F be equivalence relations on \mathbb{N} . We say that E is computably reducible to F , written $E \leq F$, if there is a computable function $p: \mathbb{N} \rightarrow \mathbb{N}$ such that $xEy \leftrightarrow p(x)Fp(y)$. We show that several natural Σ_3^0 equivalence relations are in fact Σ_3^0 complete for this reducibility. Firstly, we show that one-one equivalence of computably enumerable sets, as an equivalence relation on indices, is Σ_3^0 complete. Thereafter, we show that this equivalence relation is below the computable isomorphism relation on computable structures from classes including predecessor trees, Boolean algebras, and metric spaces. This establishes the Σ_3^0 completeness of these isomorphism relations.

1 Introduction

Invariant descriptive set theory studies the complexity of equivalence relations on the reals via Borel reductions (see [6]). An analog for equivalence relations on natural numbers, where the reductions are computable functions, was already introduced in [1], and has received considerable attention in recent years [7,3].

The isomorphism relation on a class of structures is a natural example of an equivalence relation. A countable structure in a countable signature can be encoded by a real. The complexity of the isomorphism relation on (reals encoding) countable structures has been studied in invariant descriptive set theory beginning with H. Friedman and Stanley [5]. For instance, they showed that isomorphism of countable graphs is not Borel complete for analytic equivalence relations.

We may assume that the domain of a countable structure is an initial segment of \mathbb{N} . Then the quantifier free statements involving elements of the structure can also be encoded by natural numbers. Suppose the signature is computable. We say that a presentation of a countable structure is *computable* if its atomic

^{*} The first and the second authors acknowledge the generous support of the FWF through projects Elise-Richter V206, and P22430-N13. The third author is partially supported by the Marsden Fund of New Zealand under grant 09-UOA-187.

diagram, that is, all the quantifier free facts about the structure, is a computable set. A computable index for the atomic diagram is also called a *computable index* for the structure. As a general rule, familiar countable structures all have computable presentations. Examples include $(\mathbb{Z}, +)$ and $(\mathbb{Q}, <)$.

Following Fokina et al. [4], for a class \mathcal{K} of structures, we denote by $I(\mathcal{K})$ the set of computable indices for structures in \mathcal{K} . For common classes, this will be an arithmetical set. Isomorphism can now be viewed as an equivalence relation on $I(\mathcal{K})$, and clearly is Σ_1^1 . Fokina et al. [4] studied possible analogs of some results in [5] for isomorphism on computable structures. Their reduction, denoted \leq_{FF} , was a slight extension of computable reducibility which allows for partial computable functions as reductions as long as their domain contains the relevant set $I(\mathcal{K})$. In contrast to the above-mentioned result of [5], they proved as a main result that isomorphism of computable graphs is \leq_{FF} complete for Σ_1^1 equivalence relations. Coding graphs into structures, they then obtained the similar result for other classes, such as torsion free abelian groups, and linear orders. Boolean algebras were notably absent.

In this paper, we go one step further in effectivizing the setting of [5]: we also require that the isomorphisms are computable. For computable presentations C, D of structures in the same computable signature, we write

$$C \cong_{comp} D$$

if there is a partial computable bijection between the domains of C, D (initial segments of \mathbb{N}) which induces an isomorphism of the structures. Clearly, if $I(\mathcal{K})$ is Σ_3^0 , then computable isomorphism on $I(\mathcal{K})$ is also Σ_3^0 .

We will show that for several classes of structures, the computable isomorphism relation is a Σ_3^0 -complete equivalence relation under computable reducibility: computable trees and graphs, computable Boolean algebras, and (with some adjustment of terminology) metric spaces. Note that for some classes, however, the computable isomorphism problem may be less complex than Σ_3^0 . For instance, consider the class \mathcal{K} of computable permutations of order 2. Then $I(\mathcal{K})$ is Π_2^0 . The computable isomorphism relation on $I(\mathcal{K})$ is also Π_2^0 . This is so because we only need to figure out whether for two given permutations, both have the same number of 1-cycles, and the same number of 2-cycles.

Our completeness results rely on a recursion theoretic fact of interest by itself. As usual let $(W_e)_{e \in \mathbb{N}}$ be an effective listing of the computably enumerable sets. Recall that sets $A, B \subseteq \mathbb{N}$ are *1-equivalent*, $A \equiv_1 B$, if there is a computable permutation h of \mathbb{N} such that $h(A) = B$.

Theorem 1. *For each Σ_3^0 equivalence relation S , there is a computable function g such that*

$$\begin{aligned} ySz &\Rightarrow W_{g(y)} \equiv_1 W_{g(z)}, \text{ and} \\ \neg ySz &\Rightarrow W_{g(y)}, W_{g(z)} \text{ are Turing incomparable.} \end{aligned}$$

The proof will be given in Section 3. As an immediate consequence, we have:

Corollary 2. *Many-one equivalence and 1-equivalence on indices of c.e. sets are Σ_3^0 complete for equivalence relations under computable reducibility.*

Note that this is significantly stronger than the mere Σ_3^0 completeness of \equiv_m as a set of pairs of c.e. indices, which follows for instance because the m -complete c.e. set have a Σ_3^0 complete index set.

As a further consequence, Turing equivalence on indices of c.e. sets is a Σ_3^0 hard equivalence relation for computable reducibility. However, this equivalence relation is only Σ_4^0 . We conjecture that it is in fact Σ_4^0 complete in our sense.

In the following Section 2, we will encode 1-equivalence on indices of c.e. sets into computable isomorphism for the relevant classes. We then use Corollary 2 to conclude these isomorphism relations are Σ_3^0 complete.

2 Computable Isomorphism of Computable Structures

2.1 Computable Trees and Computable Equivalence Relations

We use the terminology of Fokina, Friedman et al. [4]. In particular, a tree is a structure in the language containing the predecessor function as a single unary function symbol. The root is its own predecessor. A countable tree can be represented by a nonempty subset B of $\omega^{<\omega}$ closed under prefixes. The unary predecessor function takes off the last entry of a non-empty tuple of natural numbers, and maps the empty tuple to itself.

A tree has a computable presentation iff we can choose B c.e. For in that case B is the range of a partial computable 1-1 function ϕ with domain an initial segment of ω ; the preimage of the predecessor function under ϕ is the required computable atomic diagram.

We let

$$T_e = \{\sigma : \exists \tau \succeq \sigma [\tau \in W_e]\},$$

where the e -th c.e. set W_e is now viewed as a subset of $\omega^{<\omega}$. Then $(T_e)_{e \in \mathbb{N}}$ is a uniform listing of all computable trees.

We say a tree has height k if every leaf has length at most k .

Proposition 3. *Computable isomorphism of computable trees of height 2 where every node at level 1 has out-degree at most 1 is a complete Σ_3^0 equivalence relation.*

Proof. Let h be a computable function such for each e , $T_{h(e)}$ is the tree

$$\{\emptyset\} \cup \{\langle x \rangle : x \in \omega\} \cup \{\langle x, 0 \rangle : x \in W_e\}.$$

Clearly, $W_y \equiv_1 W_z$ iff $T_{h(y)}$ is computably isomorphic to $T_{h(z)}$. Now we apply Corollary 2

A similar argument shows:

Proposition 4. *Computable isomorphism of computable equivalence relations where every class has at most 2 members is a complete Σ_3^0 equivalence relation.*

2.2 Boolean Algebras

For a linear order L with least element, *Intalg* L denotes the subalgebra of the Boolean algebra $\mathcal{P}(L)$ generated by intervals $[a, b]$ of L where $a \in L$ and $b \in L \cup \{\infty\}$. Here ∞ is a new element greater than any element of L , and $[a, \infty)$ is short for $\{x \in L : x \geq a\}$. Note that *Intalg* L consists of all sets S of the form

$$S = \bigcup_{r=1}^n [a_r, b_r)$$

where $a_0 < b_0 < a_1 \dots < b_n \leq \infty$. From a computable presentation of L as a linear order, we may canonically obtain a computable presentation of the Boolean algebra *Intalg* L .

Theorem 5. *Computable isomorphism of computable Boolean algebras is complete for Σ_3^0 equivalence relations.*

Proof. Let $(V^e)_{e \in \mathbb{N}}$ be an effective listing of the c.e. sets containing the even numbers. The relation of 1-equivalence \equiv_1 of c.e. sets V^e is Σ_3^0 complete by Theorem [1](#) and its proof below. We will computably reduce it to computable isomorphism of computable Boolean algebras. We define the Boolean algebra C^e to be the interval algebra of a computable linear order L^e . Informally, to define L^e , we begin with the order type ω . For each $x \in \omega$, when x enters V^k we replace x by a computable copy of $[0, 1)_{\mathbb{Q}}$. More formally,

$$L^e = \bigoplus_{x \in \omega} M_x^e,$$

where M_x^e has one element $m_x^k = 2x$, until x enters V^e ; if and when that happens, we expand M_x^e to a computable copy of $[0, 1)_{\mathbb{Q}}$, using the odd numbers, while ensuring that $m_x^k = \min M_x^k$ holds in L^k . Also note that the domain of L^k is \mathbb{N} because $0 \in V^k$.

Claim. $V^e \equiv_1 V^i \Leftrightarrow C^e \cong_{\text{comp}} C^i$.

\Rightarrow : Suppose $V^e \equiv_1 V^i$ via a computable permutation π . We define a computable isomorphism $\Phi : C^e \cong C^i$.

(a) Let $\Phi(m_x^e) = m_{\pi(x)}^i$. Once x enters V^e , we know that $\pi(x) \in V^i$. So we may always ensure that Φ restricts to a computable isomorphism of linear orders $M_x^e \cong M_{\pi(x)}^i$.

(b) Consider an element S of C^e . It is given in the form $S = \bigcup_{r=1}^n [a_r, b_r)$ where $a_0 < b_0 < a_1 \dots < b_n$ for $a_r, b_r \in L^e \cup \{\infty\}$ as above. If $b_n < \infty$, we can compute the maximal $x \in \omega$ such that $M_x^e \cap S \neq \emptyset$. Define

$$\Phi(S) = \bigcup_{y \leq x} \Phi(S \cap M_y^e).$$

Note that the set $\Phi(S \cap M_y^e)$ can be determined by (a).

If $b_n = \infty$, then let $\Phi(S)$ be the complement in L^i of $\Phi(L^e \setminus S)$.

\Leftarrow : Now suppose that $C^e \cong_{comp} C^i$ via some computable isomorphism Φ . We show that $V^e \leq_1 V^i$ via some computable function f . Suppose we have defined $f(y)$ for $y < x$. We have $\Phi(M_x^e) = \bigcup_{r=1}^n [a_r, b_r]$ where $a_r, b_r \in L^i \cup \{\infty\}$ as above.

If $n > 1$ then M_x^e is not an atom in C^e , whence $x \in V^e$. Thus let $f(x)$ be the least even number that does not equal $f(y)$ for any $y < x$.

Now suppose $n = 1$. If $a_1 = m_y^i, b_1 = m_{y+1}^i$ then let $f(x) = y$. Otherwise, again we know M_x^e is not an atom in C^e , and define $f(x)$ as before.

By symmetry, we also have $V^i \leq_1 V^e$, and hence $V^i \equiv_1 V^e$ by Myhill's theorem.

2.3 Metric Spaces

Let (M, d) be a metric space, and let $(\alpha_i)_{i \in \mathbb{N}}$ be a dense sequence in M without repetitions. We say that $\mathcal{M} = (M, d, (\alpha_i)_{i \in \mathbb{N}})$ is a *computable metric space* if $d(\alpha_i, \alpha_k)$ is a computable real uniformly in i, k . We call the elements of the sequence $(\alpha_i)_{i \in \mathbb{N}}$ the *special points*. For background on computable metric spaces, see [2].

A computable metric space is *discrete* if every point is isolated. For such a space, necessarily every point is a special point.

Corollary 6. *Computable isometry of discrete computable metric spaces is complete for Σ_3^0 equivalence relations.*

Proof. Given a computable tree B , create a discrete computable metric space M_B as follows: if a string $\langle x \rangle$ enters B , add a point p_x . If later $\langle x, i \rangle$ enters B for the first i , add a further point q_x . Declare $d(p_x, q_x) = 1/4$. Declare $d(p_x, p_y) = 1$ and $d(q_x, p_y) = 1$ (if q_x exists). Clearly for trees B, C as in Cor. [3] B is computably isomorphic to C iff M_B is computably isometric to M_C .

3 Proof of Theorem [1]

Since S is Σ_3^0 , there is a uniformly c.e. triple sequence

$$(V_{y,z,i})_{y,z,i \in \omega, y < z}$$

of initial segments of \mathbb{N} such that for each $y < z$,

$$ySz \Leftrightarrow \exists i V_{y,z,i} = \omega.$$

We build a uniformly c.e. sequence of sets $A_x = W_{g(x)}$ ($x \in \omega$), g computable. We meet the following coding requirements for all $y < z$ and $i \in \omega$.

$$G_{y,z,i}: V_{y,z,i} = \omega \Rightarrow A_y \equiv_1 A_z.$$

We meet diagonalization requirements for $u \neq v$,

$$N_{u,v,e}: u = \min[u]_S \wedge v = \min[v]_S \Rightarrow A_u \neq \Phi_e(A_v).$$

where Φ_e is the e -th Turing functional, and $[x]_S$ denotes the S -equivalence class of x . Meeting these requirements suffices to establish the theorem.

The basic strategies to meet the requirements are as follows. If $V_{y,z,i} = \omega$, a strategy for $G_{y,z,i}$ “finds out” that z is S -related to the smaller y . Hence it builds a computable permutation h such that $A_y \equiv_1 A_z$ via h .

A strategy for $N_{u,v,e}$ picks a witness n , and waits for $\Phi_e(A_v; n)$ to converge. Thereafter, it ensures that this computation is stable and $A_u(n)$ does not equal its output $\Phi_e(A_v; n)$ by enumerating n into A_u if this output is 0.

The tree of strategies. To avoid conflicts between strategies that enumerate into the same set A_z , we need to provide the strategies with a guess at whether z is least in its S -equivalence class $[z]_S$. An N -type strategy will only enumerate into A_z if according to its guess, z is least in its $[z]_S$; a G -type strategy only enumerates into A_z if according to its guess, z is not least.

Fix an effective priority ordering of all requirements. We define a tree T of strategies, which is a computable subtree T of $2^{<\omega}$. We write $\alpha : R$ if strategy α is associated with the requirement R . By recursion on $|\alpha|$, we define whether $\alpha \in T$, and which is the requirement associated with α . We also define a function L mapping $\alpha \in T$ to a cofinite set $L(\alpha)$ consisting of the numbers x such that according to α 's guesses, x is least in its equivalence class.

Let $L(\emptyset) = \omega$. Assign to α the highest priority requirement R not yet assigned to a proper prefix of α such that either (a) or (b) hold.

- (a) R is $G_{y,z,i}$ and $z \in L(\alpha)$; in this case put both $\alpha 0$ and $\alpha 1$ on T , and define $L(\alpha 0) = L(\alpha) - \{z\}$ while $L(\alpha 1) = L(\alpha)$ (along $\alpha 0$ we know that x is no longer the least in its equivalence class)
- (b) R is $N_{u,v,e}$ and $u, v \in L(\alpha)$; in this case put only $\alpha 0$ on T , and define $L(\alpha 0) = L(\alpha)$.

For strings $\alpha, \beta \in 2^{<\omega}$, we write $\alpha <_L \beta$ if there is i such that $\alpha \upharpoonright_i = \beta \upharpoonright_i$, $\alpha(i) = 0$ and $\beta(i) = 1$. We let $\alpha \preceq \beta$ denote that α is a prefix of β . We define a linear ordering on strings by

$$\alpha \leq \beta \text{ if } \alpha <_L \beta \text{ or } \alpha \preceq \beta.$$

Construction of a u.c.e. sequence of sets $(A_x)_{x \in \mathbb{N}}$. We declare in advance that $A_x(4m+1) = 0$ and $A_x(4m+3) = 1$ for each x, m . The construction then only determines membership of even numbers in the A_x .

We define a computable sequence $(\delta_s)_{s \in \mathbb{N}}$ of strings on T of length s . Suppose inductively that δ_t has been defined for $t < s$. Suppose $k < s$ and that $\eta = \delta_s \upharpoonright_k$ has been defined. If $\eta : N_{u,v,e}$ let $\delta_s(k) = 0$. Otherwise $\eta : G_{y,z,i}$. Let $t < s$ be the largest stage such that $t = 0$ or $\eta \preceq \delta_t$. Let $\delta_s(k) = 0$ if $V_{y,z,i,s} \neq V_{y,z,i,t}$, and otherwise $\delta_s(k) = 1$.

The *true path* TP is the lexicographically leftmost path $f \in 2^\omega$ such that $\forall n \exists^\infty s \geq n [\delta_s \upharpoonright_n \prec f]$. To *initialize* a strategy α means to return it to its first instruction. If $\alpha : G_{y,z,i}$ we also make the partial computable function h_α built by the strategy α undefined on all inputs. At stage s , let $\text{init}(\alpha, s)$ denote the largest stage $\leq s$ at which α was initialized.

An $N_{u,v,e}$ strategy α . At stages s :

- (a) Appoint an unused even number $n > \text{init}(\alpha, s)$ as a witness for diagonalization. Initialize all the strategies $\beta \succ \alpha$.
- (b) Wait for $\Phi_e(A_v; n)[s]$ to converge with output r . If $r = 0$ then put n into A_u . Initialize all the strategies $\beta \succ \alpha$.

A $G_{y,z,i}$ strategy α . If $\alpha 0$ is on the true path then this strategy builds a computable increasing map h_α from even numbers to even numbers such that $A_y(k) = A_z(h_\alpha(k))$ for each k . Furthermore, $A_z - \text{range}(h_\alpha)$ is computable. By our definitions of A_y and A_z on the odd numbers, this implies that h_α can be extended to a computable permutation showing that $A_y \equiv_1 A_z$, as required.

At stages s , if $\alpha 0 \subseteq \delta_s$, let $t < s$ be greatest such that $t = 0$ or $\alpha 0 \subseteq \delta_t$, and do the following.

- (a) For each even $k < s$ such that $k \notin \text{dom}(h_{\alpha,t})$ pick an unused even value $m = h_{\alpha,s}(k) > \text{init}(\alpha, s)$ in such a way that h_α remains increasing.
- (b) From now on, unless α is initialized, ensure that $A_z(m) = A_y(k)$. (We will verify that this is possible.)

The stage-by-stage construction is as follows. At stage $s > 0$ initialize all strategies $\alpha \succ_L \delta_s$. Go through substages $i \leq s$. Let $\alpha = \delta_s \upharpoonright_i$. Carry out the strategy α at stage s .

Verification. To show the requirements are met, we first check that there is no conflict between different strategies that enumerate into the same set A_z .

Claim. Let $\alpha: G_{y,z,i}$. Then (b) in the strategy for α can be maintained as long as α is not initialized.

To prove the claim, suppose a strategy $\beta \neq \alpha$ also enumerates numbers into A_z . If $\alpha 0 \prec_L \beta$ then β is initialized when α extends its map h_α , so the numbers enumerated by β are not in the range of h_α . If $\beta \prec_L \alpha 0$ then α is initialized when β is active, so again the numbers enumerated by β are not in the range of h_α . Now suppose neither hypothesis holds, so $\alpha 0 \preceq \beta$ or $\beta \prec \alpha$.

Case $\beta: N_{z,v,e}$. In this case $\alpha 0 \preceq \beta$ is not possible because $z \notin L(\alpha 0)$. If $\beta \prec \alpha$ then α is initialized when β appoints a new diagonalization witness.

Case $\beta: G_{y',z,i'}$. In this case $\alpha 0 \preceq \beta$ is not possible because $z \notin L(\alpha 0)$. If $\beta 1 \preceq \alpha$ then α is initialized each time β extends its map h_β . Finally, $\beta 0 \preceq \alpha$ is not possible because $z \notin L(\beta 0)$. This proves the claim.

Claim. Let α be the $N_{u,v,e}$ strategy on the true path. Suppose α is not initialized after stage s . Then α only acts finitely often, and meets its requirement.

At some stage $\geq \text{init}(\alpha, s)$ the strategy α picks a permanent witness n . No strategy $\beta \prec \alpha$ can put n into A_u because $u \in L(\alpha)$. No other strategy can put n into A_u because of the initialization α carries out when it picks n . Suppose now that at a later stage t , a computation $\Phi_e(A_v; n)[t]$ converges. Since $v \in L(\alpha)$, no G -type strategy $\beta \prec \alpha$ enumerates into A_v . Thus the initialization of strategies $\gamma \succ \alpha$ carried out by α at that stage t will ensure that this computation is preserved with value different from $A_u(n)$. This proves the claim.

It is now clear by induction that each strategy α on the true path is initialized only finitely often. Thus the N -type requirements are met. Now suppose $\alpha: G_{y,z,i}$ and $\alpha 0$ is on the true path. Then no strategy $\beta \succeq \alpha 0$ enumerates into A_z . Thus by the initialization at stages s such that $\alpha 0 \preceq \delta_s$, the set $A_z - \text{range}(h_\alpha)$ is computable. As noted earlier, this implies that h_α can be extended to a computable permutation showing that $A_y \equiv_1 A_z$. There is a computable bijection q between the set of odd numbers and the set of numbers that are odd, or even but not in the range of h_α , so that $m \in A_y \leftrightarrow q(m) \in A_z$. Now let the permutation be $q \cup h_\alpha$.

References

1. Bernardi, C., Sorbi, A.: Classifying positive equivalence relations. *J. Symb. Log.* 48(3), 529–538 (1983)
2. Brattka, V., Hertling, P., Weihrauch, K.: A tutorial on computable analysis. In: Barry Cooper, S., Löwe, B., Sorbi, A. (eds.) *New Computational Paradigms: Changing Conceptions of What is Computable*, pp. 425–491. Springer, New York (2008)
3. Coskey, S., Hamkins, J.D., Miller, R.: The hierarchy of equivalence relations on the natural numbers under computable reducibility, pp. 1–36, <http://arxiv.org/abs/1109.3375> (submitted)
4. Fokina, E.B., Friedman, S.-D., Harizanov, V.S., Knight, J.F., McCoy, C.F.D., Montalbán, A.: Isomorphism relations on computable structures. *J. Symb. Log.* 77(1), 122–132 (2012)
5. Friedman, H., Stanley, L.: A Borel reducibility theory for classes of countable structures. *Journal of Symbolic Logic* 54, 894–914 (1989)
6. Gao, S.: *Invariant descriptive set theory*. Pure and Applied Mathematics (Boca Raton), vol. 293. CRC Press, Boca Raton (2009)
7. Gao, S., Gerdes, P.: Computably enumerable equivalence relations. *Studia Logica* 67(1), 27–59 (2001)

An Analysis of Directed Motion Expressions with Lexicalized Tree Adjoining Grammars and Frame Semantics^{*}

Laura Kallmeyer and Rainer Osswald

Department of Linguistics and Information Science
University of Düsseldorf, Germany

Abstract. We present an analysis of directed motion expressions in the framework of Lexicalized Tree Adjoining Grammars (LTAG) enriched with a decompositional frame semantics. This approach to the syntax-semantics interface allows us to combine a detailed decomposition and composition of syntactic building blocks with a parallel decomposition and composition of meaning components. In LTAG, lexical anchors can be distinguished from unanchored elementary trees which allows for the description of the meaning contributions of constructions. Furthermore, due to the metagrammatical factorization of the descriptions of unanchored elementary trees, the meaning contributions of single argument realizations and of their combinations can be described in a principle way.

1 Introduction

Investigating the interplay between the syntax and semantics of directed motion expressions, and of verb-based constructions in general, is faced with the following two issues, among others: the distinction between arguments and adjuncts and the syntactic mechanisms of semantic composition. In this paper, we show how these issues can be naturally addressed within a framework that integrates *Lexicalized Tree Adjoining Grammars* (LTAG) with *Frame Semantics*. Semantic frames have been established as an expressive way to capture detailed aspects of meaning. So far, they are mainly used to describe the meanings of single lexical items. This paper concentrates on frame-based semantic composition and its interaction with syntactic operations. There are two reasons for choosing LTAG in the context of semantic frame composition. Firstly, the elementary trees in LTAG represent entire subcategorization frames, which facilitates the linking of the syntactic components and the semantic frame components. Secondly, the underlying “metagrammatical” specification of an LTAG allows a strong factorization of the syntactic and semantic information of elementary trees and thereby enables one to capture the specific meaning contributions of fragments of constructions.

The focus of this paper is on directional expression that are constructed from verbs of motion and directional PPs. The relevant constructions include intransitive verbs of locomotion (1) as well as transitive verbs of caused motion and transport (2).

^{*} The research reported in this paper has been financially supported by the German Science Foundation (DFG) as part of the SFB 991.

- (1) a. Mary walked to the house.
b. The ball rolled into the goal.
- (2) a. John threw/kicked the ball into the goal.
b. John pushed/pulled the cart to the station.
c. John rolled the ball into the hole.

Directional specifications are not restricted to goal expressions as in (1) and (2) but can also describe the source or the course of the path in more detail. Moreover, path descriptions can be iterated to some extent (3). Below we will use this property as an indicator for distinguishing between arguments and adjuncts.

- (3) a. John walked through the gate along the fence to the house.
b. John threw the ball over the fence into the yard.

The paper is structured as follows. In Section 2, we discuss the semantics of directed motion expressions, formulated with frames. The following section introduces LTAG. Section 4 brings the previous sections together by proposing a frame-based LTAG semantics for expressions of directed motion, distinguishing between the meaning of lexical items and the meaning of unanchored elementary trees. Building on this, Section 5 develops a more fine-grained factorization of the syntax and semantics of the unanchored elementary trees, using the LTAG metagrammar.

2 Directed Motion Expressions

Modeling the syntax-semantics interface of directed motion expressions requires us to be explicit about a number of issues concerning the syntactic and semantic structure of such expressions, many of which have been discussed extensively in the literature.

2.1 Verbs of Motion

It is common to distinguish between manner-encoding and path-encoding verbs of motion. The first kind of verbs (*run, roll*) lexically encode the manner of the motion but no path-related information, while the second kind of verbs (*enter, leave*) do not encode the manner but specify the direction of motion. Manner-encoding motion verbs lexically characterize activities or processes. Directional information about the goal or path can be added by appropriate adverbials (i.e., by “satellite framing” constructions (21)). In the following, we focus on manner-encoding verbs since our goal is to model the syntactic and semantic processes of combining directional specifications with motion expressions.

There are also motion verbs where the actor differs from the entity that undergoes the motion. This class includes verbs of transport and caused motion (*carry, drag, push, throw*). As with manner-of-motion verbs, transport and caused motion verbs do not lexically specify a direction or goal. Again, directional information can be added by adverbials. The verbs of transport and caused motion are basically transitive verbs whose direct object refers to the moving entity. They can be sub-divided into different classes depending on (i) how the motion of the object is enforced by the actor and (ii) to which

extent the activity of the actor and the manner of motion are lexically specified (cf. [7]). Concerning (i), we can distinguish between *onset causation* (*throw, kick*) and *extended causation* (*pull, drag*), following the terminology of [20]. Verbs of the first type describe the punctual initiation of a motion event, verbs of the second type describe the continuous enforcement of the motion. As to (ii), some of the verbs in question specify the manner of motion of the moved object but say nothing about the activity of the actor (*roll, slide*), while for other verbs the converse is true (*pull, drag*)¹

In the following, we are concerned with locomotion/manner-of-motion verbs and caused motion and transport verbs that occur in directional constructions like those listed in (1)–(3).

2.2 Syntactic Issues

In the context of the LTAG analysis presented in the following sections, a crucial issue is whether to treat directional expression such as those in (1)–(3) as complements or as adjuncts. Moreover, an argument can be determined by the base lexeme or it can be introduced by a construction or a lexical rule. For instance, sentences of type (2-c) are often characterized as *caused motion constructions* or *causative path resultatives* [14]. That is, the directional argument is constructionally introduced. Within the LTAG approach both, the basic argument structure construction as well as the extended construction are represented by elementary trees. The relation between these trees, and the fact that one of them builds on the other, is captured in the class structure of the metagrammar (cf. Section 5).

Dowty [6] counts directional PPs as adjuncts of motion verbs since their presences is not obligatory and they do not “complete” but “modify” the meaning of the head verb. Dowty distinguishes adjuncts from elliptical complements by characterizing the latter as cases where a semantically required element must be inferred contextually. Van Valin & LaPolla [22] classify directional PPs as “argument-adjuncts”. Like adjuncts, argument-adjuncts are predicative, but they introduce an argument into the syntactic core of the head verb and they typically share an argument with the predicate encoded by the verb. A well-known distinction observed by Jackendoff, Verkuyl and Zwarts, among others, is that between *bounded* and *unbounded* directional PPs, which give rise respectively to telic (4-a) and atelic (4-b) event descriptions [15, 23, 25].

- (4) a. She walked to the brook (in half an hour/*for half an hour).
 b. She walked along the brook (*in half an hour/for half an hour).

With reference to this distinction and based on data from Dutch and other languages, Gehrke [13] argues that bounded directional PPs are complements of the verb while unbounded PPs are adjuncts. For verbs of locomotion and transport, which are lexically atelic, this means that a directional expression is regarded as a complement in case it changes the aspectual type of the expression. This assumption is compatible with the formal criterion that expressions that can be added iteratively (as, e.g., pronominal adjectives) need to be analyzed as adjuncts. In the following, we take this criterion as a preliminary working definition of adjuncthood.

¹ Cf. [7] for further distinctions.

2.3 Motion, Paths, and Directions

A locomotion event is by definition associated with some trajectory, trace or path of the moving entity. The approaches found in the literature differ with respect to the explicit representation of the path in the lexical semantics of the respective verbs. While in [5] and [18], paths are not part of the semantic representations of locomotion verbs, [25] proposes a thematic function TRACE that maps motion events to the path traversed by the moving entity and in [19], it is assumed that “manner-of-motion predicates leave a trail of the motion along an implicit path, as measured over time.” Similarly, [8] take paths as part of the semantics of verbs of motion. The paths referenced by verbs are here again understood as trajectories, that is, as the collection of “all points the object occupies during its course.”

Paths, traces or trajectories provide a straightforward semantic link between motion verbs and directional specifications. Directionals (in English) occur often morphologically combined with locatives. For example, the directional preposition *into* specifies a path whose end point is in the interior of the goal expressed by the nominal complement of the preposition. The interior region associated with an object, as well as other regions specified by locatives, can be regarded as functional attributes of that object. We will employ this view below for the frame representations of directional prepositions.

2.4 Frame-Semantic Representation

The semantics of directional expressions is often represented in terms of logical expression of one kind or another in the literature (cf. [8]). In our approach, we employ frames for semantic representation, inspired by the programmatic outlines in [9] and [2]. Frames in this context are to be understood as generalized typed feature structures with relational constraints. In contrast to the flat role frames used in FrameNet [10], we take into account semantic decomposition, which gives rise to nested frame structures. For example, the verb *throw* expresses a caused motion, that is, the described event can be analyzed as a complex causation event whose cause component consists of the activity of the thrower and whose effect is the ballistic motion of the thrown object. A possible frame-semantic representation of this decompositional analysis is shown on the right side of Fig. 1 which also shows frames for *walk* and *pull*.

In the given representations, a good part of the lexical meaning is condensed in the types or left implicit. For instance, the precise way of how the actor induces the (ballistic) motion of the object in throwing events is simply encoded by an atomic value of the attribute MANNER. Similarly, the causation type of throwing events is encoded by the type *onset-causation* of the main event. A more explicit representation would include the temporal characteristics of an onset causation, i.e., punctuality and temporal precedence of the causing event. Notice that the path or trace of the moving entity is made explicit by the frames in Fig. 1. As argued above, the trace of the moving object is an inherent semantic component of locomotion events; the path provides the anchor for directional specifications. It is important to keep in mind that the presence of the PATH attribute in the frame representation of, say, *walk* does not imply by any means that *walk* lexically encodes any information about the path of the movement.

For the frame representations of directional prepositions, we follow basically the outline discussed in the previous section. The basic idea is that frames associated with

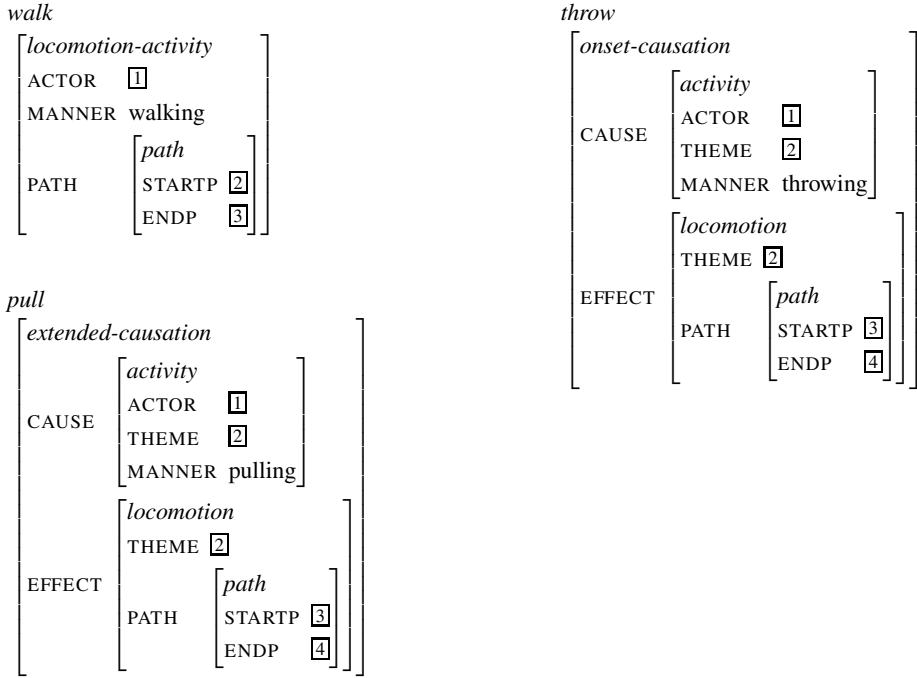


Fig. 1. Possible frame-semantic representations of some verbs of (caused) motion

directional prepositions can unify with frames of pure locomotion to frames that express directed motion. For example, the frame for the preposition *into*, which is shown on the right of Fig. 2 represents (directed) motion to the interior region 2 of an object 1 which is denoted by the nominal complement of the preposition. The frame constraint in the last line encodes the condition that the end point 3 of the path or trajectory generated by the motion is in fact contained in the region in question.

The semantic representations described so far allow us to introduce the basic ideas of syntax-driven semantic frame composition in the following sections. Of course, in a fully developed theory of frame representations for event semantics, the types and

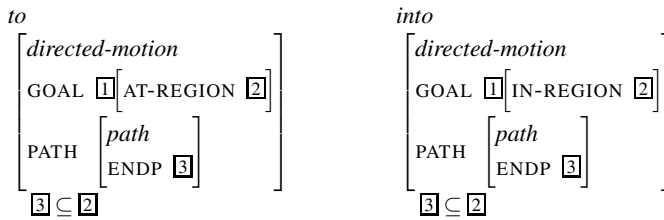


Fig. 2. Frame examples for directional prepositions

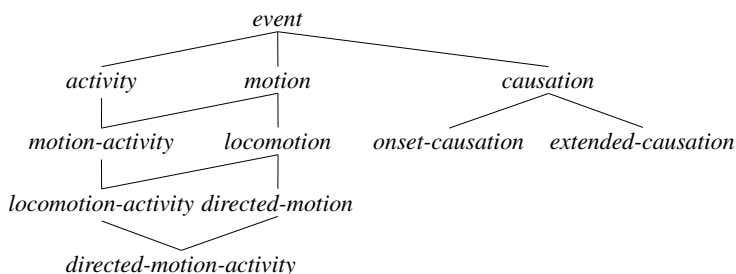


Fig. 3. Partial sketch of the event type hierarchy

features used in the frames are systematically related to each other by a type hierarchy and by feature constraints. For instance, the inheritance hierarchy of the event types introduced so far would look like the one depicted in Fig. 3. Additional feature declarations would then specify, e.g., that frames of type *causation* have a CAUSE and an EFFECT attribute, and that the value of the CAUSE attribute of *onset-causation* events is of type *punctual-event*.

3 LTAG and Grammatical Factorization

3.1 Introduction to TAG

Tree Adjoining Grammar (TAG, [16]) is a tree-rewriting formalism. A TAG consists of a finite set of trees (*elementary trees*). The nodes of these trees are labelled with non-terminals and terminals (terminals only label leaf nodes). Starting from the elementary trees, larger trees are derived by *substitution* (replacing a leaf with a new tree) and *adjunction* (replacing an internal node with a new tree). Sample elementary trees and a derivation are shown in Fig. 4. In this derivation, the elementary tree for *John* substitutes into the subject slot of the elementary tree for *came*, the *in* tree for the temporal PP modifier adjoins to the VP node and *December* substitutes into the NP leaf of the modifier tree.

In case of an adjunction, the tree being adjoined has exactly one leaf that is marked as the *foot node* (marked with an asterisk). Such a tree is called an *auxiliary tree*. To license its adjunction to a node *n*, the root and foot nodes must have the same label as *n*. When adjoining it to *n*, in the resulting tree, the subtree with root *n* from the old tree is attached to the foot node of the auxiliary tree. Non-auxiliary elementary trees are called *initial trees*. A derivation starts with an initial tree. In a final derived tree, all leaves must have terminal labels.

In a TAG, one can specify for each node whether adjunction is mandatory and which trees can be adjoined. The subscripts *NA* and *OA* indicate adjunction constraints: *NA* signifies that for this node, adjunction is not allowed while *OA* signifies that adjunction is obligatory.

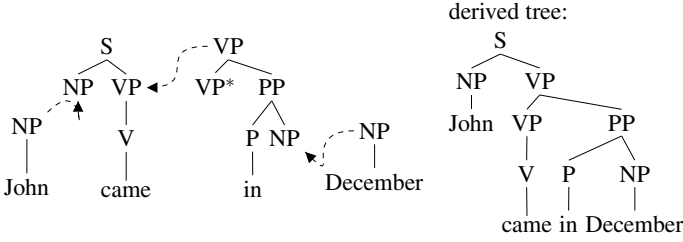


Fig. 4. A sample derivation

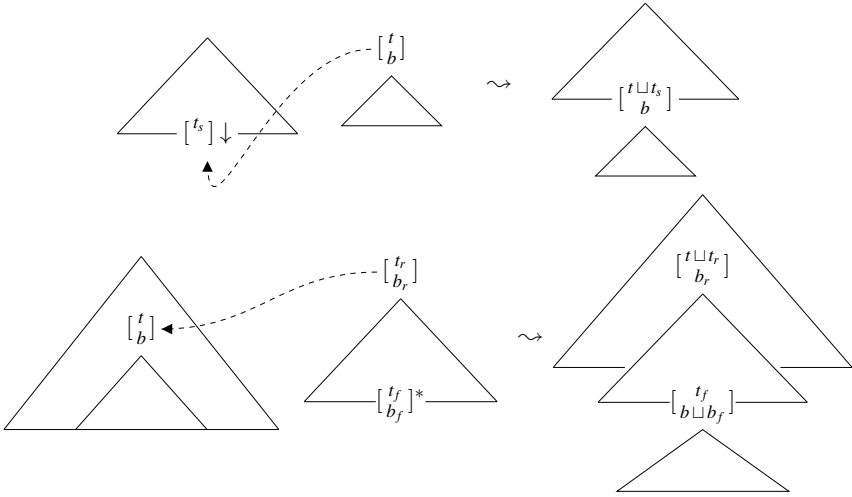


Fig. 5. Feature structure unifications in FTAG

3.2 Feature Structure Based TAG

In order to be able to capture syntactic generalizations in a more satisfying way, the non-terminal node labels in TAG elementary trees are usually enriched with feature structures. The resulting TAG variant is called *Feature-structure based TAG* (FTAG, [24]). In an FTAG, each node has a top and a bottom feature structure (except substitution nodes that have only a top). Nodes in the same elementary tree can share features (extended domain of locality). In contrast to the original TAG, an FTAG does not have separate adjunction constraints, since the constraints can be expressed by features.

During substitution and adjunction, the following unifications take place (see Fig. 5): In a substitution operation, the top of the root of the new initial tree unifies with the top of the substitution node. In an adjunction operation, the top of the root of the new auxiliary tree unifies with the top of the adjunction site and the bottom of the foot of the new tree unifies with the bottom of the adjunction site. Furthermore, in the final derived tree, top and bottom must unify for all nodes.

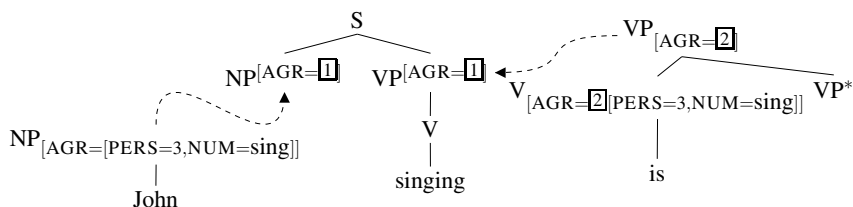


Fig. 6. Agreement with feature structures

Since nodes in the same elementary tree can share features, constraints among dependent nodes can be more easily expressed than in the original TAG formalism. See Fig. 6 for an example (the top feature structure is notated as a superscript, the bottom feature structure as a subscript of the respective node).

3.3 LTAG Elementary Trees

The elementary trees of a TAG for natural languages respect certain principles [11]. Firstly, they are lexicalized, i.e., each elementary tree has at least one non-empty lexical item, its *lexical anchor*. A *lexicalized TAG* (LTAG) is a TAG that satisfies this condition for every elementary tree. Secondly, each elementary tree associated with a predicate contains argument slots (leaves with non-terminal labels, i.e., substitution nodes or foot nodes) for each of its arguments, i.e., for each of the elements it subcategorizes for, including the subject. Furthermore, it contains argument slots only for the arguments of its lexical anchor, and for nothing else (*elementary tree minimality*, [11]).

Most argument slots are substitution nodes, in particular the nodes for nominal arguments (see the elementary tree for *lives* in Fig. 4). Sentential arguments however are realised by foot nodes. The reason is that we want to be able to extract material from sentential arguments in long-distance dependencies such as (5). Such extractions can be obtained by adjoining the embedding clause into the sentential argument.

(5) Whom does Paul think that Mary likes?

As we have seen, the elementary trees of an LTAG are lexicalized and contain non-terminal leaves for all the arguments of their lexical head. Because of this extended domain of locality, LTAG is particularly well-suited for a frame-based compositional semantics. The semantic frame of a predicate specifies, among others, the thematic roles of its arguments. In LTAG, these can be immediately linked to the corresponding syntactic argument slots.

Concerning the modeling of the syntax-semantics interface, we follow approaches that link a single semantic representation (in our case, a semantic frame) to an entire elementary tree and which model semantic composition by unifications triggered by substitution and adjunction [12, 17]. A simplified example that illustrates the locality of linking in this framework is given in Fig. 7. The substitutions trigger unifications between 1 and 3 and between 2 and 4 which leads to an insertion of the corresponding argument frames into the frame of *eats*.

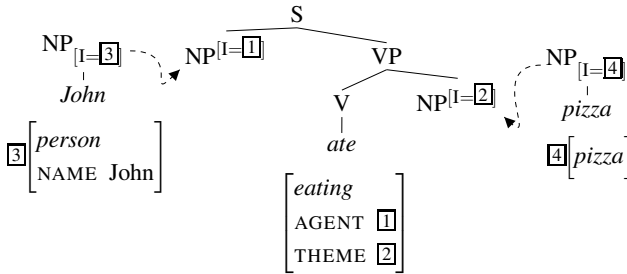


Fig. 7. Syntactic and semantic composition for *John eats pizza*

4 Motion Verbs and Directional PPs

4.1 Directed Motion Activities

This section deals with the combination of motion verbs and directional PPs as in (6).

- (6) a. Mary walked/ran to/into/towards the house.
 b. Mary walked/ran along the river.
 c. Mary walked/ran over the bridge along the fence through the meadows.

Recall that our criterion for deciding whether a constituent is an argument or an adjunct is for us its iterability. Constituents that cannot be iterated and that add a semantic role (no matter whether this is already present in the frame contributed by the verb) are taken to be complements in the sense of being integrated into the unanchored tree for the verb within the metagrammar. For this reason, the examples in (6-a) are treated as PP complements while the PP in (6-b) is an adjunct. PPs of the type in (6-b) can be iterated as can be seen in (6-c).

In the complement cases, the preposition is however not part of the elementary tree of the verb since it is not determined by the verb. This is in contrast to constructions where a specific preposition is treated as a coanchor of the elementary tree. An example is the elementary tree for *remind of* as for instance in (7) where the preposition *of* is taken to be a coanchor of the elementary tree.

- (7) This picture reminds me of my little dog.

As explained above, we assume that the motion verb defines a *locomotion activity* that takes place along a certain path. This path has a start and an end point.

In the construction, the additional PP adds a further argument with the semantic role GOAL. The way this goal combines with the path, i.e., whether it is its end point, whether it adds a direction to the path etc., depends on the preposition.

The unanchored elementary tree for an intransitive verb with an additional directional PP is given in Fig. 9. The lower VP node in the tree is inspired by the XTAG choices. It serves to allow the adjunction of modifiers between the verb and the PP object, as in (8), which would not be possible if the V and the PP were sisters. The empty

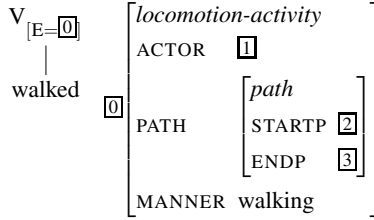


Fig. 8. Lexical entry for *walked*

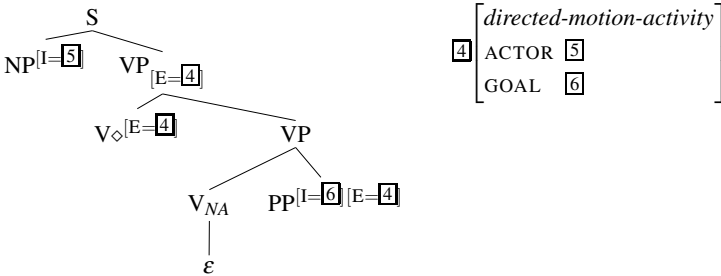


Fig. 9. Unanchored tree and semantics of n0Vpp(dir) construction

V-tree below this additional VP carries a *NA* (null adjunction) constraint. I.e., this node does not allow for adjunction.

(8) He ran every day to the river.

The decoration of the elementary tree with features *I* and *E* makes sure that the substitutions of the subject NP and the object PP will fill the corresponding argument roles and, furthermore, adjunctions of modifiers to the VP node extend the event frame [4].

The preposition determines the relation between the path of the motion and the goal. Fig. 10 shows the elementary trees of different directional prepositions. We assume that objects such as *the house* have a certain topological structure. They come with different types of regions, an *at-region* that contains all points that can be said to be *at* the object, an *in-region* that determines the space that constitutes the inner part of the object etc. The preposition *to* refers to the *at-region* of an object; it expresses that the endpoint of the path must be contained in the *at-region* of the object in the PP. Similarly, *into* expresses that the endpoint must be contained in the *in-region* of the PP object. In contrast to this, *towards* does not determine the end or start point, it only says something about the direction of the path.

The nature of the contained-in relation is different from the functional attributes in frames. A region can of course be contained in several other regions, consequently a formalization via a frame attribute *CONTAINED_IN* is not possible. Therefore, the relation of containment between regions is formulated outside the attribute value structure itself, i.e., we formalize it as an additional relation \sqsubseteq between elements of type *region*

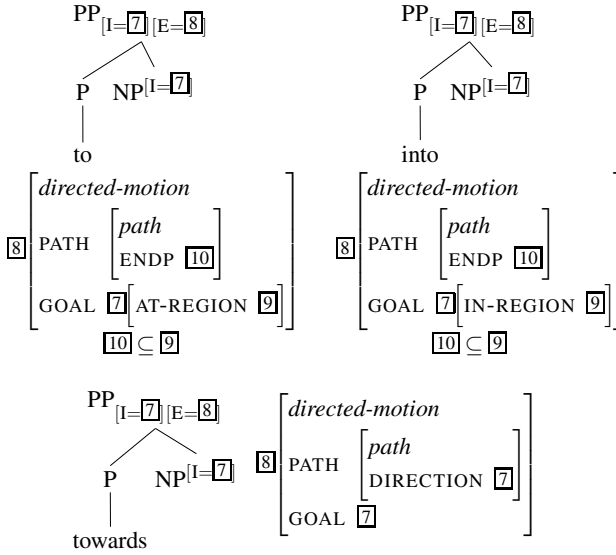


Fig. 10. Elementary trees for prepositions

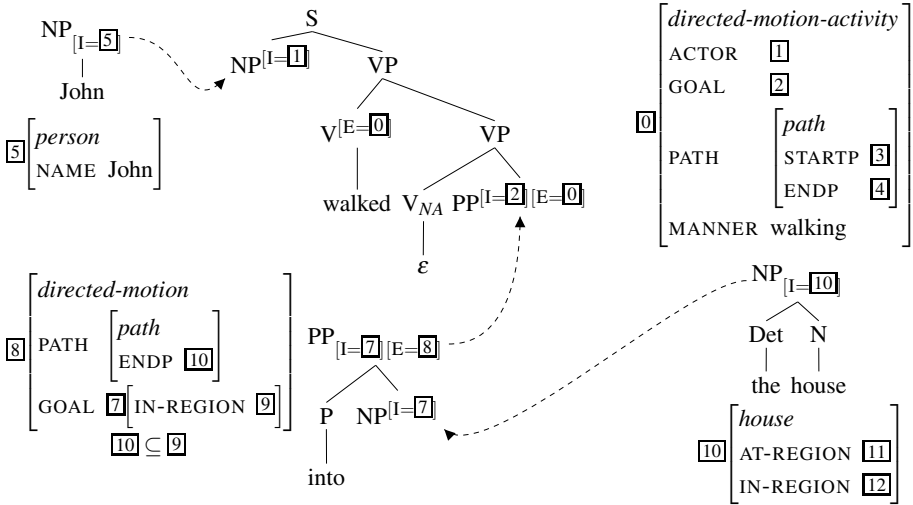


Fig. 11. Derivation of (9)

in our frames. We assume a mereological structure on regions where single location points are considered as regions as well. This relation is a partial order relation, i.e., it is reflexive, transitive and antisymmetric.

Now let us consider as an example the derivation of (9). Fig. 11 shows the elementary trees and frames that are involved and how they are combined. The tree for *the house*

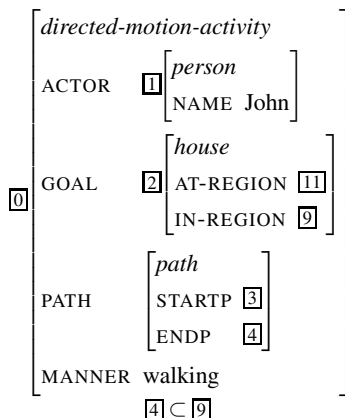


Fig. 12. Frame obtained for (9) *John walked into the house*

comes with both an in-region and an at-region. (The composition of the determiner and the noun into the NP *the house* is left aside in this example.) The preposition *into* links the in-region to the end point of the path traversed throughout the walking activity. Because of the various substitution, we obtain the following equations: $\boxed{1} = \boxed{5}$, $\boxed{2} = \boxed{7} = \boxed{10}$ and $\boxed{0} = \boxed{8}$. With the corresponding unifications, the resulting frame is the one given in Fig. 12.

(9) John walked into the house.

The difference between verbs of locomotion such as in (9) and motion verbs as in (10) that are turned into a directed motion by adding a goal and a path is the semantics of the verb. *walk* comes with a path while *dance* does not. The lexical frame for *dance* is shown in Fig. 13. When combining it with the unanchored construction tree, the path attribute is added and the goal argument is linked to the PP.

(10) Mary danced into the room.

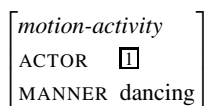


Fig. 13. Frame for *dance*

4.2 Path Modification

Now let us consider the case where the directional PP is an adjunct that gives an additional specification of the path of the event as in (6-b). In these cases, the verb of locomotion anchors an intransitive activity tree as in Fig. 14. As before, *walked* comes with a path. But there need not be a goal restricting this path.

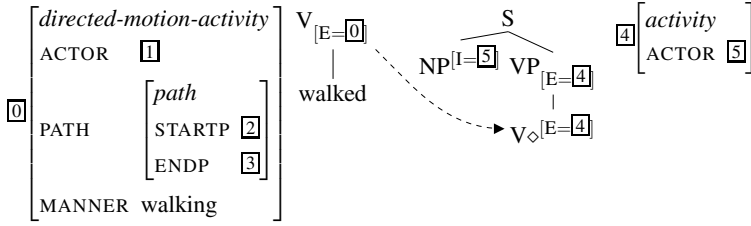


Fig. 14. Lexical anchoring for intransitive *walked*

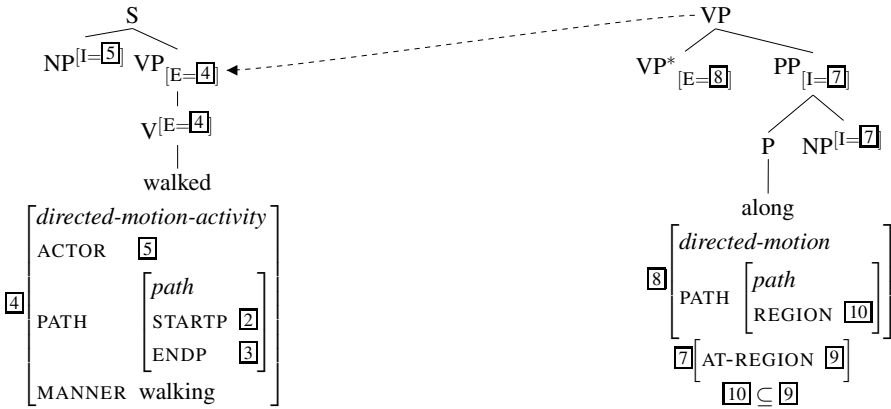


Fig. 15. Derivation for (11)

As an example, consider the derivation of (11). Fig. 15 shows the adjunction of the *along* elementary tree into the anchored elementary tree of the intransitive *walked*. The frame(s) linked to *along* express that the NP within the PP has an at-region that must contain the entire region of the path. Note that the frame contributed by the preposition does not have a unique root. The reason for this is that the NP does not contribute an argument and therefore it does not fill a semantic role slot. The link between it and the walking activity concerns only its at-region.

(11) John walked along the brook.

As a result, when combining further with the elementary trees for *John* and *the brook*, we obtain the frame in Fig. 16. We represent the frame using avms with relational constraints and, in order to emphasize that this frame is not a tree and, if we do not consider the additional relation \subseteq , not even a connected graph, we also depict the corresponding graph in Fig. 16.

Obviously, examples with motion verbs that are not necessarily directional such as (12) work as well.

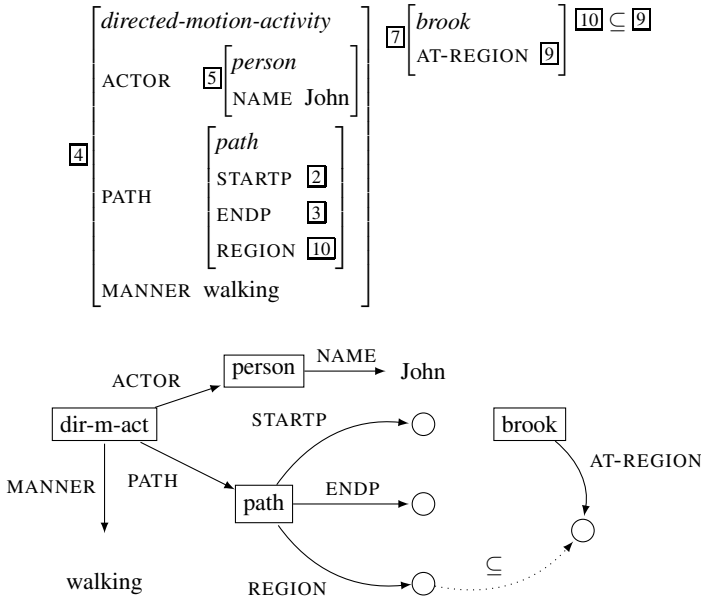


Fig. 16. Frame for (11)

(12) Mary danced along the fence

As a last example, let us consider a combination of argument directional PPs and adjoining directional PPs.

(13) Mary walked along the brook into the field

The derivation step combining the *along the brook* PP with the rest of the sentence is shown in Fig. 17. As one can see, when performing the unification of 8 and 9 triggered by the adjunction, we obtain a resulting frame that combines the two constraints on the path contributed by the two PPs: The entire path (i.e., its REGION) must be contained in the AT-REGION of the brook and the ENDP (endpoint) of the path must be contained in the IN-REGION of the field.

4.3 Caused Motion

We now turn to verbs of transport and caused motion as exemplified in (14).

- (14) a. Mary threw the ball into the hole.
- b. Mary pulled the cart along the river.
- c. Mary kicked the ball along the line into the goal.

Our proposal for the unanchored construction and its semantics is shown in Fig. 18. The difference to the directed motion construction without a direct object n0Vpp(dir)

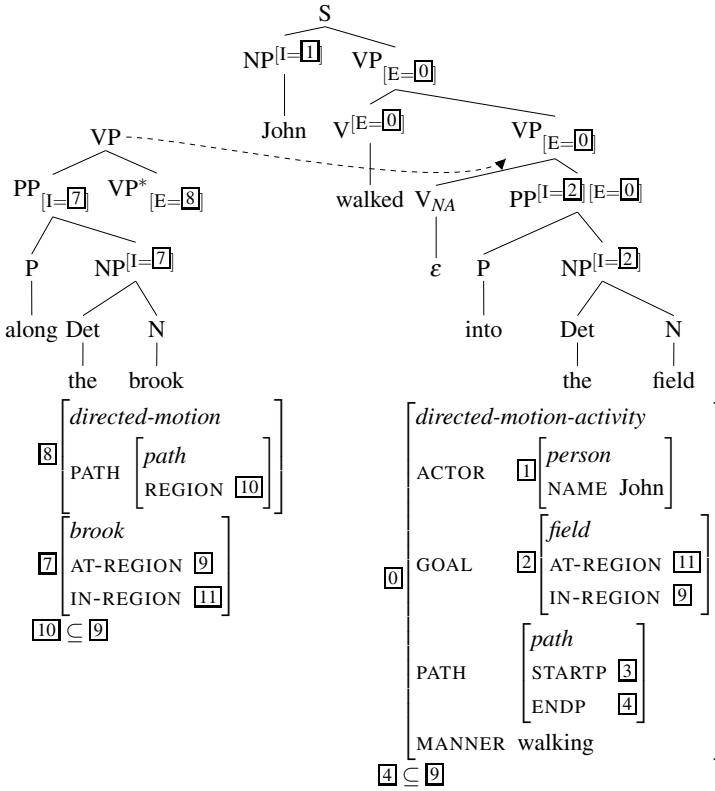


Fig. 17. Derivation of (13)

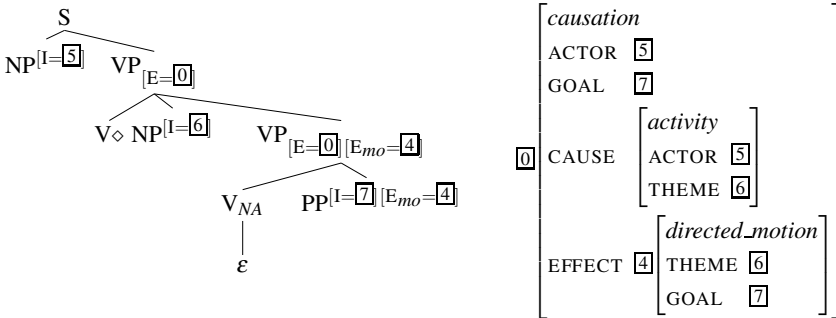


Fig. 18. Unanchored tree and semantics of nOVn1pp(dir) construction

discussed above is that now the object, i.e., the theme is moved. This movement is the effect of an action performed by the actor that affects the theme. Therefore the directed motion of the object (the theme) is embedded as the effect of a causation whose cause is an action performed by the subject.

A difficulty with this construction is that the PP argument and also directional PP modifiers want to access the embedded event [4] while other modifiers might want to access [0]. As a solution that makes both accessible and that distinguishes them, we propose to use the feature E in the syntactic trees for the highest event (here [0]) while using a feature E_{mo} for the relevant motion event if this exists. In $n0Vpp(dir)$ the two features have the same value.

Obviously, when using this construction, a directional PP argument that substitutes into the PP slot as in (14-a) can modify the embedded motion event via the E_{mo} feature. Similarly, a directional PP that adjoins as a modifier to the lower VP node as in (14-c) can also access the embedded event via the E_{mo} feature and modify its $PATH$ attribute.

5 Metagrammar Decomposition

This section deals with the further decomposition of the meaning of unanchored elementary trees.

5.1 Metagrammar and Factorization

LTAG allows for a high degree of factorization inside the lexicon, i.e., inside the set of lexicalized elementary trees. Firstly, as we have seen above, unanchored elementary trees are specified separately from their lexical anchors. The set of unanchored elementary trees is partitioned into *tree families* where each family represents the different realizations of a single subcategorization frame. For transitive verbs such as *hit*, *kiss*, *admire*, etc. there is a tree family (see Fig. 19) containing the patterns for different realizations of the arguments (canonical position, extraction, etc.) in combination with active and passive. The node marked with a diamond is the node that gets filled by the lexical anchor.

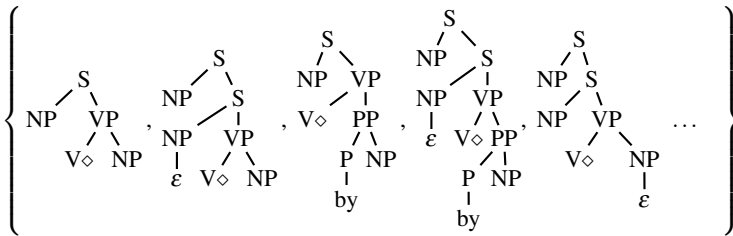


Fig. 19. Unanchored tree family for transitive verbs

Secondly, unanchored elementary trees are usually specified by means of a *metagrammar* [3,4] which consists of dominance and precedence constraints and category assignments. The elementary trees of the grammar are defined as the *minimal models* of this constraint system. The metagrammar formalism allows for a compact grammar definition and for the formulation of linguistic generalizations. In particular, the metagrammatical specification of a subcategorization frame defines the set of all unanchored

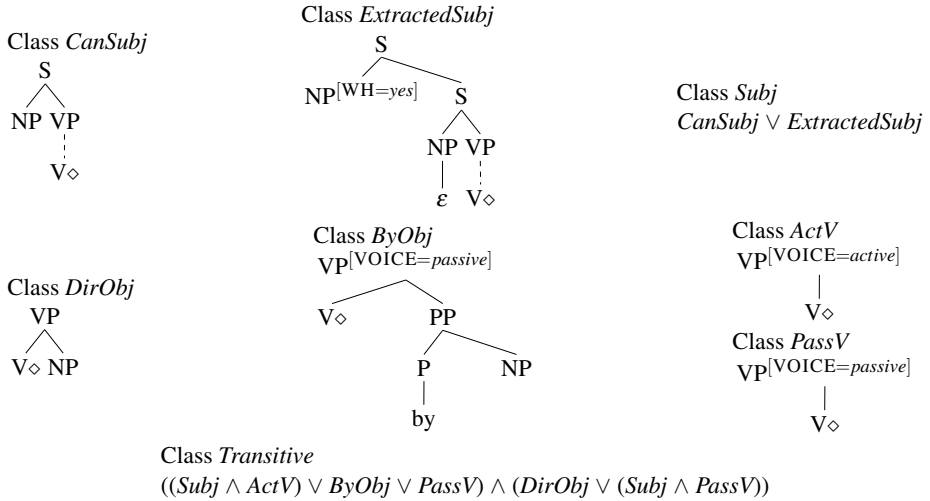


Fig. 20. MG fragment for transitive verbs

elementary trees that realize this frame. Moreover, the formalism allows us to define tree fragments that can be used in different elementary trees and tree families, thereby giving rise to an additional factorization and linguistic generalization. Phenomena that are shared between different tree families such as passivization or the extraction of a subject or an object are specified only once in the metagrammar and these descriptions become part of the descriptions of several tree families.

Let us illustrate this with the small metagrammar fragment given in Fig. 20, which is of course very incomplete in that many tree fragments are missing and features are almost totally omitted. The first two tree fragments describe possible subject realizations: the subject can be in canonical position, immediately preceding the VP, or it can be extracted, with a trace in the canonical subject position. The class *Subj* comprises the different subject realizations. Similar classes exist for the different realizations of the object, while in Fig. 20 only the canonical position class is listed. Furthermore, there is a class for the *by*-PP in a passive construction. This is used only for passive, therefore the tree fragment contains a corresponding feature *VOICE = passive*. Besides these argument classes, our fragment contains two classes for active/passive morphology. Finally, the class *Transitive* specifies for each argument its different grammatical functions: the first argument can be the subject of an active sentence or the *by*-PP of a passive sentence or it can be omitted in a passive sentence². The second argument can be the direct object or it can be promoted to a subject in a passive sentence. If we assume that the metagrammar constraints require the identification of the lexical anchor nodes, then the minimal models of this class are among others the first four tree in Fig. 19. Note that the difference between canonical subject and extracted subject is factored out in the class *Subj*, which can also be used for the definition of other tree families.

² We are computing minimal models, this is why the third possibility in the disjunction signifies that this argument is not realized.

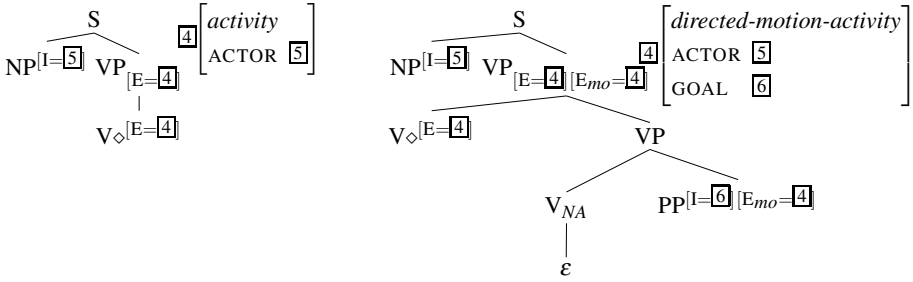


Fig. 21. Unanchored trees for intransitive activity verbs, possibly with a directional PP

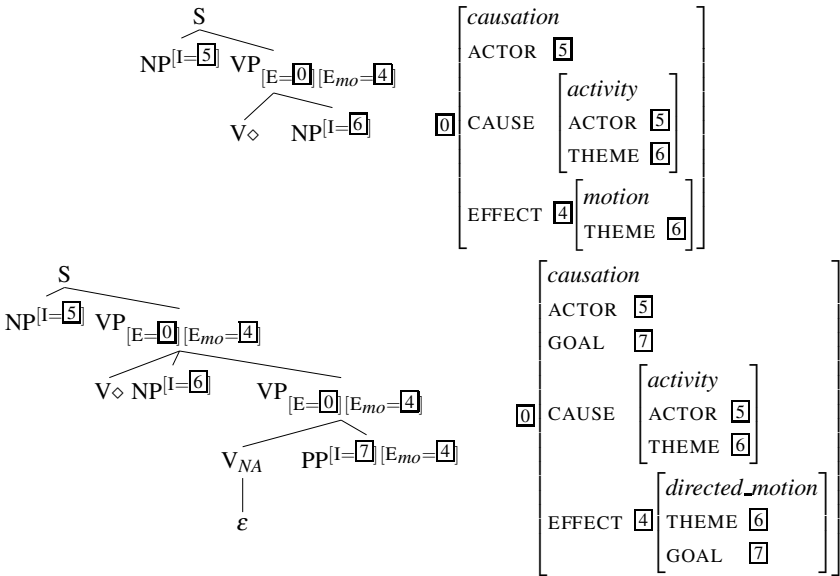


Fig. 22. Unanchored trees for transitive caused motion verbs, possibly with a directional PP

A similar factorization is possible within the semantics. The semantic contribution of unanchored elementary trees, i.e., constructions, can be separated from their lexicalization, and the meaning of a construction can be decomposed further into the meaning of fragments of the construction. Due to this factorization, relations between the different parts of a certain syntactic construction and the components of a semantic representation can be expressed.

5.2 Metagrammar Decomposition of Directed Motion Constructions

So far, we have seen how lexical anchoring contributes to semantic composition and how substitutions and adjunctions trigger semantic unifications that yield then the frame

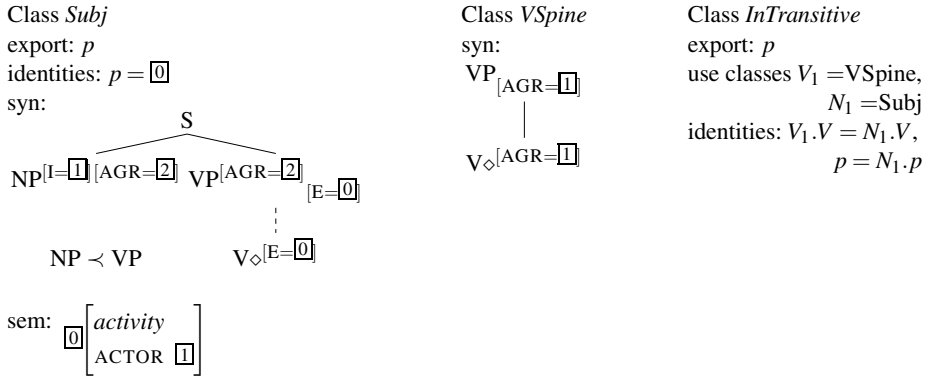


Fig. 23. MG classes for intransitive activity verbs

of an entire sentence. Now we will have a look at the metagrammatical decomposition of the unanchored trees for $n0Vpp(\text{dir})$ and $n0Vn1pp(\text{dir})$, shown in Fig. 21 and 22.

In the following, we restrict ourselves to the base trees when explaining the syntactic and semantic decomposition. Of course, other argument realizations are possible as well and should be taken into account in the metagrammar classes. We leave this aside in this paper.

Let us first consider the classes needed for the subject, the verbal spine and the intransitive elementary trees. They are shown in Fig. 23. Each class has a name, a declaration of variables that one can refer to when using this class (the export variables), a list of equations, and a syntactic dimension and a semantic dimension. The syntactic dimension contains a tree description that is depicted in the usual way in the figure. I.e., solid lines indicate immediate dominance, dotted lines indicate dominance and the order of sisters indicates linear precedence (but not necessarily immediate linear precedence). Furthermore, \prec denotes immediate linear precedence. In the class *Subj* for instance, the tree description tells us that there are three nodes n_1, n_2, n_3 with labels S, NP and VP such that n_2 has a top feature I with value $\boxed{1}$. Furthermore, n_1 immediately dominates n_2 and n_3 (depicted by the edges) and n_2 immediately precedes n_3 (constraint NP \prec VP). The picture is a little sloppy since it mixes node variables with node categories. The subject adds an actor to the semantic frame.³

Concerning the semantic dimension, we assume this to be a description of a typed feature structure. When we say “unification”, speaking of combining frames in the metagrammar, we actually mean conjunction and feature value equation. So far, our impression is that we need only a simple feature logic without quantification or negation.

The class for the verbal spine takes care of the percolation of features (for instance AGR) along the verbal spine. *InTransitive* combines the verbal spine with the subject. This yields an identification of the VP and V nodes in both classes and the resulting frame is the one coming from the subject class. When computing the minimal model of *InTransitive*, we obtain the unanchored tree on the left of Fig. 21.

³ This is of course not the only way this syntactic fragment can be used; other possibilities for the semantic role of the subject exist as well.

Note that we assume that, whenever we use a class, its meta-variables ($[0]$, $[1]$, etc.) get instantiated with fresh values. This avoids unintended unifications.

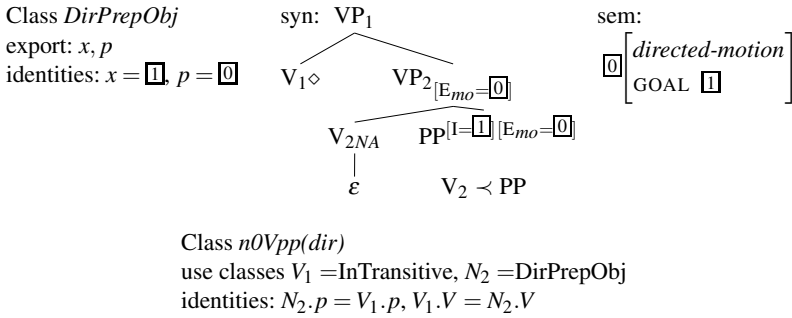


Fig. 24. MG classes for intransitive construction with directional PP

For the construction involving an additional directional PP complement, we combine the *InTransitive* class with a class *DirPPObj* for a directional PP-argument. The PP contributes the goal of some directed motion. The higher class *n0Vpp(dir)* arises from a combination of the *InTransitive* class and the class for the directional PP. The motion frame contributed by the PP is unified with the activity frame contributed by the *InTransitive* class. Note that only in the *n0Vpp(dir)* class, the E feature and the E_{mo} feature get identified via the equation $N_2.p = V_1.p$. The class *DirPPObj* can also be used in a context where the E and E_{mo} features are different.

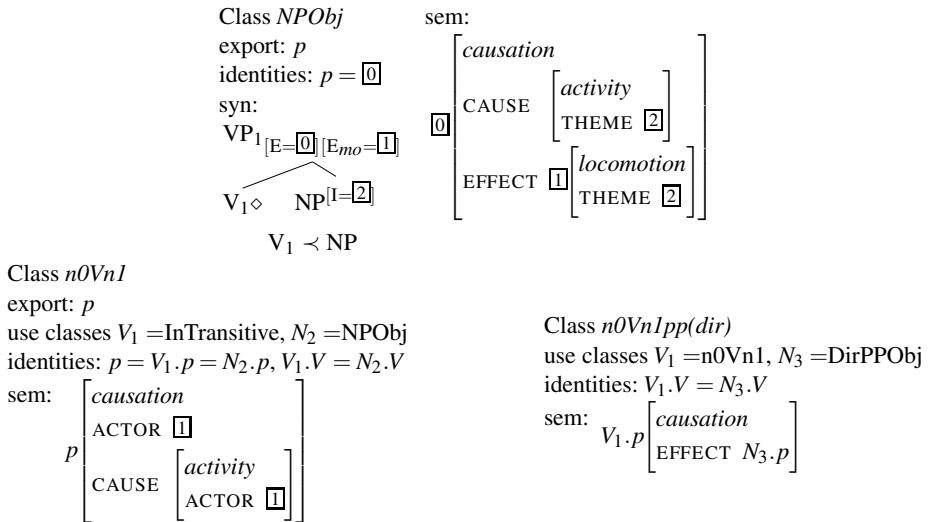


Fig. 25. MG classes for the n0vn1pp(dir) construction

For the class $n0Vn1pp(dir)$, we need a further argument class for the direct NP object in this construction. A direct NP argument can have different roles of course. In the context we are interested in, namely the caused motion, the NP object contributes an argument that is the theme of the causing activity and also the theme of the caused directed motion. Therefore we assume the metagrammar class $NPObj$ from Fig. 25. When combining this with the intransitive class, the activity event denoted by the p feature of the intransitive class gets further specified as being a causation where the actor is also the actor of the causing event. This is expressed in the class $n0Vn1$. Its minimal model is the unanchored tree on the left of Fig. 22.

Finally, when adding the directional PP, its directed motion event becomes the *EFFECT* of the causation. This embedding is specified in the values of the *EFFECT* feature in the class $n0Vn1pp(dir)$.

With this metagrammar decomposition we were able to capture the fact that the directional PP always contributes the goal of a directed motion, independent from the construction it gets combined with. In the $n0Vpp(dir)$ case, the directed motion is the event denoted by the lexical anchor while in the $n0Vn1pp(dir)$ case, the directed motion is embedded as the effect of the causation denoted by the lexical anchor.

6 Conclusion

In this paper, we proposed to combine an LTAG-based syntax-semantics interface with a fine-grained frame-based semantics. We have shown that this architecture provides the means to associate a detailed decomposition and composition of syntactic building blocks with a parallel decomposition and composition of meaning components. Due to its various possibilities for decomposing elementary trees and because of its extended domain of locality, LTAG allows one to pair not only lexical items with lexical meaning but also constructions with their meaning contributions. Furthermore, due to the metagrammatical specification of TAG elementary trees, the meaning contributions of single argument realizations and of their combinations can be described in a principle way, in parallel to a similar decomposition of the syntactic elementary trees.

We have discussed the case of directed motion expressions and we have shown how to capture the various ways a directional PP adds information about the path of the motion event. Besides giving a detailed frame-based analysis of lexical and constructional meaning aspects, our approach integrates this into a syntax-semantics interface. Via substitution and adjunction, the frame-based characterization of the events described by entire sentences can be compositionally derived.

References

1. Abeillé, A.: Une Grammaire Électronique du Français. CNRS Editions, Paris (2002)
2. Barsalou, L.W.: Frames, concepts, and conceptual fields. In: Lehrer, A., Kittay, E.F. (eds.) *Frames, Fields, and Contrasts*, pp. 21–74. Lawrence Erlbaum Associates, Hillsdale (1992)
3. Candito, M.H.: Organisation modulaire et paramétrable de grammaires électroniques lexicalisées. Application au français et à l'italien. Ph.D. thesis, Université Paris 7 (1999)
4. Crabbé, B., Duchier, D.: Metagrammar Redux. In: Christiansen, H., Skadhauge, P.R., Villadsen, J. (eds.) *CSLP 2005. LNCS (LNAI)*, vol. 3438, pp. 32–47. Springer, Heidelberg (2005)

5. Dowty, D.: *Word Meaning and Montague Grammar*. D. Reidel, Dordrecht (1979)
6. Dowty, D.: The dual analysis of adjuncts/complements in Categorical Grammar. In: Lang, E., Maienborn, C., Fabricius-Hansen, C. (eds.) *Modifying Adjuncts*. Interface Explorations 4, pp. 33–66. Mouton de Gruyter, Berlin (2003)
7. Ehrich, V.: *Verbbedeutung und Verbgrammatik: Transportverben im Deutschen*. In: Lang, E., Zifonun, G. (eds.) *Deutsch - Typologisch*, pp. 229–260. de Gruyter, Berlin (1996)
8. Eschenbach, C., Tschander, L., Habel, C., Kulik, L.: Lexical specifications of paths. In: Habel, C., Brauer, W., Freksa, C., Wender, K.F. (eds.) *Spatial Cognition 2000*. LNCS (LNAI), vol. 1849, pp. 127–144. Springer, Heidelberg (2000)
9. Fillmore, C.J.: *Frame semantics*. In: *Linguistics in the Morning Calm*, pp. 111–137. Hanshin Publishing Co., Seoul (1982)
10. Fillmore, C.J., Johnson, C.R., Petruck, M.R.L.: Background to FrameNet. *International Journal of Lexicography* 16(3), 235–250 (2003)
11. Frank, R.: *Phrase Structure Composition and Syntactic Dependencies*. MIT Press, Cambridge (2002)
12. Gardent, C., Kallmeyer, L.: *Semantic Construction in FTAG*. In: *Proc. EACL 2003*, pp. 123–130 (2003)
13. Gehrke, B.: *Ps in Motion. On the semantics and syntax of P elements and motion events*. LOT, Utrecht (2008)
14. Goldberg, A.E., Jackendoff, R.: The English resultative as a family of constructions. *Language* 80, 532–568 (2004)
15. Jackendoff, R.: Parts and boundaries. *Cognition* 41, 9–45 (1991)
16. Joshi, A.K., Schabes, Y.: *Tree-Adjoining Grammars*. In: *Handbook of Formal Languages*, pp. 69–123. Springer (1997)
17. Kallmeyer, L., Romero, M.: Scope and situation binding in LTAG using semantic unification. *Research on Language and Computation* 6(1), 3–52 (2008)
18. Kaufmann, I.: *Konzeptuelle Grundlagen semantischer Dekompositionsstrukturen. Die Kombinatorik lokaler Verben und prädikativer Argumente*. Niemeyer, Tübingen (1995)
19. Mani, I., Pustejovsky, J.: *Interpreting Motion. Grounded Representations for Spatial Language*. Oxford University Press, Oxford (2012)
20. Talmy, L.: *Toward a Cognitive Semantics. Concept Structuring Systems, vol. I*. MIT Press, Cambridge (2000)
21. Talmy, L.: *Toward a Cognitive Semantics. Typology and Process in Concept Structuring, vol. II*. MIT Press, Cambridge (2000)
22. Van Valin, R.D., LaPolla, R.J.: *Syntax*. Cambridge University Press, Cambridge (1997)
23. Verkuyl, H., Zwarts, J.: Time and space in conceptual and logical semantics: The notion of path. *Linguistics* 30, 483–511 (1992)
24. Vijay-Shanker, K., Joshi, A.K.: Feature structures based tree adjoining grammar. In: *Proceedings of COLING, Budapest*, pp. 714–719 (1988)
25. Zwarts, J.: Prepositional aspect and the algebra of paths. *Linguistics and Philosophy* 28(6), 739–779 (2005)

Admissible Rules: From Characterizations to Applications

George Metcalfe*

Mathematics Institute, University of Bern
Sidlerstrasse 5, Bern 3012, Switzerland
george.metcalfe@math.unibe.ch

Abstract. The admissible rules of a logic (understood as a structural consequence relation) may be described as rules that can be added to the logic without producing any new theorems, or, equivalently, as rules such that any substitution making the premises into theorems, also makes the conclusion into a theorem. However, this equivalence collapses once multiple-conclusion or other, more exotic, admissible rules are considered. The first aim of this paper is to explain how such distinctions can be explained and characterized. The second aim is to explore how these rules can be useful in determining properties of classes of algebras.

1 Introduction

The notion of an admissible rule was introduced explicitly by Lorenzen in the 1950s in the context of intuitionistic logic [20], but appears also, at least implicitly, in Gentzen's papers on the sequent calculus [10] and Whitman's work on free lattices [28]. Admissible rules have since been studied intensively by many authors. In particular, Rybakov showed that the set of admissible rules of intuitionistic logic is decidable, but not finitely axiomatizable [26]. An elegant infinite axiomatization (conjectured by De Jongh and Visser) of this set of rules was later provided by Iemhoff [14] (based on the work of Ghilardi [11,12] relating admissibility to unification) and, independently, by Rozière [25]. Axiomatizations have also been provided for a range of intermediate logics [15,8], transitive modal logics [17], and various many-valued logics [18,19,5], leading in some cases also to proof systems for checking admissibility [13,16,3,24].

The starting point for the work reported here is the observation that two seemingly quite opposed notions of admissibility are employed in the literature. Informally, for a system S and rules consisting of a finite set of premises and finite set of conclusions:

- (A) A rule is *admissible* in S if the set of theorems of S does not change when the rule is added to the existing rules of S .
- (B) A rule is *admissible* in S if each substitution mapping all of its premises to theorems of S , also maps one of its conclusions to a theorem of S .

* Supported by Swiss National Science Foundation grant 20002_129507.

In many cases – in particular, for the single-conclusion rules of a logic (structural consequence relation) – these two notions of admissibility coincide. Moreover, in an algebraic setting, with logics and single-conclusion rules corresponding, respectively, to quasivarieties and quasiequations, admissibility amounts to validity in free algebras on countably infinitely many generators. For multiple-conclusion rules, however, and more exotic rules with restrictions on variables, these two notions may diverge.

The first task of this paper, undertaken in Section 2, is to give an algebraic account of admissibility according to notions (A) and (B), seeing where these two notions agree and where the equivalence breaks down. (Note that a treatment of admissibility for algebraizable logics, i.e., structural consequence relations enjoying well-behaved translations between equations and formulas, follows directly from this algebraic account.) A new, more general, first-order framework is then introduced in Section 3, relating admissibility to the preservation of certain classes of sentences (equations, quasiequations, etc.) by particular sentences with respect to a given class of algebras. Finally, in Section 4, some applications of admissibility for determining properties of classes of algebras are described.

2 An Algebraic Perspective

For convenience, let us assume in what follows that \mathcal{L} is an algebraic language and that an \mathcal{L} -algebra \mathbf{A} is an algebraic structure for this language with universe A . We denote the *term algebra* (absolutely free algebra) for \mathcal{L} over countably infinitely many variables by $\mathbf{Tm}_{\mathcal{L}}$ and let s, t, u stand for \mathcal{L} -terms in $\mathbf{Tm}_{\mathcal{L}}$. An \mathcal{L} -equation is an ordered pair of \mathcal{L} -terms, written $s \approx t$, and we let the metavariables Γ, Δ stand for finite sets of \mathcal{L} -terms. An \mathcal{L} -clause is an ordered pair of finite sets of \mathcal{L} -equations, written $\Gamma \Rightarrow \Delta$, called an \mathcal{L} -quasiequation if $|\Delta| = 1$, an \mathcal{L} -positive clause if $\Gamma = \emptyset$, and identified with the single equation in Δ if $|\Delta| = 1$ and $\Gamma = \emptyset$.

Throughout this paper, whenever \mathcal{L} contains a binary connective \wedge , we make use of $s \leq t$ as an abbreviation for $s \wedge t \approx s$.

Let us fix a class of \mathcal{L} -algebras \mathcal{K} and a finite set of \mathcal{L} -equations $\Gamma \cup \Delta$. We write $\Gamma \models_{\mathcal{K}} \Delta$ to denote that for every $\mathbf{A} \in \mathcal{K}$ and homomorphism $h: \mathbf{Tm}_{\mathcal{L}} \rightarrow \mathbf{A}$, $\Gamma \subseteq \ker h$ implies $\Delta \cap \ker h \neq \emptyset$. In this case, we say that $\Gamma \Rightarrow \Delta$ is *valid* in each $\mathbf{A} \in \mathcal{K}$. That is, $\Gamma \Rightarrow \Delta$ may be understood as the universal formula (i.e., of first-order logic) $(\forall \bar{x})(\bigwedge \Gamma \Rightarrow \bigvee \Delta)$ where \bar{x} are the variables occurring in $\Gamma \cup \Delta$ and $\bigwedge \emptyset = 1, \bigvee \emptyset = 0$. Conversely, an arbitrary universal formula of the language \mathcal{L} may be associated (by putting the quantified formula into conjunctive normal form) with a finite set of \mathcal{L} -clauses. We abbreviate $\emptyset \models_{\mathcal{K}} \Delta$ by $\models_{\mathcal{K}} \Delta$, and $\Gamma \models_{\{\mathbf{A}\}} \Delta$ by $\Gamma \models_{\mathbf{A}} \Delta$. We also drop the brackets in Γ, Δ when no confusion may occur. As usual, if the language is clear from the context we may omit the prefix \mathcal{L} when referring to these concepts.

\mathcal{K} is said to be an \mathcal{L} -universal class if there exists a set of \mathcal{L} -clauses Λ such that $\mathbf{A} \in \mathcal{K}$ iff all clauses in Λ are valid in \mathbf{A} . If there exists such a Λ consisting only of quasiequations, positive clauses, or equations, then \mathcal{K} is called, respectively,

an \mathcal{L} -quasivariety, an \mathcal{L} -positive universal class, or an \mathcal{L} -variety. The variety $\mathbb{V}(\mathcal{K})$, quasivariety $\mathbb{Q}(\mathcal{K})$, positive universal class $\mathbb{U}^+(\mathcal{K})$, and universal class $\mathbb{U}(\mathcal{K})$ generated by \mathcal{K} are, respectively, the smallest variety, quasivariety, positive universal class, and universal class containing \mathcal{K} .

Let \mathbb{H} , \mathbb{I} , \mathbb{S} , \mathbb{P} , and \mathbb{P}_U be, respectively, the class operators of taking homomorphic images, isomorphic images, subalgebras, products, and ultraproducts. Then $\mathbb{V}(\mathcal{K}) = \mathbb{HSP}(\mathcal{K})$, $\mathbb{Q}(\mathcal{K}) = \mathbb{ISP}\mathbb{P}_U(\mathcal{K})$, $\mathbb{U}^+(\mathcal{K}) = \mathbb{HSP}_U(\mathcal{K})$, and $\mathbb{U}(\mathcal{K}) = \mathbb{ISP}_U(\mathcal{K})$. Moreover, if \mathcal{K} is a finite set of finite algebras, these latter equivalences refine to $\mathbb{Q}(\mathcal{K}) = \mathbb{ISP}(\mathcal{K})$, $\mathbb{U}^+(\mathcal{K}) = \mathbb{HS}(\mathcal{K})$, and $\mathbb{U}(\mathcal{K}) = \mathbb{IS}(\mathcal{K})$ (see [4, Theorems II.9.5, II.11.9, V.2.20, and V.2.25] and [6, Exercise 3.2.2] for further details).

Let us consider first the “standard” characterization of admissibility (notion (B) from the introduction), fixing a quasivariety \mathcal{Q} for the remainder of this section. We say that an \mathcal{L} -clause $\Gamma \Rightarrow \Delta$ is \mathcal{Q} -admissible if for every homomorphism (substitution) $\sigma: \mathbf{Tm}_{\mathcal{L}} \rightarrow \mathbf{Tm}_{\mathcal{L}}$:

$$\begin{array}{ccc} \models_{\mathcal{Q}} \sigma(s) \approx \sigma(t) & \text{implies} & \models_{\mathcal{Q}} \sigma(s') \approx \sigma(t') \\ \text{for all } s \approx t \in \Gamma & & \text{for some } s' \approx t' \in \Delta. \end{array}$$

This notion of \mathcal{Q} -admissibility is equivalent (at least in algebraic contexts) to validity in the free algebra on countably many generators of \mathcal{Q} . Recall that for a cardinal κ , an \mathcal{L} -algebra \mathbf{B} is called a *free κ -generated algebra* $\mathbf{F}_{\mathcal{Q}}(\kappa)$ if there exists $X \subseteq B$ such that $|X| = \kappa$ and \mathbf{B} has the *universal mapping property* for \mathcal{Q} over X ; that is, for every $\mathbf{A} \in \mathcal{Q}$, and map $f: X \rightarrow \mathbf{A}$ there exists a (unique) homomorphism $g: \mathbf{B} \rightarrow \mathbf{A}$ extending f .

Note that when considering admissibility, it can be helpful to view the elements of $\mathbf{F}_{\mathcal{Q}}(\kappa)$ for $\kappa \leq \omega$ as equivalence classes $[t]$ of terms t containing at most κ variables, defined with respect to the congruence relating s and t whenever $\models_{\mathcal{K}} s \approx t$. In particular, the *canonical homomorphism* $h_{\mathcal{Q}}: \mathbf{Tm}_{\mathcal{L}} \rightarrow \mathbf{F}_{\mathcal{Q}}(\omega)$ is the unique homomorphism mapping a term t to its equivalence class $[t]$ in $\mathbf{F}_{\mathcal{Q}}(\omega)$, recalling (see [4, Corollary II.11.6]) that for each \mathcal{L} -equation $s \approx t$:

$$\models_{\mathcal{Q}} s \approx t \quad \text{iff} \quad \models_{\mathbf{F}_{\mathcal{Q}}(\omega)} s \approx t \quad \text{iff} \quad h_{\mathcal{Q}}(s) = h_{\mathcal{Q}}(t). \quad (1)$$

Quasivarieties are closed under taking products; hence, given any finite set of \mathcal{L} -equations $\Gamma \cup \Delta$:

$$\Gamma \models_{\mathcal{Q}} \Delta \quad \text{iff} \quad \Gamma \models_{\mathcal{Q}} s \approx t \text{ for some } s \approx t \in \Delta \quad (2)$$

and combining (1) and (2):

$$\models_{\mathcal{Q}} \Delta \quad \text{iff} \quad \models_{\mathcal{Q}} s \approx t \text{ for some } s \approx t \in \Delta \quad \text{iff} \quad \models_{\mathbf{F}_{\mathcal{Q}}(\omega)} \Delta. \quad (3)$$

Lemma 1. $\Gamma \Rightarrow \Delta$ is \mathcal{Q} -admissible iff $\Gamma \models_{\mathbf{F}_{\mathcal{Q}}(\omega)} \Delta$.

Proof. (\Rightarrow) Suppose that $\Gamma \Rightarrow \Delta$ is \mathcal{Q} -admissible and consider a homomorphism $g: \mathbf{Tm}_{\mathcal{L}} \rightarrow \mathbf{F}_{\mathcal{Q}}(\omega)$ such that $\Gamma \subseteq \ker g$. Let σ be a map sending each variable x to a member of the equivalence class $g(x)$. By the universal mapping property

for $\mathbf{Tm}_{\mathcal{L}}$ for \mathcal{L} -algebras, this extends to a homomorphism $\sigma: \mathbf{Tm}_{\mathcal{L}} \rightarrow \mathbf{Tm}_{\mathcal{L}}$. But since $h_{\mathcal{Q}}(\sigma(x)) = g(x)$ for each variable x , it follows that $h_{\mathcal{Q}} \circ \sigma = g$. I.e., $\Gamma \subseteq \ker(h_{\mathcal{Q}} \circ \sigma)$. So for each $s' \approx t' \in \Gamma$, also $h_{\mathcal{Q}}(\sigma(s')) = h_{\mathcal{Q}}(\sigma(t'))$ and, by (II), $\models_{\mathcal{Q}} \sigma(s') \approx \sigma(t')$. Hence, by assumption, $\models_{\mathcal{Q}} \sigma(s) \approx \sigma(t)$ for some $s \approx t \in \Delta$. Therefore, again by (II), $g(s) = h_{\mathcal{Q}}(\sigma(s)) = h_{\mathcal{Q}}(\sigma(t)) = g(t)$ as required.

(\Leftarrow) Suppose that $\Gamma \models_{\mathbf{F}_{\mathcal{Q}}(\omega)} \Delta$ and let $\sigma: \mathbf{Tm}_{\mathcal{L}} \rightarrow \mathbf{Tm}_{\mathcal{L}}$ be a homomorphism such that $\models_{\mathcal{Q}} \sigma(s') \approx \sigma(t')$ for all $s' \approx t' \in \Gamma$. But then also $\sigma(\Gamma) \models_{\mathbf{F}_{\mathcal{Q}}(\omega)} \sigma(\Delta)$, so $\models_{\mathbf{F}_{\mathcal{Q}}(\omega)} \sigma(\Delta)$. By (B), $\models_{\mathcal{Q}} \sigma(s) \approx \sigma(t)$ for some $s \approx t \in \Delta$ as required. \square

Example 1. The variety \mathcal{KA} of Kleene algebras is generated as a quasivariety by the three-element algebra $\mathbf{C}_3 = \langle \{\perp, a, \top\}, \wedge, \vee, \neg, \perp, \top \rangle$ where $\perp < a < \top$ and \neg swaps \perp and \top , fixing a ; that is, $\mathcal{KA} = \mathbb{Q}(\mathbf{C}_3)$. Since no term is constantly a , the quasiequation

$$\{x \approx \neg x\} \Rightarrow x \approx y$$

is \mathcal{KA} -admissible, but $\{x \approx \neg x\} \not\models_{\mathcal{KA}} x \approx y$ (just consider mapping x to a and y to \perp). Indeed, it is shown in [5] (using a natural duality for \mathcal{KA}) that the admissible quasiequations of \mathcal{KA} are axiomatized relative to \mathcal{KA} by

$$\{\neg x \leq x, x \wedge \neg y \leq \neg x \vee y\} \Rightarrow \neg y \leq y. \quad (4)$$

That is, $\mathbb{Q}(\mathbf{F}_{\mathcal{KA}}(\omega))$ consists of all algebras in \mathcal{KA} satisfying (4). An axiomatization of the admissible clauses of \mathcal{KA} is also obtained in [5] by adding the “disjunction property”

$$\{x \vee y \approx \top\} \Rightarrow \{x \approx \top, y \approx \top\}. \quad (5)$$

That is, $\mathbb{U}(\mathbf{F}_{\mathcal{KA}}(\omega))$ consists of all algebras in \mathcal{KA} satisfying (5).

There is, however, another natural notion of admissibility (notion (A) from the introduction), which in logical contexts may be expressed as the property that adding the rule to a consequence relation (i.e., considering the smallest consequence relation containing both the rule and the original consequence relation) does not change the set of theorems. Expressed algebraically, this characterization corresponds to the following well-known equivalence (see, e.g., [26]):

Lemma 2. $\Gamma \Rightarrow s \approx t$ is \mathcal{Q} -admissible iff $\mathbb{V}(\mathcal{Q}) = \mathbb{V}(\{\mathbf{A} \in \mathcal{Q} \mid \Gamma \models_{\mathbf{A}} s \approx t\})$.

Example 2. The following quasiequation is admissible but not valid in the variety of Heyting algebras \mathcal{HA} :

$$\{\top \approx \neg x \rightarrow (y \vee z)\} \Rightarrow \top \approx (\neg x \rightarrow y) \vee (\neg x \rightarrow z). \quad (6)$$

Hence \mathcal{HA} is generated as a variety (but clearly not as a quasivariety) by all Heyting algebras satisfying (6). Consider now also the variety of Gödel algebras \mathcal{GA} : Heyting algebras satisfying $\top \approx (x \rightarrow y) \vee (y \rightarrow x)$. Validity and admissibility in \mathcal{GA} coincide; that is, $\mathcal{GA} = \mathbb{Q}(\mathbf{F}_{\mathcal{GA}}(\omega))$ (\mathcal{GA} is said to be *structurally complete*) and indeed, $\mathcal{GA} = \mathbb{U}(\mathbf{F}_{\mathcal{GA}}(\omega))$ (\mathcal{GA} is *universally complete*). Consider, however, the class $\mathcal{GA}_{\text{lin}}$ of Gödel algebras satisfying the positive clause.

$$\Rightarrow \{x \leq y, y \leq x\}$$

Then $\mathcal{GA}_{\text{lin}}$, the class of linearly ordered Gödel algebras, generates \mathcal{GA} as a variety, i.e., $\mathbb{V}(\mathcal{GA}_{\text{lin}}) = \mathcal{GA}$. However, this clause does not hold in the free algebra $\mathbf{F}_{\mathcal{GA}}(\omega)$.

As the previous example shows, the equivalence established in Lemma 2 does not extend to clauses. Rather the added clause should not increase the set of valid positive clauses.

Lemma 3. $\Gamma \Rightarrow \Delta$ is \mathcal{Q} -admissible iff $\mathbb{U}^+(\mathcal{Q}) = \mathbb{U}^+(\{\mathbf{A} \in \mathcal{Q} \mid \Gamma \models_{\mathbf{A}} \Delta\})$.

Proof. (\Rightarrow) Suppose that $\Gamma \Rightarrow \Delta$ is \mathcal{Q} -admissible. Then since for any quasivariety \mathcal{Q} , we have $\mathbf{F}_{\mathcal{Q}}(\omega) \in \mathcal{Q}$, it follows by Lemma 1 that $\mathbf{F}_{\mathcal{Q}}(\omega) \in \{\mathbf{A} \in \mathcal{Q} \mid \Gamma \models_{\mathbf{A}} \Delta\}$ and $\mathbb{U}^+(\mathbf{F}_{\mathcal{Q}}(\omega)) \subseteq \mathbb{U}^+(\{\mathbf{A} \in \mathcal{Q} \mid \Gamma \models_{\mathbf{A}} \Delta\}) \subseteq \mathbb{U}^+(\mathcal{Q})$. But also $\mathbb{U}^+(\mathcal{Q}) = \mathbb{U}^+(\mathbf{F}_{\mathcal{Q}}(\omega))$, using (3). Hence $\mathbb{U}^+(\mathcal{Q}) = \mathbb{U}^+(\{\mathbf{A} \in \mathcal{Q} \mid \Gamma \models_{\mathbf{A}} \Delta\})$.

(\Leftarrow) Suppose that $\mathbb{U}^+(\mathcal{Q}) = \mathbb{U}^+(\{\mathbf{A} \in \mathcal{Q} \mid \Gamma \models_{\mathbf{A}} \Delta\})$ and let $\sigma: \mathbf{Tm}_{\mathcal{L}} \rightarrow \mathbf{Tm}_{\mathcal{L}}$ be a homomorphism such that $\models_{\mathcal{Q}} \sigma(s') \approx \sigma(t')$ for all $s' \approx t' \in \Gamma$. Then $\models_{\mathbf{A}} \sigma(\Delta)$ for all $\mathbf{A} \in \mathcal{Q}$ such that $\Gamma \models_{\mathbf{A}} \Delta$. Hence also $\models_{\mathcal{Q}} \sigma(\Delta)$. By (3), $\models_{\mathcal{Q}} \sigma(s) \approx \sigma(t)$ for some $s \approx t \in \Delta$ as required. \square

Note that Lemma 2 follows directly from this result, since $\mathbb{U}^+(\mathcal{Q}) = \mathbb{V}(\mathcal{Q})$, and when $|\Delta| = 1$, also $\mathbb{U}^+(\{\mathbf{A} \in \mathcal{Q} \mid \Gamma \models_{\mathbf{A}} \Delta\}) = \mathbb{V}(\{\mathbf{A} \in \mathcal{Q} \mid \Gamma \models_{\mathbf{A}} \Delta\})$.

However, this raises the question as to what it means algebraically for a positive clause to “preserve” the set of valid equations, but perhaps not the set of valid positive clauses. An answer is provided below for congruence distributive varieties that makes use of Jónsson’s Lemma (referring to [4] for this result and other undefined concepts from Universal Algebra). Indeed in this case all valid quasiequations are preserved.

Lemma 4. If \mathcal{V} is a congruence distributive variety, then the following are equivalent:

- (1) $\models_{\mathbf{A}} \Delta$ for all subdirectly irreducible algebras $\mathbf{A} \in \mathcal{V}$
- (2) $\mathcal{V} = \mathbb{Q}(\{\mathbf{A} \in \mathcal{V} \mid \models_{\mathbf{A}} \Delta\})$
- (3) $\mathcal{V} = \mathbb{V}(\{\mathbf{A} \in \mathcal{V} \mid \models_{\mathbf{A}} \Delta\})$.

Proof. (1) \Rightarrow (2) Follows immediately from the fact that \mathcal{V} is generated as a quasivariety by its subdirectly irreducible members.

(2) \Rightarrow (3) Trivial since \mathcal{V} is a variety.

(3) \Rightarrow (1) Suppose that $\mathcal{V} = \mathbb{V}(\{\mathbf{A} \in \mathcal{V} \mid \models_{\mathbf{A}} \Delta\})$ and consider a subdirectly irreducible algebra $\mathbf{B} \in \mathcal{V}$. By Jónsson’s Lemma, $\mathbf{B} \in \mathbb{U}^+(\{\mathbf{A} \in \mathcal{V} \mid \models_{\mathbf{A}} \Delta\})$. Hence $\models_{\mathbf{B}} \Delta$ as required. \square

This result of course raises further questions. For example, what does it mean algebraically for a clause to “preserve” the set of valid equations (or quasiequations), but perhaps not the set of valid positive clauses?

Finally, for this section, observe that there are interesting and useful rules employed in the literature that do not seem to have a direct algebraic interpretation. Consider, for example, the following “density rule” for the variety \mathcal{GA}

of Gödel algebras, introduced by Takeuti and Titani in the logical setting of first-order Gödel logic [27]:

$$\{\top \approx (s \rightarrow x) \vee (x \rightarrow t) \vee u\} \Rightarrow \top \approx (s \rightarrow t) \vee u$$

where s , t , and u are terms not containing the variable x .

This rule was shown to be admissible for GA in [2] in the following sense: equations that can be derived in a proof system for GA extended with the rule can also be derived in the proof system without the rule. This admissibility result was extended to other classes of algebras in [22,7] and used to show that these classes are generated as quasivarieties by their linearly and densely ordered members. We will have more to say on this issue in Section 4.

3 A First-Order Framework

As the results and remarks of the previous section make plain, our two different notions of admissibility coincide in some but not all contexts. The goal of this section is to explain and explore these differences by considering admissibility in the more expressive setting of first-order logic.

We will assume the usual terminology and definitions of classical first-order logic with equality, making use of the symbols \forall , \exists , \sqcap , \sqcup , \Rightarrow , \sim , and \approx . In particular, for a first-order language \mathcal{L} , $\text{Sen}(\mathcal{L})$ is the set of sentences of \mathcal{L} with respect to a countably infinite set of variables, denoting formulas by φ, ψ and sets of formulas by Σ, Θ . For a class of \mathcal{L} -structures \mathcal{K} and $\Sigma \subseteq \text{Sen}(\mathcal{L})$, we set

$$\text{Th}_\Sigma(\mathcal{K}) = \{\psi \in \Sigma \mid \mathcal{K} \models \psi\}$$

and say that $\varphi \in \text{Sen}(\mathcal{L})$ *preserves* Σ in \mathcal{K} if

$$\text{Th}_\Sigma(\mathcal{K}) = \text{Th}_\Sigma(\{\mathbf{A} \in \mathcal{K} \mid \mathbf{A} \models \varphi\}).$$

In particular, if \mathcal{K} is axiomatized by $\Theta \subseteq \text{Sen}(\mathcal{L})$, then $\varphi \in \text{Sen}(\mathcal{L})$ preserves Σ in \mathcal{K} if for all $\psi \in \Sigma$:

$$\Theta \models \psi \quad \text{iff} \quad \Theta \cup \{\varphi\} \models \psi.$$

Let us again consider an algebraic language \mathcal{L} , and denote the set of \mathcal{L} -clauses (understood now as first-order sentences) by $\text{Cl}(\mathcal{L})$, positive \mathcal{L} -clauses by $\text{Cl}^+(\mathcal{L})$, \mathcal{L} -quasiequations by $\text{Qe}(\mathcal{L})$, and \mathcal{L} -equations by $\text{Eq}(\mathcal{L})$. We recall that for classes of \mathcal{L} -algebras \mathcal{K}_1 and \mathcal{K}_2 , $\mathbb{V}(\mathcal{K}_1) = \mathbb{V}(\mathcal{K}_2)$ iff \mathcal{K}_1 and \mathcal{K}_2 satisfy the same \mathcal{L} -equations, $\mathbb{Q}(\mathcal{K}_1) = \mathbb{Q}(\mathcal{K}_2)$ iff they satisfy the same \mathcal{L} -quasiequations, and $\mathbb{U}^+(\mathcal{K}_1) = \mathbb{U}^+(\mathcal{K}_2)$ iff they satisfy the same positive \mathcal{L} -clauses. Hence Lemmas 3 and 4 may be reinterpreted as:

Corollary 1. *If \mathcal{Q} is a quasivariety, then the following are equivalent for any clause $\varphi \in \text{Cl}(\mathcal{L})$:*

(1) $\mathbf{F}_{\mathcal{Q}}(\omega) \models \varphi$

- (2) $\mathbb{U}^+(\mathcal{Q}) = \mathbb{U}^+(\{\mathbf{A} \in \mathcal{Q} \mid \mathbf{A} \models \varphi\})$
 (3) φ preserves $\text{Cl}^+(\mathcal{L})$ in \mathcal{Q} .

In particular, if $\varphi \in \text{Qe}(\mathcal{L})$, then (1)-(3) are equivalent also to:

- (2') $\mathbb{V}(\mathcal{Q}) = \mathbb{V}(\{\mathbf{A} \in \mathcal{Q} \mid \mathbf{A} \models \varphi\})$
 (3') φ preserves $\text{Eq}(\mathcal{L})$ in \mathcal{Q} .

Corollary 2. *If \mathcal{V} is a congruence distributive variety, then the following are equivalent for any $\varphi \in \text{Cl}^+(\mathcal{L})$:*

- (1) $\mathbf{A} \models \varphi$ for all subdirectly irreducible algebras $\mathbf{A} \in \mathcal{V}$
 (2) $\mathcal{V} = \mathbb{Q}(\{\mathbf{A} \in \mathcal{V} \mid \mathbf{A} \models \varphi\})$
 (3) φ preserves $\text{Qe}(\mathcal{L})$ in \mathcal{V}
 (4) $\mathcal{V} = \mathbb{V}(\{\mathbf{A} \in \mathcal{V} \mid \mathbf{A} \models \varphi\})$
 (5) φ preserves $\text{Eq}(\mathcal{L})$ in \mathcal{V} .

Example 3. Consider the variety \mathcal{BA} of Boolean algebras in a language $\mathcal{L}_{\text{Bool}}$ and the $\mathcal{L}_{\text{Bool}}$ -sentence:

$$\varphi = (\forall x)((x \approx \perp) \sqcup (x \approx \top)).$$

Then φ preserves $\text{Qe}(\mathcal{L}_{\text{Bool}})$ in \mathcal{BA} , since a quasiequation is valid in all Boolean algebras iff it is valid in the standard two element Boolean algebra. Clearly, however, $\mathbf{F}_{\mathcal{BA}} \not\models \varphi$. On the other hand, $\neg\varphi$ also preserves $\text{Qe}(\mathcal{L}_{\text{Bool}})$ in \mathcal{BA} , since a quasiequation is valid in all Boolean algebras iff it is valid in the four element Boolean algebra.

An \mathcal{L} -sentence can be translated (in the standard way) into a set of clauses in an expanded language: first find an equivalent \mathcal{L} -sentence in prenex normal form, then skolemize to obtain a universal sentence, possibly containing extra function symbols, equivalent to a conjunction of clauses. In particular, consider again an algebraic language \mathcal{L} and a prenex formula $\varphi \in \text{Sen}(\mathcal{L})$. The *Skolem form* $\text{sk}(\varphi) \in \text{Sen}(\mathcal{L}')$ of φ in an algebraic language \mathcal{L}' extending \mathcal{L} with additional function symbols is defined in the usual way, so that for any $\Theta \cup \{\psi\} \subseteq \text{Sen}(\mathcal{L})$:

$$\Theta \cup \{\varphi\} \models \psi \quad \text{iff} \quad \Theta \cup \{\text{sk}(\varphi)\} \models \psi.$$

Let us fix \mathcal{K} to be an elementary class of \mathcal{L} -structures and, for any extension of the language \mathcal{L}' with additional function symbols, let \mathcal{K}' be the elementary class of \mathcal{L}' -structures whose \mathcal{L} -reducts are in \mathcal{K} .

Proposition 1. *The following are equivalent for any $\Sigma \cup \{\varphi\} \subseteq \text{Sen}(\mathcal{L})$:*

- (1) φ preserves Σ in \mathcal{K}
 (2) $\text{sk}(\varphi) \in \text{Sen}(\mathcal{L}')$ preserves Σ in \mathcal{K}' .

Proof. Suppose that \mathcal{K} is axiomatized by $\Theta \subseteq \text{Sen}(\mathcal{L})$. Then φ preserves Σ in \mathcal{K} iff for all $\psi \in \Sigma$:

$$\Theta \models \psi \quad \text{iff} \quad \Theta \cup \{\varphi\} \models \psi.$$

Axioms	Cut rule
$\frac{}{t \leq t} \text{ (ID)}$	$\frac{s \leq u \quad u \leq t}{s \leq t} \text{ (CUT)}$
Left logical rules	Right logical rules
$\frac{t_1 \leq s}{t_1 \wedge t_2 \leq s} \text{ } (\wedge \leq)_1$	$\frac{s \leq t_1}{s \leq t_1 \vee t_2} \text{ } (\leq \vee)_1$
$\frac{t_2 \leq s}{t_1 \wedge t_2 \leq s} \text{ } (\wedge \leq)_2$	$\frac{s \leq t_2}{s \leq t_1 \vee t_2} \text{ } (\leq \vee)_2$
$\frac{t_1 \leq s \quad t_2 \leq s}{t_1 \vee t_2 \leq s} \text{ } (\vee \leq)$	$\frac{s \leq t_1 \quad s \leq t_2}{s \leq t_1 \wedge t_2} \text{ } (\leq \wedge)$

Fig. 1. The proof system GLat

But this holds iff for all $\psi \in \Sigma$:

$$\Theta \models \psi \quad \text{iff} \quad \Theta \cup \{\text{sk}(\varphi)\} \models \psi.$$

That is, φ preserves Σ in \mathcal{K} iff $\text{sk}(\varphi) \in \text{Sen}(\mathcal{L}')$ preserves Σ in \mathcal{K}' . □

Example 4. Consider the variety of semilattices in a language \mathcal{L} with one binary connective \wedge , and the \mathcal{L} -sentence:

$$\varphi = (\forall x)(\forall y)(\exists z)(\forall w)((x \leq z) \wedge (y \leq z) \wedge (((x \leq w) \wedge (y \leq w)) \Rightarrow (z \leq w))).$$

Skolemizing, we obtain a language \mathcal{L}' with an additional binary connective \vee , and an \mathcal{L}' -sentence $\text{sk}(\varphi)$ of the form

$$(\forall x)(\forall y)(\forall w)((x \leq x \vee y) \wedge (y \leq x \vee y) \wedge (((x \leq w) \wedge (y \leq w)) \Rightarrow (x \vee y \leq w))).$$

Moreover, we may interpret $\text{sk}(\varphi)$ as the clauses:

$$\Rightarrow x \leq x \vee y, \quad \Rightarrow y \leq x \vee y, \quad \{x \leq w, y \leq w\} \Rightarrow x \vee y \leq w.$$

It is not hard to see that semilattices satisfying φ , are in fact lattices, and that φ preserves $\text{Eq}(\mathcal{L})$ in the variety of semilattices.

4 Applications

Let us turn our attention now to describing some applications of admissibility in determining properties of classes of algebras. More precisely, we use admissible rules to show that certain quasivarieties are generated as varieties or quasivarieties by their members satisfying order-theoretic properties such as boundedness,

unboundedness, linearity, and density. The goal here is not to provide striking new algebraic results, but rather to illustrate the potential of the methodology, leaving more general investigations for future work.

We begin with the variety $\mathcal{L}at$ of lattices in the language $\mathcal{L}_{\mathcal{L}at}$ with operation symbols \wedge and \vee , making use of the proof system $GLat$ of Fig. [1](#), the lattice fragment of the Full Lambek Calculus (see, e.g., [9](#) or [21](#)). For a given proof system GL , let us write $\vdash_{GL} W$ to denote that there exists a derivation in GL (a finite tree of structures built according to the rules of the system) with root W , and say that GL admits *cut-elimination* if there exists a procedure for transforming a derivation of W in GL into a derivation of W in GL that makes no use of the rule (CUT). (In fact, cut-elimination for GL implies that (CUT) understood as a quasiequation is admissible in the quasivariety defined by the other rules of GL .)

Theorem 1 (see, e.g., [21](#))

- (a) $\vdash_{GLat} s \leq t$ iff $\models_{\mathcal{L}at} s \leq t$.
- (b) $GLat$ admits cut-elimination.

Example 5. Note that by considering derivations in $GLat$ that do not make use of (CUT), we obtain

$$\begin{aligned} \models_{\mathcal{L}at} s_1 \wedge s_2 \leq t_1 \vee t_2 \quad \text{implies} \quad & \models_{\mathcal{L}at} s_1 \leq t_1 \vee t_2 \text{ or } \models_{\mathcal{L}at} s_2 \leq t_1 \vee t_2 \text{ or} \\ & \models_{\mathcal{L}at} s_1 \wedge s_2 \leq t_1 \text{ or } \models_{\mathcal{L}at} s_1 \wedge s_2 \leq t_2, \end{aligned}$$

and therefore the $\mathcal{L}at$ -admissibility of *Whitman's condition* (see [28](#))

$$\{x_1 \wedge x_2 \leq y_1 \vee y_2\} \Rightarrow \{x_1 \leq y_1 \vee y_2, x_2 \leq y_1 \vee y_2, x_1 \wedge x_2 \leq y_1, x_1 \wedge x_2 \leq y_2\}.$$

Let us consider now the following $\mathcal{L}_{\mathcal{L}at}$ -sentence for expressing boundedness:

$$\varphi_{bd} = (\exists x)(\exists y)(\forall z)((x \leq z) \cap (z \leq y)).$$

Skolemizing this sentence gives

$$sk(\varphi_{bd}) = (\forall z)((\perp \leq z) \cap (z \leq \top))$$

in the expanded language $\mathcal{L}_{\mathcal{L}at}^b$ containing additional constants \perp and \top .

Theorem 2. φ_{bd} preserves $\text{Eq}(\mathcal{L}_{\mathcal{L}at})$ in $\mathcal{L}at$.

Proof. It suffices by Proposition [1](#) to show that $sk(\varphi_{bd})$ preserves $\text{Eq}(\mathcal{L}_{\mathcal{L}at})$ in $\mathcal{L}at^b$ where $\mathcal{L}at^b$ consists of all lattices with additional constants \perp and \top . Let $\mathcal{B}\mathcal{L}at = \{\mathbf{A} \in \mathcal{L}at^b \mid \mathbf{A} \models \varphi_{bd}\}$. Then it is enough, using Corollary [1](#), to show that whenever $\models_{\mathcal{B}\mathcal{L}at} s \leq t$ for $s \leq t \in \text{Eq}(\mathcal{L}_{\mathcal{L}at})$, also $\models_{\mathcal{L}at^b} s \leq t$.

We define $GBLat$ to be $GLat$ extended with the rules:

$$\overline{\perp \leq t} \quad (\perp \leq) \quad \text{and} \quad \overline{s \leq \top} \quad (\leq \top).$$

Then it is easily shown that (a) $\vdash_{\text{GBLat}} s \leq t$ iff $\models_{\mathcal{B}\mathcal{L}\text{at}} s \leq t$, and (b) GBLat admits cut-elimination. Suppose that $\vdash_{\text{GBLat}} s \leq t$ for $s \leq t \in \text{Eq}(\mathcal{L}_{\mathcal{L}\text{at}})$. Then there is a cut-free derivation of $s \leq t$ in GBLat and hence also (since the extra rules cannot be used in this derivation) in GLat . I.e., $\vdash_{\text{GLat}} s \leq t$ and therefore $\models_{\mathcal{L}\text{at}^b} s \leq t$ as required. \square

Hence by Corollary [1](#), we obtain:

Corollary 3. $\mathcal{L}\text{at} = \mathbb{V}(\{\mathbf{A} \in \mathcal{L}\text{at} \mid \mathbf{A} \text{ is bounded}\})$.

We consider now the following $\mathcal{L}_{\mathcal{L}\text{at}}$ -sentence for expressing unboundedness:

$$\varphi_{\text{unbd}} = (\forall x)(\exists y)(\exists z)(\neg(x \leq y) \sqcap \neg(z \leq x)).$$

Skolemizing this sentence gives

$$\text{sk}(\varphi_{\text{unbd}}) = (\forall x)(\neg(x \leq \downarrow x) \sqcap \neg(\uparrow x \leq x))$$

in the expanded language $\mathcal{L}_{\mathcal{L}\text{at}}^u$ with additional unary function symbols \downarrow and \uparrow .

Theorem 3. φ_{unbd} preserves $\text{Eq}(\mathcal{L}_{\mathcal{L}\text{at}})$ in $\mathcal{L}\text{at}$.

Proof. It suffices by Proposition [1](#) to show that $\text{sk}(\varphi_{\text{unbd}})$ preserves $\text{Eq}(\mathcal{L}_{\mathcal{L}\text{at}})$ in $\mathcal{L}\text{at}^u$ where $\mathcal{L}\text{at}^u$ consists of all lattices with additional unary functions \downarrow and \uparrow . Let $\mathcal{U}\mathcal{L}\text{at} = \{\mathbf{A} \in \mathcal{L}\text{at}^u \mid \mathbf{A} \models \varphi_{\text{unbd}}\}$. Then it is enough to show that whenever $\models_{\mathcal{U}\mathcal{L}\text{at}} s \leq t$ for $s \leq t \in \text{Eq}(\mathcal{L}_{\mathcal{L}\text{at}})$, also $\models_{\mathcal{L}\text{at}^u} s \leq t$.

We define GULat to be GLat extended with the rules:

$$\frac{u \leq \downarrow u}{s \leq t} (\leq\downarrow) \quad \text{and} \quad \frac{\uparrow u \leq u}{s \leq t} (\uparrow\leq).$$

Then it is easily shown that (a) $\vdash_{\text{GULat}} s \leq t$ iff $\models_{\mathcal{U}\mathcal{L}\text{at}} s \leq t$, GULat admits cut-elimination, and (c) $\not\vdash_{\text{GULat}} s \leq \downarrow t$ and $\not\vdash_{\text{GULat}} \uparrow s \leq t$ for all $\mathcal{L}_{\mathcal{L}\text{at}}^u$ -terms s, t . Hence, if $\vdash_{\text{GULat}} s \leq t$ for $s \leq t \in \text{Eq}(\mathcal{L}_{\mathcal{L}\text{at}})$, then there is a derivation of $s \leq t$ in GLat as required. \square

Hence by Corollary [1](#), we obtain:

Corollary 4. $\mathcal{L}\text{at} = \mathbb{V}(\{\mathbf{A} \in \mathcal{L}\text{at} \mid \mathbf{A} \text{ is unbounded}\})$.

Note that although these generation results for lattices are straightforward to prove algebraically, for other classes of algebras this may no longer be the case. In particular, we aim to use this methodology to provide general conditions for classes of algebras to be generated as a variety by its bounded or unbounded members. Note for example, that the variety of lattice-ordered abelian groups cannot be generated by its bounded members (since there is only one, the trivial algebra), and that a variety of commutative residuated lattices (see below) satisfying weakening conditions such as $x \cdot y \leq x$ cannot be generated by its unbounded members.

Let us conclude by considering the more interesting case of the density rule already mentioned at the end of Section 2. A *commutative residuated lattice* for the language \mathcal{L} is an algebra $\mathbf{A} = \langle A, \wedge, \vee, \cdot, \rightarrow, e \rangle$ such that $\langle A, \wedge, \vee \rangle$ is a lattice, $\langle A, \cdot, e \rangle$ is a commutative monoid, and $x \cdot y \leq z$ iff $x \leq y \rightarrow z$ for all $x, y, z \in A$. The algebra \mathbf{A} is called *semilinear* if it is distributive and satisfies $e \leq (x \rightarrow y) \vee (y \rightarrow x)$ for all $x, y \in A$. We denote the class of all semilinear commutative residuated lattices by \mathcal{CRL}^c .

A proof system GCRL^c for \mathcal{CRL}^c , originally defined with extra constants in [22] (see also [23]), is presented in Fig. 2 in the framework of hypersequents (introduced by Avron in [1]). A (*single-conclusion*) *sequent* S is an ordered pair consisting of a finite multiset of \mathcal{L} -terms Π and a term t , written $\Pi \leq t$. A (*single-conclusion*) *hypersequent* \mathcal{G} is a finite multiset of sequents, written $S_1 \mid \dots \mid S_n$. We define the following interpretation of sequents and hypersequents:

$$\begin{aligned} i(s_1, \dots, s_m \leq t) &= (s_1 \cdot \dots \cdot s_m) \rightarrow t \\ i(\leq t) &= t \\ i(S_1 \mid \dots \mid S_n) &= i(S_1) \vee \dots \vee i(S_n) \end{aligned}$$

and write $\models_{\mathcal{CRL}^c} \mathcal{G}$ iff $\models_{\mathcal{CRL}^c} e \leq i(\mathcal{G})$.

Theorem 4 (see [22], also [23])

- (a) $\vdash_{\text{GCRL}^c} \mathcal{G}$ iff $\models_{\mathcal{CRL}^c} \mathcal{G}$.
- (b) GCRL^c admits cut-elimination.

Consider now the following \mathcal{L} -sentence expressing linearity and density:

$$\varphi = (\forall x)(\forall y)(\exists z)((x \leq y) \sqcup (y \leq x)) \sqcap (((x \leq z) \sqcup (z \leq y)) \Rightarrow (x \leq y)).$$

Skolemizing, we obtain the sentence

$$\text{sk}(\varphi) = (\forall x)(\forall y)((x \leq y) \sqcup (y \leq x)) \sqcap (((x \leq d(x, y)) \sqcup (d(x, y) \leq y)) \Rightarrow (x \leq y)).$$

in an expanded language \mathcal{L}^d containing an additional binary function symbol d .

Theorem 5. φ preserves $\text{Eq}(\mathcal{L})$ in \mathcal{CRL}^c .

Proof. It suffices by Proposition 1 to show that $\text{sk}(\varphi)$ preserves $\text{Eq}(\mathcal{L})$ in $\mathcal{CRL}^{c'}$ where $\mathcal{CRL}^{c'}$ consists of all semilinear commutative residuated lattices with an additional binary function d . Let $\mathcal{CRL}^{cD} = \{\mathbf{A} \in \mathcal{CRL}^{c'} \mid \mathbf{A} \models \varphi\}$. Then it is enough to show that whenever $\models_{\mathcal{CRL}^{cD}} s \leq t$ for $s \leq t \in \text{Eq}(\mathcal{L})$, also $\models_{\mathcal{CRL}^{c'}} s \leq t$.

We define GCRL^{cD} to be GCRL^c extended with the rule:

$$\frac{\mathcal{G} \mid \Pi_1 \leq x \mid \Pi_2, x \leq t}{\mathcal{G} \mid \Pi_1, \Pi_2 \leq t} \text{ (DENSITY)}$$

where x does not occur in \mathcal{G} , Π_1 , Π_2 , or t .

It is proved in [22] (see also [23]) that (a) $\vdash_{\text{GCRL}^{cD}} \mathcal{G}$ iff $\models_{\mathcal{CRL}^{cD}} \mathcal{G}$, and (b) for all \mathcal{G} not containing d , $\vdash_{\text{GCRL}^{cD}} \mathcal{G}$ iff $\vdash_{\text{GCRL}^c} \mathcal{G}$. Hence if $\models_{\mathcal{CRL}^{cD}} s \leq t$ for $s \leq t \in \text{Eq}(\mathcal{L})$, then $\vdash_{\text{GCRL}^{cD}} s \leq t$ and so $\vdash_{\text{GCRL}^c} s \leq t$ as required. \square

<p>Axioms</p> $\frac{}{\mathcal{G} \mid t \leq t} \text{ (ID)}$ <p>External weakening</p> $\frac{\mathcal{G}}{\mathcal{G} \mid \mathcal{H}} \text{ (EW)}$ <p>Left logical rules</p> $\frac{\mathcal{G} \mid \Pi \leq u}{\mathcal{G} \mid \Pi, e \leq u} \text{ (e}\leq\text{)}$ $\frac{\mathcal{G} \mid \Pi_1 \leq s \quad \mathcal{G} \mid \Pi_2, t \leq u}{\mathcal{G} \mid \Pi_1, \Pi_2, s \rightarrow t \leq u} \text{ (}\rightarrow\leq\text{)}$ $\frac{\mathcal{G} \mid \Pi, s, t \leq u}{\mathcal{G} \mid \Pi, s \cdot t \leq u} \text{ (}\cdot\leq\text{)}$ $\frac{\mathcal{G} \mid \Pi, s \leq u}{\mathcal{G} \mid \Pi, s \wedge t \leq u} \text{ (}\wedge\leq\text{)}_1$ $\frac{\mathcal{G} \mid \Pi, t \leq u}{\mathcal{G} \mid \Pi, s \wedge t \leq u} \text{ (}\wedge\leq\text{)}_2$ $\frac{\mathcal{G} \mid \Pi, s \leq u \quad \mathcal{G} \mid \Pi, t \leq u}{\mathcal{G} \mid \Pi, s \vee t \leq u} \text{ (}\vee\leq\text{)}$	<p>Cut rule</p> $\frac{\mathcal{G} \mid \Pi_1, t \leq u \quad \mathcal{G} \mid \Pi_2 \leq t}{\mathcal{G} \mid \Pi_1, \Pi_2 \leq u} \text{ (CUT)}$ <p>External contraction</p> $\frac{\mathcal{G} \mid \mathcal{H} \mid \mathcal{H}}{\mathcal{G} \mid \mathcal{H}} \text{ (EC)}$ <p>Right logical rules</p> $\frac{}{\mathcal{G} \mid \leq e} \text{ (}\leq e\text{)}$ $\frac{\mathcal{G} \mid \Pi, s \leq t}{\mathcal{G} \mid \Pi \leq s \rightarrow t} \text{ (}\leq\rightarrow\text{)}$ $\frac{\mathcal{G} \mid \Pi_1 \leq s \quad \mathcal{G} \mid \Pi_2 \leq t}{\mathcal{G} \mid \Pi_1, \Pi_2 \leq s \cdot t} \text{ (}\leq\cdot\text{)}$ $\frac{\mathcal{G} \mid \Pi \leq s}{\mathcal{G} \mid \Pi \leq s \vee t} \text{ (}\leq\vee\text{)}_1$ $\frac{\mathcal{G} \mid \Pi \leq t}{\mathcal{G} \mid \Pi \leq s \vee t} \text{ (}\leq\vee\text{)}_2$ $\frac{\mathcal{G} \mid \Pi \leq s \quad \mathcal{G} \mid \Pi \leq t}{\mathcal{G} \mid \Pi \leq s \wedge t} \text{ (}\leq\wedge\text{)}$
---	--

Fig. 2. GCRL^c

Hence by Corollary [11](#), we obtain:

Corollary 5. $\mathcal{CRL}^c = \mathbb{V}(\{\mathbf{A} \in \mathcal{Lat} \mid \mathbf{A} \text{ is linearly and densely ordered}\})$.

Note finally that in [\[22\]](#) (see also [\[23\]](#)) it is proved that \mathcal{CRL}^c is generated as a quasivariety by its linearly and densely ordered members. This follows here from the previous corollary using the fact that a suitable local deduction theorem holds for both classes of algebras.

Acknowledgements. I would like to thank both Leonardo Cabrer and Christoph Röthlisberger for their helpful comments on the work reported here.

References

1. Avron, A.: A constructive analysis of RM. *Journal of Symbolic Logic* 52(4), 939–951 (1987)

2. Baaz, M., Zach, R.: Hypersequents and the Proof Theory of Intuitionistic Fuzzy Logic. In: Clote, P.G., Schwichtenberg, H. (eds.) CSL 2000. LNCS, vol. 1862, pp. 187–201. Springer, Heidelberg (2000)
3. Babenyshev, S., Rybakov, V., Schmidt, R.A., Tishkovsky, D.: A tableau method for checking rule admissibility in S4. In: Proceedings of UNIF 2009. ENTCS, vol. 262, pp. 17–32 (2010)
4. Burris, S., Sankappanavar, H.P.: A Course in Universal Algebra. Graduate Texts in Mathematics, vol. 78. Springer, New York (1981)
5. Cabrer, L.M., Metcalfe, G.: Admissibility via unifiability (preprint)
6. Chang, C., Keisler, H.: Model Theory. Studies in Logic and the Foundations of Mathematics, vol. 73. Elsevier (1977)
7. Ciabattoni, A., Metcalfe, G.: Density elimination. Theoretical Computer Science 403, 328–346 (2008)
8. Cintula, P., Metcalfe, G.: Admissible rules in the implication-negation fragment of intuitionistic logic. Annals of Pure and Applied Logic 162(10), 162–171 (2010)
9. Galatos, N., Jipsen, P., Kowalski, T., Ono, H.: Residuated Lattices: An Algebraic Glimpse at Substructural Logics. Elsevier (2007)
10. Gentzen, G.: Untersuchungen über das Logische Schliessen. Math. Zeitschrift 210, 176–210, 405–431 (1935)
11. Ghilardi, S.: Unification in intuitionistic logic. Journal of Symbolic Logic 64(2), 859–880 (1999)
12. Ghilardi, S.: Best solving modal equations. Annals of Pure and Applied Logic 102(3), 184–198 (2000)
13. Ghilardi, S.: A resolution/tableaux algorithm for projective approximations in IPC. Logic Journal of the IGPL 10(3), 227–241 (2002)
14. Iemhoff, R.: On the admissible rules of intuitionistic propositional logic. Journal of Symbolic Logic 66(1), 281–294 (2001)
15. Iemhoff, R.: Intermediate logics and Visser’s rules. Notre Dame Journal of Formal Logic 46(1), 65–81 (2005)
16. Iemhoff, R., Metcalfe, G.: Proof theory for admissible rules. Annals of Pure and Applied Logic 159(1-2), 171–186 (2009)
17. Jeřábek, E.: Admissible rules of modal logics. Journal of Logic and Computation 15, 411–431 (2005)
18. Jeřábek, E.: Admissible rules of Łukasiewicz logic. Journal of Logic and Computation 20(2), 425–447 (2010)
19. Jeřábek, E.: Bases of admissible rules of Łukasiewicz logic. Journal of Logic and Computation 20(6), 1149–1163 (2010)
20. Lorenzen, P.: Einführung in die operative Logik und Mathematik. Grundlehren der mathematischen Wissenschaften, vol. 78. Springer (1955)
21. Metcalfe, G.: Proof theory of mathematical fuzzy logic. In: Handbook of Mathematical Fuzzy Logic, ch. 3, vol. I, pp. 209–282. King’s College Publications (2011)
22. Metcalfe, G., Montagna, F.: Substructural fuzzy logics. Journal of Symbolic Logic 72(3), 834–864 (2007)
23. Metcalfe, G., Olivetti, N., Gabbay, D.: Proof Theory for Fuzzy Logics. Springer (2008)
24. Metcalfe, G., Röthlisberger, C.: Unifiability and Admissibility in Finite Algebras. In: Cooper, S.B., Dawar, A., Löwe, B. (eds.) CiE 2012. LNCS, vol. 7318, pp. 485–495. Springer, Heidelberg (2012)

25. Rozière, P.: Admissible and derivable rules in intuitionistic logic. *Mathematical Structures in Computer Science* 2(3), 129–136 (1993)
26. Rybakov, V.: Admissibility of Logical Inference Rules. *Studies in Logic and the Foundations of Mathematics*, vol. 136. Elsevier, Amsterdam (1997)
27. Takeuti, G., Titani, T.: Intuitionistic fuzzy logic and intuitionistic fuzzy set theory. *Journal of Symbolic Logic* 49(3), 851–866 (1984)
28. Whitman, P.: Free lattices. *Annals of Mathematics* 42, 325–329 (1941)

On Distributed Monitoring of Asynchronous Systems

Volker Diekert¹ and Anca Muscholl²

¹ Universität Stuttgart, FMI, Germany

² LaBRI, Univ. of Bordeaux, France

1 Introduction

Distributed systems are notoriously difficult to understand and analyze in order to assert their correction w.r.t. given properties. They often exhibit a huge number of different behaviors, as soon as the active entities (peers, agents, processes, . . .) behave in an asynchronous manner. Already the modelization of such systems is a non-trivial task, let alone their formal verification.

Several automata-based distributed models have been proposed and studied over the past twenty years, capturing various aspects of distributed behavior. Depending on the motivation, such models fall into two large categories. In the first one we find rather simple models, expressing basic synchronization mechanisms, like Petri nets or communicating automata. In the second category we see more sophisticated models, conceived for supporting practical system design, like statecharts or I/O automata. It is clear that being able to develop automated verification techniques requires a good understanding of the simpler models, in particular since more complex ones are often built as a combination of basic models.

This purpose of this paper is to discuss the problem of distributed monitoring on a simple model of finite-state distributed automata based on shared actions, called *asynchronous automata*. Monitoring is a question related to runtime verification: assume that we have to check a property L against an unknown or very complex system \mathcal{A} , so that classical static analysis is not possible. Therefore instead of model-checking a *monitor* is used, that checks the property on the underlying system at runtime. The question is which properties can be checked in this way, that is, which properties L are *monitorable*. A classical example for monitorable properties are safety properties, like “no alarm is raised”. A monitor for a property L is an automaton \mathcal{M}_L that after each finite execution tells whether (1) every possible extension of the execution is in L , or (2) every possible extension is in the complement of L , or neither (1) nor (2) holds. The notion of monitorable properties has been proposed by Pnueli and Zaks [15], and the theory has been extended to various kinds of systems, for instance to probabilistic systems [3,10] or real-time systems [12].

We are interested here in monitoring distributed systems modelled as asynchronous automata. It is natural to require that monitors should be of the same kind as the underlying system, so we consider here distributed monitoring. A distributed monitor does not have a global view of the system, therefore we propose

the notion of *locally monitorable* trace language. Our main result shows that if the distributed alphabet of actions is connected and if L is a set of Γ -infinite traces (for some subset of processes Γ) such that both L and its complement L^c are countable unions of locally safety languages, then L is locally monitorable. We also show that over Γ -infinite traces, recognizable countable unions of locally safety languages are precisely the complements of deterministic languages.

2 Preliminaries

The idea of describing concurrency by a fixed independence relation on a given set of actions Σ goes back to the late seventies, to Mazurkiewicz [12] and Keller [11] (see also [6]). One can start with a *distributed action alphabet* (Σ, dom) on a finite set $Proc$ of processes, where $dom : \Sigma \rightarrow (2^{Proc} \setminus \emptyset)$ is a *location function*. The location $dom(a)$ of action $a \in \Sigma$ comprises all processes that need to synchronize in order to perform this action. It defines in a natural way an *independence relation* $I \subseteq \Sigma \times \Sigma$ by letting $(a, b) \in I$ if and only if $dom(a) \cap dom(b) = \emptyset$.

The execution order of two independent actions $(a, b) \in I$ is irrelevant, they can be executed as a, b , or b, a - or even concurrently. More generally, we can consider the congruence \sim_I on Σ^* generated by I . An equivalence class $[w]_I$ of \sim_I is called a (finite) *Mazurkiewicz trace*, and it can be also viewed as labeled pomset $t = \langle V, \leq, \lambda \rangle$ of a special kind: if $w = a_0 \cdots a_n$ then the vertex set is $V = \{0, \dots, n\}$, the labeling function is $\lambda(i) = a_i$ and $\leq = (\{(i, j) \mid i < j, (a_i, a_j) \notin I\})^*$ is the partial order. The word w is a *linearization* of t defined as above, i.e., a total order compatible with the partial order of t .

Infinite traces can be defined in a similar way from ω -words. Finite and infinite traces are also called *real traces*, and the set of real traces is written $\mathbb{R}(\Sigma, I)$ (or simply \mathbb{R} when Σ, I are clear from the context). A trace t is a *prefix* of a trace t' (denotes as $t \leq t'$) if t is isomorphic to a downwards-closed subset of t' . The set of prefixes of t is denoted $pref(t)$. If $L \subseteq \mathbb{R}$ then we denote by $Lin(L) \subseteq \Sigma^\infty$ the set of linearizations of traces from L .

A language $K \subseteq \Sigma^\infty$ is called *trace-closed* if $K = Lin(L)$ for some $L \subseteq \mathbb{R}$. Whenever convenient, we talk about trace languages $L \subseteq \mathbb{R}$ or trace-closed word languages $K \subseteq \Sigma^\infty$ in equivalent terms. A language $L \subseteq \mathbb{R}$ is *recognizable* if $Lin(L) \subseteq \Sigma^\infty$ is a regular language of finite and infinite words.

Linear temporal properties like *safety* and *liveness* [14] can be translated into topological properties, as closed and dense sets in the Cantor topology. For real traces, these notions generalize smoothly to the Scott topology, by replacing word prefixes by trace prefixes. The Scott topology corresponds to a global view in traces, where one needs to reason on global configurations, i.e., configurations involving several processes. However, in the setting of monitoring that we discuss here, such a global view is not available. Therefore we use here *local safety* as basic notion, as introduced in [4] and explained in the following.

A trace $t = \langle V, \leq, \lambda \rangle$ is called *prime* if it is finite and has a unique maximal element. That is, $|\max(t)| = 1$, where $\max(t)$ is the set of maximal elements of t w.r.t. the partial order \leq . The set of prime traces in \mathbb{R} is denoted $\mathbb{P}(\mathbb{R})$. The set of prime prefixes of elements of $L \subseteq \mathbb{R}$ is denoted $\mathbb{P}(L)$.

Definition 1. Let $L \subseteq \mathbb{R}$.

1. L is called *prime-open* if it is of the form $\bigcup\{p\mathbb{R} \mid p \in U\}$ for some $U \subseteq \mathbb{P}$. Complements of prime-open sets are called *prime-closed*.
2. \bar{L} is the intersection of all prime-closed sets containing L (and denoted as *prime-closure* of L). Note that \bar{L} is prime-closed.
3. A prime-closed, recognizable language $L \subseteq \mathbb{R}$ is called a *locally safety language*.

Remark 1. 1. Every prime-open set is also Scott-open, and prime-open sets are closed under union, but not under intersection. As an example consider $a\mathbb{R} \cap b\mathbb{R}$ which is not prime-open for $(a, b) \in I$.

2. A first-order locally safety language $L \subseteq \mathbb{R}$ is a prime-closed set such that $\text{Lin}(L)$ is a first-order language. It is known from [4] that first-order locally safety languages are characterized by formulas of the form $G\psi$, with ψ a past formula in a local variant of LTL called LocTL.

We end this section by introducing our model for distributed automata. An *asynchronous automaton* $\mathcal{A} = \langle (S_\alpha)_{\alpha \in \text{Proc}}, s_{in}, (\delta_a)_{a \in \Sigma} \rangle$ is given by

- for every process α a finite set S_α of (local) states,
- the initial state $s_{in} \in \prod_{\alpha \in \text{Proc}} S_\alpha$,
- for every action $a \in \Sigma$ a transition relation $\delta_a \subseteq (\prod_{\alpha \in \text{dom}(a)} S_\alpha)^2$ on tuples of states of processes in $\text{dom}(a)$.

For convenience, we abbreviate a tuple $(s_\alpha)_{\alpha \in P}$ of local states by s_P , where $P \subseteq \text{Proc}$. We also denote $\prod_{\alpha \in \text{Proc}} S_\alpha$ as *global states* and $\prod_{\alpha \in P} S_\alpha$ as S_P .

An asynchronous automaton can be seen as a sequential automaton with the state set $S = \prod_{\alpha \in \text{Proc}} S_\alpha$ and transitions $s \xrightarrow{a} s'$ if $(s_{\text{dom}(a)}, s'_{\text{dom}(a)}) \in \delta_a$, and $s_{\text{Proc} \setminus \text{dom}(a)} = s'_{\text{Proc} \setminus \text{dom}(a)}$. By $\mathcal{L}(\mathcal{A})$ we denote the set of words labeling runs of this sequential automaton that start from the initial state. It can be easily noted that $\mathcal{L}(\mathcal{A})$ is trace-closed. The automaton is *deterministic* if each δ_a is a (partial) function.

Example 1. Let us consider the asynchronous automaton \mathcal{A} given by $S_p = \{0\}$, $S_q = S_r = \{0, 1\}$, and transition function $\delta_a(s_p, s_q) = (s_p, \neg s_q)$ if $s_q = 1$ (undefined otherwise), $\delta_d(s_r) = \neg s_r$ if $s_r = 1$ (undefined otherwise), $\delta_b(s_q, s_r) = (1, 1)$ if $s_q \wedge s_r = 0$ (undefined otherwise) and $\delta_c(s_p) = s_p$. Starting with $s_0 = (0, 0, 0)$, an accepting run of \mathcal{A} checks that between any two successive b -events, there is either an a or a d (or both), and there is a b -event before all a and d .

Since the notion of a trace was formulated without a reference to an accepting device, it is natural to ask if the model of asynchronous automata is powerful enough for capturing the notion of regularity. Zielonka's theorem below says that this is indeed the case, hence these automata are a right model for the simple view of concurrency captured by Mazurkiewicz traces.

Theorem 1. [17] Let $\text{dom} : \Sigma \rightarrow (2^{\text{Proc}} \setminus \{\emptyset\})$ be a distribution of letters. If a language $L \subseteq \Sigma^*$ is regular and trace-closed then there is a deterministic asynchronous automaton accepting L (of size exponential in the number of processes and polynomial in the size of the minimal automaton for L , see [9]).

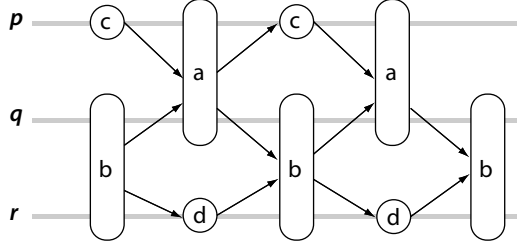


Fig. 1. The pomset associated with the trace $t = [c b a d c b a d b]$, with $\text{dom}(a) = \{p, q\}$, $\text{dom}(b) = \{q, r\}$, $\text{dom}(c) = \{p\}$, $\text{dom}(d) = \{r\}$

3 Safety Languages

A set of traces $C \subseteq \mathbb{R}$ is called *coherent* if $C \subseteq \text{pref}(t)$ for some $t \in \mathbb{R}$. This means that $\sqcup C \in \mathbb{R}$ exists, and it is a prefix of t . By L^c we denote the complement $\mathbb{R} \setminus L$ of L . Recall that $\mathbb{P}(L)$ is the set of prime prefixes of traces in $L \subseteq \mathbb{R}$.

We use in our characterizations below a basic property of automata on traces, which is for instance satisfied by (runs of) asynchronous automata, called *forward diamond property*. A set $K \subseteq \Sigma^*$ satisfies the forward diamond property if the following holds:

If $ua \in K$ and $ub \in K$, then $uab \in K$, for every $u \in \Sigma^*$ and $(a, b) \in I$.

Lemma 1. For $L \subseteq \mathbb{R}$ we have

$$\overline{L} = \{\sqcup C \mid C \subseteq \mathbb{P}(L) \text{ and } C \text{ is coherent}\}.$$

We have $\overline{L} = \overline{K}$ if and only if $\mathbb{P}(L) = \mathbb{P}(K)$.

Proof. Let $X = \{\sqcup C \mid C \subseteq \mathbb{P}(L) \text{ and } C \text{ is coherent}\}$. By definition, $X^c = U\mathbb{R}$ with $U = \mathbb{P} \setminus \mathbb{P}(L)$, thus X is prime-closed (and contains L). Let $K \supseteq L$ be prime-closed, thus $K^c = V\mathbb{R}$ with $V \subseteq \mathbb{P}$. Consider some coherent set $C \subseteq \mathbb{P}(L)$, and assume that $\sqcup C \in v\mathbb{R}$ for some $v \in V$. But then $v \in \mathbb{P}(L)$, thus $K^c \cap L \neq \emptyset$, a contradiction. So $X \subseteq K$, which shows that $\overline{L} = X$.

Lemma 2. If $L \subseteq \mathbb{R}$ is recognizable, then the prime closure \overline{L} is recognizable, too. Moreover, on input (Σ, dom) and (sequential) Büchi automaton \mathcal{B} such that $L = \mathcal{L}(\mathcal{B})$ is trace-closed, we can compute an exponential-size, deterministic asynchronous automaton \mathcal{A} accepting \overline{L} , such that all states of \mathcal{A} are final.

Proof. Given $L \subseteq \mathbb{R}$ recognizable, we have that $\mathbb{P}(L)$ is recognizable, too. Then it is easy to see that \overline{L} is recognizable, by using for instance monadic second-order logic over traces.

Let us consider the complexity of the construction of a deterministic asynchronous automaton for \overline{L} in more detail. We assume that the input L is given

by a (sequential) Büchi automaton \mathcal{B} . We first determinize \mathcal{B} and get a deterministic (say Rabin) automaton \mathcal{B}' for L . From \mathcal{B}' we can easily construct a DFA accepting $\mathbb{P}(L)$: we just need to store the set of maximal processes in the control state. The resulting DFA is exponential in both \mathcal{B} and $Proc$. By applying the construction cited in Thm. [1](#) we obtain a deterministic asynchronous automaton \mathcal{A} for $\mathbb{P}(L)$ which is still exponential in \mathcal{B} and $Proc$. Using classical timestamping we may assume that each local state reached by the maximal processes of a prime trace contains the complete information about the global state of \mathcal{A} reached on that prime trace - the size of the deterministic asynchronous automaton \mathcal{A}' thus obtained remains exponential. It remains to construct the automaton accepting \overline{L} . Recall that \overline{L} contains precisely those traces where all prime prefixes belong to $\mathbb{P}(L)$. Thus, it suffices to take \mathcal{A}' and forbid transitions that produce bad local states of \mathcal{A}' , that is, local states that are non-final viewed as global states of \mathcal{A} . On finite or infinite traces, the automaton \mathcal{A}' accepts precisely \overline{L} . By construction, all its reachable states are final.

Proposition 1. *The following are equivalent characterizations for $L \subseteq \mathbb{R}$:*

1. L is a locally-safety language.
2. $K = Lin(L) \subseteq \Sigma^\infty$ is a regular, prefix-closed language such that $K \cap \Sigma^\omega$ is a safety language, and $K \cap \Sigma^*$ satisfies the forward diamond condition.
3. L is accepted by a deterministic asynchronous automaton where all reachable states are final.

Proof. The implications (1) \Rightarrow (2) and (3) \Rightarrow (1) are immediate. For (2) \Rightarrow (3) let us assume that $K = Lin(L)$ is regular, prefix-closed and satisfies the two additional conditions in the statement. Since $K \cap \Sigma^*$ is prefix-closed, trace-closed and satisfies the forward diamond property, there exists a deterministic asynchronous automaton \mathcal{B} recognizing $K \cap \Sigma^*$ (equivalently, the set of finite traces in L) such that all reachable states are final [\[16\]](#). Since K is assumed to be prefix-closed and $K \cap \Sigma^\omega$ is a safety language, we obtain that the automaton \mathcal{B} accepts precisely $\overline{L} = L$ over \mathbb{R} .

Example 2. Assume that $\Sigma = \{a, b, c\}$ with $dom(a) = \{\alpha\}$, $dom(b) = \{\beta\}$ and $dom(c) = \{\alpha, \beta\}$. The trace language “no two consecutive c ’s” is a locally safety language, and it can be recognized by an asynchronous automaton where both processes remember their last action, and do not allow two consecutive c ’s.

The trace language “no a in parallel with a b ” is not a locally safety language (but it is Scott-closed).

For first-order languages we have, as usual, also a characterization by temporal logics:

Proposition 2. *The following are equivalent characterizations for $L \subseteq \mathbb{R}$:*

1. L is a locally-safety language definable in first-order logic.
2. L is definable by a globally past formula in LocTL.
3. $K = Lin(L) \subseteq \Sigma^\infty$ is a first-order, prefix-closed language such that $K \cap \Sigma^\omega$ is a safety language, and $K \cap \Sigma^*$ satisfies the forward diamond property.

Proof. The equivalence (1) \Leftrightarrow (2) follows from [4], and the implication (1) \Rightarrow (3) is immediate. For (3) \Rightarrow (1) it suffices to show that $L = \overline{L}$ (since we know by [7] that L must be first-order). So let $t = \sqcup C$, with $C \subseteq \mathbb{P}(L)$ coherent. For every $u \in \mathbb{P}(L)$ and every linearization x of u , we have $x \in K$ since K is prefix-closed. Moreover, if $\{t, t'\}$ is coherent and K contains all linearizations of t and t' , respectively, then by the forward diamond property, K contains some (and thus all) linearization(s) of $t \sqcup t'$. This shows the claim for finite traces t . For infinite traces it follows from $K \cap \Sigma^\omega$ being a safety language.

4 Local Monitoring

Here and in the following we write $s \leq L$ for a (finite) trace $s \in \mathbb{R}$ and a language $L \subseteq \mathbb{R}$ if there exists some $t \in L$ with $s \leq t$.

Definition 2. A set $L \subseteq \mathbb{R}$ is called *locally monitorable* if for all $s \in \mathbb{P}$ there exists some $t \in \mathbb{P}$ with (1) $s \leq t\mathbb{R}$ and (2) either $t\mathbb{R} \subseteq L$ or $t\mathbb{R} \subseteq L^c$.

Notice that in the definition of locally monitorable sets, the first condition says that $\{s, t\}$ is coherent. So a set L is *locally monitorable* if for every *prime* trace s there is another *prime* trace t that is coherent with s and such that after t we know that every extension belongs either to L or to its complement L^c .

The following lemma extends a well-known observation from words to traces:

Lemma 3. Every prime-closed trace language is locally monitorable. In particular, every locally-safety (or locally-co-safety) language is locally monitorable.

Proof. Let $L = \overline{L}$ and $s \in \mathbb{P}$. If $s\mathbb{R}$ is not a subset of L , then there exists some $t = sx \in L^c$. Since L is prime-closed this means that there is some $u \in \mathbb{P} \setminus \mathbb{P}(L)$ with $u \leq t$. But then $\{u, s\}$ is coherent, thus $s \leq u\mathbb{R}$ and $u\mathbb{R} \subseteq L^c$.

The next proposition characterizes locally monitorable sets in terms of the closure operator defined in the previous section:

Proposition 3. $L \subseteq \mathbb{R}$ is locally monitorable if and only if $\overline{L} \cap \overline{L^c}$ does not contain any non-empty prime-open subset.

Proof. First, assume by contradiction that L is locally monitorable, but $s\mathbb{R} \subseteq \overline{L} \cap \overline{L^c}$ for some $s \in \mathbb{P}$. By symmetry in L and L^c we may assume that we find $t \in \mathbb{P}$ and $s \leq t\mathbb{R} \subseteq L$. Hence, $t \notin \mathbb{P}(L^c)$ and thus $t\mathbb{R} \cap \overline{L^c} = \emptyset$. But $s\mathbb{R} \cap t\mathbb{R} \neq \emptyset$. Contradiction.

For the other direction let $s \in \mathbb{P}$. We may assume (again by symmetry in L and L^c) that $s\mathbb{R} \cap \overline{L^c} \neq \emptyset$. Hence, there is $x \notin \overline{L}$ with $s \leq x$. This implies that there is $t \in \mathbb{P} \setminus \mathbb{P}(L)$ with $s \leq t\mathbb{R}$. Thus, $t\mathbb{R} \subseteq L^c$ and L is locally monitorable.

We state now the main result of this section, which shows that whenever a recognizable property over traces is locally monitorable, we can build a monitor that is of the same type as the system on which it runs, i.e., an asynchronous automaton.

Theorem 2. *Let $L \subseteq \mathbb{R}$ be recognizable. Then we can decide whether L is locally monitorable. Moreover, if L is locally monitorable, then we find a deterministic asynchronous finite state monitor for L .*

Proof. By Lemma 2 there exist deterministic asynchronous automata \mathcal{A} , \mathcal{A}' accepting \overline{L} and \overline{L}^c , resp., such that all their reachable states are final.

Let $(\delta_a)_{a \in \Sigma}, (\delta'_a)_{a \in \Sigma}$ be the transition functions of $\mathcal{A}, \mathcal{A}'$, resp. We modify the product automaton $\mathcal{A} \times \mathcal{A}'$ to a (deterministic) asynchronous automaton \mathcal{C} with transition functions $(\Delta_a)_{a \in \Sigma}$: first we add two local states $\perp_\alpha, \top_\alpha$ on each process $\alpha \in Proc$. Consider $a \in \Sigma$ and some trace t on which \mathcal{A} reaches state s and \mathcal{A}' reaches state s' . Note that ta belongs to one of \overline{L} or \overline{L}^c (or both). If \mathcal{A} has no a -transition on $s_{dom(a)}$ then we add $\Delta_a((s_\alpha, s'_\alpha)_{\alpha \in dom(a)}) = (\perp_\alpha)_{\alpha \in dom(a)}$. If \mathcal{A}' has no a -transition on $s'_{dom(a)}$ then we add the transition $\Delta_a((s_\alpha, s'_\alpha)_{\alpha \in dom(a)}) = (\top_\alpha)_{\alpha \in dom(a)}$. The first case corresponds to $ta\mathbb{R} \cap \overline{L} = \emptyset$, the second one to $ta\mathbb{R} \cap \overline{L}^c = \emptyset$. Else, $\Delta_a((s_\alpha, s'_\alpha)_{\alpha \in dom(a)})$ is defined as the componentwise product of $\delta_a(s_{dom(a)})$ and $\delta'_a(s'_{dom(a)})$. Finally, for each $a \in \Sigma$ and each tuple $\hat{s}_{dom(a)}$ of states of $\mathcal{A} \times \mathcal{A}'$: if some component of $\hat{s}_{dom(a)}$ is \perp , then all components of $\Delta_a(\hat{s}_{dom(a)})$ become \perp , and symmetrically for \top . The language L is not locally monitorable if and only if the automaton \mathcal{C} has some infinite run where no process gets into state \top or \perp .

Proposition 4. *The following problem is PSPACE-hard:*

- *Input:* A Büchi automaton $\mathcal{B} = \langle Q, \Sigma, \delta, q_0, F \rangle$.
- *Question:* Is the accepted language $\mathcal{L}(\mathcal{B}) \subseteq \Sigma^\omega$ monitorable?

Proof. The universality problem for non-deterministic finite automata (NFA) is one of the well-known PSPACE complete problems. We reduce this problem to the problem of monitorability.

Start with an NFA $\mathcal{A} = \langle Q', \Gamma, \delta', q_0, F' \rangle$. We will construct a Büchi automaton \mathcal{B} such that we have $\mathcal{L}(\mathcal{A}) = \Gamma^*$ if and only if $\mathcal{L}(\mathcal{B}) \subseteq \Sigma^\omega$ is monitorable.

For this we use a new letter b and we let $\Sigma = \Gamma \cup \{b\}$. We use three new states d, e, f and we let $Q = Q' \cup \{d, e, f\}$. The repeated (or final) states of \mathcal{B} are defined as $F = \{e, f\}$. The initial state is the same as before: q_0 . It remains to define δ . We keep all arcs from δ' and we add the following new arcs.

- $q \xrightarrow{b} d \xrightarrow{a} e \xrightarrow{a} e$ for all $q \in Q' \setminus F'$ and all $a \in \Gamma$.
- $e \xrightarrow{b} d \xrightarrow{b} d$
- $q \xrightarrow{b} f \xrightarrow{c} f$ for all $q \in F'$ and all $c \in \Sigma$.

In order to understand the construction, consider what happens if we reach state d or state f . Starting in f we accept everything, because we loop in a final state of \mathcal{B} . On the other hand starting in d we accept all words except those which end in b^ω . Starting in d we are nowhere monitorable.

Now, let $w \in \Sigma^*$. This can be written as uv where $u \in \Gamma^*$ is the maximal prefix without any occurrence of b .

Assume we have $\mathcal{L}(\mathcal{A}) = \Gamma^*$, then there is path from q_0 to f labelled by wb since reading u leads us to some state in F' . This implies that $wb\Sigma^\omega \subseteq \mathcal{L}(\mathcal{B})$ for all $w \in \Gamma^*$; and $\mathcal{L}(\mathcal{B})$ is monitorable.

On the other hand, if $\mathcal{L}(\mathcal{A}) \neq \Gamma^*$, then there is some word $u \in \Gamma^*$ such that u leads to states in $Q' \setminus F'$, only. Thus, reading ub we are necessarily in state d . The language $\mathcal{L}(\mathcal{B})$ is not monitorable, due to the word $ub \in \Sigma^*$.

We have a matching upper bound for Büchi automata in the theorem below. Note that the input is a Büchi automaton accepting a trace-closed language, therefore we may see the accepted language also as a subset of \mathbb{R} .

Theorem 3. *The following problem is PSPACE-complete:*

- *Input: A Büchi automaton $\mathcal{B} = \langle Q, \Sigma, \delta, q_0, F \rangle$ and (Σ, dom) such that $\mathcal{L}(\mathcal{B})$ is trace-closed.*
- *Question: Is the accepted language $\mathcal{L}(\mathcal{B}) \subseteq \mathbb{R}$ locally monitorable?*

Proof. For a subset $P \subseteq Q$ let us write $\mathcal{L}(\mathcal{B}, P)$ for the accepted language of \mathcal{B} when P is used as a set of initial states. We say that P is *good* if either $\mathcal{L}(\mathcal{B}, P) = \Sigma^\omega$ or $\mathcal{L}(\mathcal{B}, P) = \emptyset$. The predicate whether P is good can be computed in PSPACE. For a letter $a \in \Sigma$ and $P, P' \subseteq Q$ we define another predicate $\text{Reach}(P, P', a)$, which is defined to be *true*, if:

$$P' = \{q \in Q \mid \exists p \in P \exists ta \in \mathbb{P} \text{ and } p \xrightarrow{ta} q\}.$$

Note that $\text{Reach}(P, P', a)$ is computable in PSPACE, too. If there is no $a \in \Sigma$ such that $\text{Reach}(\{q_0\}, P', a)$ becomes true for some good $P' \subseteq Q$, then $L = \mathcal{L}(\mathcal{B})$ is not locally monitorable. Thus, we may assume that such P and a exist. If there are two letters a and b in different connected components of (Σ, dom) with this property, then L is locally monitorable. Hence we assume in the following that there is only one component where such a letter a exist. Indeed, letters occurring in some prime traces belong to a single connected component of (Σ, dom) ; and due to $\text{Reach}(\{q_0\}, P', a)$ it is enough to consider monitorability of prime traces which belong to the same component as the letter a . Since every such prime trace can be made longer such that it ends with this letter a , we fix a in the following.

Now, the language $L \subseteq \mathbb{R}$ is locally monitorable if and only if for all $P \subseteq Q$ such that $\text{Reach}(\{q_0\}, P, a)$ holds, there is some good subset P' such that we have $\text{Reach}(P, P', a)$.

To see this, let $L \subseteq \mathbb{R}$ be locally monitorable. Consider a subset P such that $\text{Reach}(\{q_0\}, P, a)$ holds. This corresponds to some word s such that the corresponding trace $s = s'a$ is a prime. Since L is locally monitorable, there exists some prime t such that $s \leq t\mathbb{R}$ and either $t\mathbb{R} \subseteq L$ or $t\mathbb{R} \subseteq L^c$. However, by the assumption above, we may assume that s and t belong to the same component. We can make t longer and actually assume $s \leq t$ and such that $t = t'a$. Choose some representing word w for t . If P' is the subset of states we can reach after reading w starting in q_0 we have $\text{Reach}(P, P', a)$. The set

P' is good, because L is trace-closed. Indeed, if $t\mathbb{R} \subseteq L$, then $w\Sigma^\omega \subseteq L$, hence $\mathcal{L}(\mathcal{B}, P') = \Sigma^\omega$. If $t\mathbb{R} \subseteq L^c$, then $\mathcal{L}(\mathcal{B}, P') = \emptyset$.

For the converse it is clear that the condition is strong enough to ensure local monitorability of L .

The condition to monitor a single language might be an unnecessary restriction. We can imagine a certain family of properties or languages L_1, \dots, L_n and we content ourselves with a monitor which selects one of these possibilities, even if certain L_i and L_j do intersect non-trivially for $i \neq j$. This leads to the following definition.

Definition 3. Let $n \in \mathbb{N}$ and L_1, \dots, L_n be subsets of \mathbb{R} . We say that the family $\{L_1, \dots, L_n\}$ is locally monitorable, if

$$\forall s \in \mathbb{P} \exists t \in \mathbb{P} \exists 1 \leq i \leq n : s \leq t\mathbb{R} \subseteq L_i.$$

Remark 2. A language L is locally monitorable if and only if the family $\{L, L^c\}$ is locally monitorable.

A distributed alphabet (Σ, dom) can be split into several *connected components*. This is a partition $\Sigma = \Sigma_1 \cup \dots \cup \Sigma_k$ such that all Σ_i are non-empty and $\Sigma_i \times \Sigma_j \subseteq I$ for all $1 \leq i < j \leq k$. We say that (Σ, dom) is *connected*, if $k = 1$ and *disconnected* otherwise. For $k \geq 2$ we can write $\mathbb{R} = \mathbb{R}' \times \mathbb{R}''$ such that \mathbb{R}' and \mathbb{R}'' are both infinite.

4.1 Disconnected Case

We assume in this section that (Σ, dom) is disconnected and we write $\mathbb{R} = \mathbb{R}' \times \mathbb{R}''$. Let $L \subseteq \mathbb{R}$. If L is locally monitorable then, necessarily $s\mathbb{R} \subseteq L$ or $s\mathbb{R} \subseteq L^c$ for some prime $s \in \mathbb{P} = \mathbb{P}(\mathbb{R}') \cup \mathbb{P}(\mathbb{R}'')$. By symmetry we may assume $s \in \mathbb{P}(\mathbb{R}')$ and $s\mathbb{R} \subseteq L$. As a consequence, there is no $t \in \mathbb{P}(\mathbb{R}'')$ such that $t\mathbb{R} \subseteq L^c$. On the other hand, if there is some prime $t \in \mathbb{P}(\mathbb{R}'')$ such that $t\mathbb{R} \subseteq L$, then L is locally monitorable for a trivial reason: For every prime trace $u \in \mathbb{P}$ we either have $u \in \mathbb{R}'$ or $u \in \mathbb{R}''$; and by choosing either the prime s or t in the other component as u we satisfy the required condition for L to be locally monitorable.

Hence we are only interested in the case that there is no prime $t \in \mathbb{R}''$ such that $t\mathbb{R} \subseteq L$. In this case we can reduce the problem whether L is locally monitorable to the component of \mathbb{R}' as follows: First, let us define languages of prime traces $L_1 = \{u \in \mathbb{P}(\mathbb{R}') \mid u\mathbb{R} \subseteq L\}$ and $L_2 = \{u \in \mathbb{P}(\mathbb{R}') \mid u\mathbb{R} \subseteq L^c\}$. Note that if L is recognizable, then L_1, L_2 , as well as $L_1\mathbb{R}', L_2\mathbb{R}'$, are recognizable too. Moreover, we can construct the corresponding automata.

Theorem 4. Let $L \subseteq \mathbb{R} = \mathbb{R}' \times \mathbb{R}''$ and assume that there is some $s \in \mathbb{P}(\mathbb{R}')$ such that $s\mathbb{R} \subseteq L$ but there is no $t \in \mathbb{P}(\mathbb{R}'')$ with $t\mathbb{R} \subseteq L$. Then L is locally monitorable if and only if the family $\{L_1\mathbb{R}', L_2\mathbb{R}'\}$ is locally monitorable w.r.t. \mathbb{R}' .

Proof. First, let L be locally monitorable and $s \in \mathbb{P}$ be a prime. Choose some prime $t \in \mathbb{P}$ with $s \leq t\mathbb{R}$ such that either $t\mathbb{R} \subseteq L$ or $t\mathbb{R} \subseteq L^c$. We cannot have

$t \in R''$, hence $t \in \mathbb{P}(R')$. Thus, either $t \in L_1$ or $t \in L_2$. It follows that $tR' \subseteq L_1R'$ or $tR' \subseteq L_2R'$, and hence $\{L_1R', L_2R'\}$ is locally monitorable w.r.t. R' .

For the other direction let $\{L_1R', L_2R'\}$ be locally monitorable w.r.t. R' . Then for every prime $u \in \mathbb{P}(R')$ there is some $v \in \mathbb{P}(R')$ such that $u \leq vR'$ such that either $vR' \subseteq L_1R'$ or $vR' \subseteq L_2R'$. In particular, either $v \in L_1$ or $v \in L_2$, since $r \leq v$ with $r \in L_i$ implies $v \in L_i$. By definition, either $vR \subseteq L$ or $vR \subseteq L^c$. Thus, L is locally monitorable on all primes of R' . Now, let $u \in \mathbb{P}(R'')$. By assumption there is some $s \in \mathbb{P}(R')$ such that $sR \subseteq L$. Since $R = R' \times R''$ we have $u \leq sR$. Thus, L is locally monitorable.

4.2 Connected Case

Recall that a distributed alphabet (Σ, dom) is *connected* if it cannot be partitioned as $\Sigma = \Sigma_1 \cup \Sigma_2$ such that $\Sigma_1 \times \Sigma_2 \subseteq I$ with $\Sigma_1 \neq \emptyset \neq \Sigma_2$. For connected (Σ, dom) we obtain a nicer characterization of locally monitorable sets:

Lemma 4. *Let (Σ, dom) be connected. Then L is locally monitorable if and only if*

$$\forall s \in \mathbb{P} \exists s \leq t \in \mathbb{P} : tR \subseteq L \vee tR \subseteq L^c.$$

Proof. Let L be such that $\forall s \in \mathbb{P} \exists t \in \mathbb{P} : s \leq tR \subseteq L \vee s \leq tR \subseteq L^c$. We have to show that we can choose s to be a prefix of t . But this is clear: if $s \leq tR$, then there is a prime p with $s \leq p$ and $t \leq p$. The result follows because $pR \subseteq tR$ in this case.

Proposition 5. *The following assertions are equivalent.*

1. (Σ, dom) is connected.
2. The family of locally monitorable sets is closed under finite union.
3. The family of locally monitorable sets is a Boolean algebra.

Proof. Since the locally monitorable property is symmetric for L, L^c , the last two items of the proposition are equivalent. Let (Σ, dom) be connected, we show that locally monitorable is preserved by taking finite unions. Let L and K be locally monitorable and consider $s \in \mathbb{P}$. If we find $s \leq t \in \mathbb{P}$ and either $tR \subseteq L$ or $tR \subseteq K$, we are done. Hence there is $s \leq t \in \mathbb{P}$ and $tR \subseteq L^c$. Now, we may assume that there is $t \leq u \in \mathbb{P}$ and $uR \subseteq K^c$. But then $s \leq u$ and $uR \subseteq (L \cup K)^c$.

Conversely, let $a, b \in \Sigma$ be in different connected components of (Σ, dom) and let $L =$ “no occurrence of a ” and $K =$ “no occurrence of b ”. Both sets are locally monitorable, since they are prime-closed. However, for every prime s we have $s \in L \cup K$ and $sR \cap (L \cup K)^c \neq \emptyset$. This shows that $L \cup K$ is not locally monitorable.

Again, for connected alphabets and a family of languages, we can make the condition to be locally monitorable more precise by using Lem. 4. Indeed, if (Σ, dom) is connected, then a family $\{L_1, \dots, L_n\}$ is locally monitorable if and only if

$$\forall s \in \mathbb{P} \exists s \leq t \in \mathbb{P} \exists 1 \leq i \leq n : tR \subseteq L_i.$$

Theorem 5. *Let (Σ, dom) be connected, and L_1, \dots, L_n be subsets of \mathbb{R} such that*

1. $\mathbb{R} = L_1 \cup \dots \cup L_n$.
2. Each L_k is a countable union of prime-closed sets.

Then the family $\{L_1, \dots, L_n\}$ is locally monitorable.

Proof. We give the proof for $n = 2$, the one for $n > 2$ is similar. Let $L = L_1^c$ and $K = L_2^c$. Write $L = \bigcap_{i \geq 0} U_i \mathbb{R}$ and $K = \bigcap_{i \geq 0} V_i \mathbb{R}$ where all $U_i, V_i \subseteq \mathbb{P}$. Without restriction we have $U_0 \mathbb{R} = V_0 \mathbb{R} = \mathbb{R}$.

By contradiction, assume that $\{L_1, L_2\}$ is not locally monitorable. This means that we can find some $s \in \mathbb{P}$ such that for all $t \in \mathbb{P}$ with $\{s, t\}$ coherent it holds that $t\mathbb{R} \cap L \neq \emptyset \neq t\mathbb{R} \cap K$. Let $p_0 = x_0 = q_0 = y_0 = s$.

By induction let for some $k \geq 1$ prime traces p_i, x_i, q_i , and y_i for all $0 \leq i < k$ be defined such that $U_i \ni p_i \leq x_i \leq y_i$, $V_i \ni q_i \leq y_i$, and $y_{i-1} \leq x_i$.

We define x_k, p_k as follows. Since $s \leq y_{k-1} \in \mathbb{P}$ we have by assumption $y_{k-1} \mathbb{R} \cap L \neq \emptyset$, and thus we find $y_{k-1} \leq x \in L$. Thus, there is $p_k \in U_k$ with $p_k \leq x$. The set $\{y_{k-1}, p_k\}$ is coherent, hence there is common finite trace w with $y_{k-1} \leq w$ and $p_k \leq w$. Since (Σ, dom) is connected, we find some prime $x_k \in \mathbb{P}$ with $w \leq x_k$. The definition of y_k follows the same pattern. We have $s \leq x_1 \leq y_1 \leq x_2 \dots$ and $x = \sqcup \{x_i \mid i \in \mathbb{N}\}$ exists. However, $x \in \bigcap_{i \geq 0} U_i \mathbb{R} \cap \bigcap_{i \geq 0} V_i \mathbb{R}$. Contradiction, because $L \cap K = \emptyset$.

Remark 3. Notice that the above proof still works if (Σ, dom) has only two connected components. In the general case it is open whether the statement of Thm. 5 still holds.

5 Infinite Traces

Prime-closed languages are prefix closed, so they always intersect. In particular, for any language $L \subseteq \mathbb{R}$, it can never happen that both L and L^c are countable unions of prime-closed sets (or equivalently, countable intersections of prime-open sets), as required by Thm. 5.

Thus, in order to define a trace analogue of $G_\delta \cap F_\sigma$ we will restrict our attention to infinite traces where a (given) subset Γ of processes is active infinitely often and “sees” all other processes. In this way monitoring can be performed by processes in Γ . Another motivation for the new notion is due to the fact that in order to monitor a language we should be able to gather information into longer and longer prime prefixes.

For a finite trace t we write $\max(t) \subseteq \Gamma$ if $\text{dom}(a) \cap \Gamma \neq \emptyset$ for each $a \in \max(t)$.

Definition 4. *Let Γ be a (non-empty) subset of Proc. A trace x is called Γ -infinite if*

- Every process from Γ has infinitely many actions in x .
- x can be written as $x = x_0 x_1 \dots$ such that $\max(x_n) \subseteq \Gamma$ for each $n \geq 0$.

– $\text{alph}(x)$ is connected.

The set of Γ -infinite traces is written as \mathbb{R}_Γ .

Remark 4. If Γ is a singleton, then for every trace $x \in \mathbb{R}_\Gamma$, both $\text{alph}(x)$ and $\text{alphinf}(x)$ are connected (and non-empty).

In the following everything is within Γ -infinite traces, for a fixed set $\Gamma \subseteq \text{Proc}$. In particular, the notion of closed and open are meant to be induced. The notion of locally monitorable is also relative to \mathbb{R}_Γ : a set $L \subseteq \mathbb{R}_\Gamma$ is locally monitorable if $\forall s \in \mathbb{P}(\mathbb{R}_\Gamma) \exists s \leq t \in \mathbb{P}(\mathbb{R}_\Gamma) : t\mathbb{R} \cap \mathbb{R}_\Gamma \subseteq L \vee t\mathbb{R} \cap \mathbb{R}_\Gamma \subseteq L^c$ (where $L^c = \mathbb{R}_\Gamma \setminus L$).

Definition 5. Let $\Gamma \subseteq \text{Proc}$ be a non-empty set of processes.

1. A set $X \subseteq \mathbb{R}_\Gamma$ is prime- G_δ if it has the form $X = \bigcap_{i \geq 0} U_i$ where all U_i are prime-open in \mathbb{R}_Γ . The family of prime- G_δ -sets is denoted PG_δ .
2. A set $X \subseteq \mathbb{R}_\Gamma$ is prime- F_σ if its complement is prime- G_δ . The family of prime- G_δ -sets is denoted PF_σ .

Example 3. Let $\Gamma = \text{Proc} = \{\alpha, \beta\}$ and $\Sigma = \{a, b, d\}$ with $\text{dom}(a) = \{\alpha\}$, $\text{dom}(b) = \{\beta\}$ and $\text{dom}(d) = \{\alpha, \beta\}$. Let $L \subseteq \mathbb{R}_\Gamma$ contain all traces without the (trace) factor abd . Such traces are formed either by a trace from $((a^* + b^*)d^+)^*(a^* + b^*)d^+$ followed by $a^\omega b^\omega$, or they belong to $((a^* + b^*)d^+)^\omega$. Clearly, L is prime-closed. The complement of L is in PF_σ , since $L^c = \bigcup_{w \in \Sigma^*, i, j > 0} X_{w, i, j}$ where $X_{w, i, j}$ contains all traces from \mathbb{R}_Γ with prefix $wa^i b^j d$. Each $X_{w, i, j}$ is prime-closed.

The next lemma generalizes the case of ω -words. Note that we need the restriction to \mathbb{R}_Γ (or some similar restriction). As an example, consider $\Sigma = \{a, b\}$ with $(a, b) \in I$. The language $L = a\mathbb{R}$ is prime-open. But its complement $L^c = b^\infty$ cannot be written as countable intersection of prime-open sets in \mathbb{R} , since we cannot avoid occurrences of a in such sets.

Lemma 5. Prime-closed sets of \mathbb{R}_Γ are in PG_δ .

Proof. Let $L \subseteq \mathbb{R}_\Gamma$ be prime-closed. By definition, every $\sqcup C \in \mathbb{R}_\Gamma$ where C is coherent and $C \subseteq \mathbb{P}(L)$, belongs to L . For $K \subseteq \mathbb{P}$, $\alpha \in \Gamma$ and $k \in \mathbb{N}$ let

$$K_{\alpha, k} = \{p \in K \mid |p| \geq k, \alpha \in \text{dom}(\max(p))\}.$$

We claim that

$$L = \bigcap_{k \in \mathbb{N}, \alpha \in \Gamma} \mathbb{P}(L)_{\alpha, k} \mathbb{R}_\Gamma.$$

The inclusion from left to right follows from $L \subseteq \mathbb{R}_\Gamma$ and the definition of \mathbb{R}_Γ . Let $x \in \mathbb{R}_\Gamma$ be such that for every $k \in \mathbb{N}$ and $\alpha \in \Gamma$, there is some $p_{\alpha, k} \leq x$ with $p_{\alpha, k} \in \mathbb{P}(L)_{\alpha, k}$. By definition of \mathbb{R}_Γ and of $\mathbb{P}(L)_{\alpha, k}$, we have that $x = \sqcup \{p_{\alpha, k} \mid k \in \mathbb{N}, \alpha \in \Gamma\}$. Hence, x is of the form $\sqcup C$ for $C \subseteq \mathbb{P}(L)$ coherent, and thus in L .

- Theorem 6.** 1. $\text{PG}_\delta \cap \text{PF}_\sigma$ is a Boolean algebra containing all prime-open and all prime-closed subsets of \mathbb{R}_Γ .
 2. All $\text{PG}_\delta \cap \text{PF}_\sigma$ subsets of \mathbb{R}_Γ are locally monitorable.

Proof. PG_δ is closed under union. Hence, $\text{PG}_\delta \cap \text{PF}_\sigma$ is a Boolean algebra. It contains all prime-open and all prime-closed subsets of \mathbb{R}_Γ by Lem. 5.

The proof of the second claim follows along the same lines as the one of Thm. 5. Assume that $\mathbb{R}_\Gamma \neq \emptyset$ and choose some connected subalphabet Σ' of Σ that contains for each $\alpha \in \Gamma$ some letter a with $\alpha \in \text{dom}(a)$. The prime traces x_k, y_k can be chosen such that $\max(x_k) \subseteq \Gamma$, $\max(y_k) \subseteq \Gamma$, and $\text{alph}(x_k^{-1}y_k) = \text{alph}(y_{k-1}^{-1}x_k) = \Sigma'$. Thus, $x = \sqcup_i x_i \in \mathbb{R}_\Gamma$.

Asynchronous Büchi and Muller automata have been studied in [8,5]. McNaughton's theorem [13] stating the equivalence of non-deterministic Büchi and deterministic Muller automata over omega-word languages, extends to recognizable languages of infinite traces and asynchronous automata [5]. If we restrict to traces from \mathbb{R}_Γ , then the Büchi and Muller acceptance conditions are simpler:

Definition 6. Let $\Gamma \subseteq \text{Proc}$ be a non-empty set of processes, and let $\mathcal{A} = \langle (S_\alpha)_{\alpha \in \text{Proc}}, (\delta_a)_{a \in \Sigma}, s^0 \rangle$ be an asynchronous automaton.

1. A Büchi acceptance condition is a set $F \subseteq S_\Gamma$.

An infinite run $s^0 = s_0, a_0, s_1, a_1, \dots$ of \mathcal{A} is accepting if for some $f_\Gamma \in F$ and for every $\alpha \in \Gamma$, there are infinitely many $n \geq 0$ with $(s_n)_\alpha = f_\alpha$.

2. A Muller acceptance condition is a set $\mathcal{F} \subseteq \prod_{\alpha \in \Gamma} 2^{S_\alpha}$.

An infinite run $s^0 = s_0, a_0, s_1, a_1, \dots$ of \mathcal{A} is accepting if for some $T_\Gamma \in \mathcal{F}$ and for every $\alpha \in \Gamma$, the set of states from S_α such that $(s_n)_\alpha = f_\alpha$ for infinitely many n , is precisely T_α .

The language $\mathcal{L}(\mathcal{A})$ is the set of all traces from \mathbb{R}_Γ that have an accepting run. The next result is a generalization from ω -word languages to \mathbb{R}_Γ trace languages:

Theorem 7. Let $L \subseteq \mathbb{R}_\Gamma$ be recognizable. Then L is in PG_δ if and only if L is accepted by a deterministic Büchi asynchronous automaton.

Proof. Assume first that $L = \mathcal{L}(\mathcal{A})$, where \mathcal{A} is a deterministic asynchronous Büchi automaton, and fix a final state $f \in F$. For $n > 0, \alpha \in \Gamma$ we define $K_{n,\alpha}^f$ as the set of all traces $t \in \mathbb{P}$ with $\alpha \in \text{dom}(\max(t))$ and such that in the run of \mathcal{A} on t , at least n letters on process α are in state f_α . It is easy to see that the set $\bigcup_{f \in F} \bigcap_{\alpha \in \Gamma, n > 0} K_{n,\alpha}^f \mathbb{R}_\Gamma$ is precisely $\mathcal{L}(\mathcal{A})$. The remaining of the proof will show that PG_δ is closed under finite union, thus $\mathcal{L}(\mathcal{A}) \in \text{PG}_\delta$.

For the converse let $L = \bigcap_{n > 0} U_n \subseteq \mathbb{R}_\Gamma$ be recognizable, with U_n prime-open in \mathbb{R}_Γ . We first define $V_n = \bigcap_{k \leq n} U_k$. It is not difficult to see that each V_n can be assumed to be of the form $K_n \mathbb{R}_\Gamma$ with $\max(t) \subseteq \Gamma$ for each $t \in K_n$. Let now $K'_n \subseteq K_n$ consist of all elements of K_n that have no proper prefix in K_n . Let $K = \bigcup_{n > 0} K'_n X_n$, where X_n is the set of traces t such that (1) $\max(t) \subseteq \Gamma$, (2) $|t|_\alpha \geq n$ for each $\alpha \in \Gamma$, and (3) no proper prefix of t satisfies (1) and (2).

Let us first show that $L = \{\sqcup C \mid C \subseteq K, C \text{ coherent}\}$. The inclusion from left to right follows from $L = \bigcap_{n>0} U_n = \bigcap_{n>0} K_n \mathbb{R}_\Gamma = \bigcap_{n>0} K'_n \mathbb{R}_\Gamma = \bigcap_{n>0} K'_n X_n \mathbb{R}_\Gamma$. Conversely, let $t = x_0 x_1 \dots$ with $x_0 \dots x_n \in K$ for all n . Observe that we must have infinitely many n such that $x_0 \dots x_m \in K_n$ for some m , since K'_n is prefix-free. Thus, $t \in V_n$ for infinitely many n and $t \in U_n$ for all n .

To conclude, we show that if $L = \{\sqcup C \mid C \subseteq K, C \text{ coherent}\}$ for some K , and $L \subseteq \mathbb{R}_\Gamma$ is recognizable, then L is the language of a deterministic asynchronous Büchi automaton. We assume as above that $\max(t) \subseteq \Gamma$ for all $t \in K$. Since L is recognizable, there is some deterministic Muller automaton \mathcal{A} with acceptance condition \mathcal{F} and $\mathcal{L}(\mathcal{A}) = L$. We may also assume that on every finite trace t the states of processes from $\text{dom}(\max(t))$ reached on t determine the states of all other processes. First we test for every $T \in \mathcal{F}$ if there is some trace from \mathbb{R}_Γ accepted with T . Without restriction this is the case for all $T \in \mathcal{F}$. For each T we can determine a reachable state $s(T) \in \prod_{\alpha \in \Gamma} T_\alpha$ and finite traces $t_0(T), t(T)$ with $\max(t_0(T), \max(t(T))) \subseteq \Gamma$ such that (1) $t_0(T)$ leads from the initial state to $s(T)$, (2) $t(T)$ is a loop on state $s(T)$ and (3) the set of α -states in the loop $t(T)$ is precisely T_α . In addition, $t_0(T)$ is connected.

We claim that \mathcal{A} accepts L with the following (Büchi) condition: a trace is accepted if for some $T \in \mathcal{F}$, every state from T_α occurs infinitely often, for every $\alpha \in \Gamma$. It is clear that all of L is accepted in this way by \mathcal{A} . Conversely, let x be an arbitrary trace with $\max(x) \subseteq \Gamma$ and looping on state $s(T)$. We have $t_0 t(T)^\omega \in L$, so there is some n_0 and u_0 in K such that $u_0 \leq t_0 t(T)^{n_0}$. Since $t_0 t(T)^{n_0} x t(T)^\omega \in L$ we find some n_1 such that $u_1 \leq t_0 t(T)^{n_0} x t(T)^{n_1}$ for some $u_1 \in K$ with $u_0 < u_1$. In this way we can build a trace t from \mathbb{R}_Γ , $t = t_0 t(T)^{n_0} x t(T)^{n_1} x \dots$, with $t = \sqcup_{n \geq 0} u_n \in \{\sqcup C \mid C \subseteq K, C \text{ coherent}\}$ and such that for each $\alpha \in \Gamma$, the set of states from S_α repeated infinitely often is a superset of T_α . The claim follows since $L = \{\sqcup C \mid C \subseteq K, C \text{ coherent}\}$.

Remark 5. For the previous proof we do not need the connectedness assumption in the definition of \mathbb{R}_Γ . On the other hand, it is open whether without this assumption all $\text{PG}_\delta \cap \text{PF}_\sigma$ sets are still locally monitorable.

6 Conclusion

Our aim in this paper was to propose a reasonable notion of distributed monitoring for asynchronous systems. We argued that distributed monitors should have the same structure as the system that is monitored. We showed that properties over Γ -infinite traces that are deterministic and co-deterministic, are locally monitorable. It would be interesting to consider alternative restrictions to Γ -infinite traces, that capture some reasonable (partial) knowledge about the asynchronous system and for which $\text{PG}_\delta \cap \text{PF}_\sigma$ sets are locally monitorable.

References

1. Bauer, A., Leucker, M., Schallhart, C.: Monitoring of Real-Time Properties. In: Arun-Kumar, S., Garg, N. (eds.) FSTTCS 2006. LNCS, vol. 4337, pp. 260–272. Springer, Heidelberg (2006)

2. Bauer, A., Leucker, M., Schallhart, C.: Runtime verification for LTL and TLTL. *ACM Trans. Softw. Eng. Methodol.* 20(4) (2011)
3. Chadha, R., Sistla, A.P., Viswanathan, M.: On the expressiveness and complexity of randomization in finite state monitors. *J. ACM* 56(5) (2009)
4. Diekert, V., Gastin, P.: Local safety and local liveness for distributed systems. In: *Perspectives in Concurrency Theory*, pp. 86–106. IARCS-Universities (2009)
5. Diekert, V., Muscholl, A.: Deterministic asynchronous automata for infinite traces. *Acta Informatica* 31, 379–397 (1994)
6. Diekert, V., Rozenberg, G. (eds.): *The Book of Traces*. World Scientific (1995)
7. Ebinger, W., Muscholl, A.: Logical definability on infinite traces. *Theoretical Computer Science* 154(3), 67–84 (1996)
8. Gastin, P., Petit, A.: Infinite traces. In: Diekert, V., Rozenberg, G. (eds.) *The Book of Traces*. World Scientific (1995)
9. Genest, B., Gimbert, H., Muscholl, A., Walukiewicz, I.: Optimal Zielonka-Type Construction of Deterministic Asynchronous Automata. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) *ICALP 2010, Part II*. LNCS, vol. 6199, pp. 52–63. Springer, Heidelberg (2010)
10. Gondi, K., Patel, Y., Prasad Sistla, A.: Monitoring the Full Range of ω -Regular Properties of Stochastic Systems. In: Jones, N.D., Müller-Olm, M. (eds.) *VMCAI 2009*. LNCS, vol. 5403, pp. 105–119. Springer, Heidelberg (2009)
11. Keller, R.M.: Parallel program schemata and maximal parallelism I. Fundamental results. *Journal of the Association of Computing Machinery* 20(3), 514–537 (1973)
12. Mazurkiewicz, A.: Concurrent program schemes and their interpretations. *DAIMI Rep. PB 78*. Aarhus University, Aarhus (1977)
13. McNaughton, R.: Testing and generating infinite sequences by a finite automaton. *Information & Control* 9, 521–530 (1966)
14. Pnueli, A.: The temporal logic of programs. In: *18th Symposium on Foundations of Computer Science*, pp. 46–57 (1977)
15. Pnueli, A., Zaks, A.: PSL Model Checking and Run-Time Verification Via Testers. In: Misra, J., Nipkow, T., Karakostas, G. (eds.) *FM 2006*. LNCS, vol. 4085, pp. 573–586. Springer, Heidelberg (2006)
16. Ștefănescu, A., Esparza, J., Muscholl, A.: Synthesis of Distributed Algorithms Using Asynchronous Automata. In: Amadio, R.M., Lugiez, D. (eds.) *CONCUR 2003*. LNCS, vol. 2761, pp. 27–41. Springer, Heidelberg (2003)
17. Zielonka, W.: Notes on finite asynchronous automata. *R.A.I.R.O. — Informatique Théorique et Applications* 21, 99–135 (1987)

On the Expressive Power of Logics with Invariant Uses of Arithmetic Predicates

Nicole Schweikardt

Goethe-University Frankfurt, Frankfurt am Main, Germany

schweika@cs.uni-frankfurt.de

<http://www.tks.cs.uni-frankfurt.de/schweika>

Abstract. In this talk I consider first-order formulas (FO, for short) where, apart from the symbols in the given vocabulary, also predicates for linear order and arithmetic may be used. For example, order-invariant formulas are formulas for which the following is true: If a structure satisfies the formula with one particular linear order of the structure's universe, then it satisfies the formula with any linear order of the structure's universe. Arithmetic-invariant formulas are defined analogously, where apart from the linear order other arithmetic predicates may be used in an invariant way. The aim of this talk is to give an overview of the state-of-the-art concerning the expressive power of order-invariant and arithmetic-invariant logics.

One way of enhancing the expressive power of a logic is to add predicates for linear order and arithmetic on the elements of a structure, and to allow formulas to use these predicates. To achieve closure under isomorphisms, one wants the formulas to be *invariant* under the particular interpretation of these predicates. For example, *order-invariant* formulas are formulas for which the following is true: If a structure satisfies the formula with one particular linear order, then it satisfies the formula with *any* linear order. From the Immerman-Vardi Theorem it follows that the polynomial-time computable graph properties are precisely captured by *order-invariant* IFP (cf., e.g., [9]). Similarly, *arithmetic-invariant* FO captures the graph properties that belong to the circuit complexity class AC^0 (and thus are highly parallelisable, as they can be checked in constant time using a polynomial number of processors), and arithmetic-invariant IFP captures the graph properties that belong to the class P/Poly [10,11].

In fact, restricting attention to logical formulas that use linear order and arithmetic in an invariant way closely corresponds to restricting attention to computations whose output is independent of the particular encoding of an input graph. This way, invariant logics serve as natural candidates for providing logical characterisations of complexity classes. However, Trakhtenbrot's Theorem implies that it is impossible to automatically check if a given formula is order- or arithmetic-invariant. Thus, invariant logics do not have a decidable syntax. When speaking of "invariant logics", it therefore should be kept in mind that these are "logical systems", but not "logics" in the strict formal sense. The well-known "quest for a logic capturing polynomial time" can thus be re-formulated

as the quest for a logical system that has a decidable syntax and that has the same expressive power as order-invariant IFP.

An easy application of Craig’s Interpolation Theorem shows that first-order sentences that are order-invariant on the class of all structures (finite and infinite), are no more expressive than plain first-order logic. When restricting attention to finite structures, however, the situation is different: A famous example by Gurevich (cf., Theorem 5.3 in [9]) shows that order-invariant FO is strictly more expressive than FO on a class of finite structures suitable for encoding Boolean algebras. This was strengthened by Otto [13] and Rossman [14], showing that already epsilon-invariant FO and successor-invariant FO are more expressive than FO on the class of all finite graphs (epsilon-invariant FO is an extension of FO by invariant uses of an operator that allows to choose an arbitrary element in a given set of elements). Up-to-date, no decidable characterisations of these logical systems on the class of all finite structures are known.

In [15], Rossman showed that the bounded variable hierarchy of order-invariant FO, as well as arithmetic-invariant FO, is strict.

Grohe and Schwentick [6] showed that order-invariant FO queries are Gaifman local with respect to a constant locality radius. In [112] it was shown that arithmetic-invariant FO queries are Gaifman local with respect to a locality radius that is polylogarithmic in the size of the underlying structure. For the particular case of addition-invariant FO it remains open if the result can be strengthened to a constant locality radius.

In [17], Schweikardt and Segoufin obtained a decidable characterisation of addition-invariant FO on the class of all finite coloured sets: On these structures, addition-invariant FO is precisely as expressive as FO_{card} , the extension of FO with predicates for testing the cardinality of a structure’s universe modulo some fixed number. In fact, as shown in [177], FO_{card} precisely characterises the regular word languages and the regular tree languages definable in addition-invariant FO. It remains open, however, whether all languages definable in addition-invariant FO are regular.

In [3], Benedikt and Segoufin showed that on trees and words, order-invariant FO is no more expressive than plain FO, and that on finite graphs of bounded valence or bounded tree-width, order-invariant FO is no more expressive than monadic second-order logic MSO.

By results of Courcelle and Lapoire [48] it is known that order-invariant MSO on finite graphs of bounded tree width has exactly the same expressive power as the extension of MSO with counting quantifiers (CMSO, for short). Ganzow and Rubin [5] proved, however, that on the class of all finite structures, order-invariant MSO is strictly more expressive than CMSO.

Allowing addition-invariance rather than order-invariance with respect to monadic logics drastically increases the expressive power: Already on the class of finite word structures, addition-invariant monadic least fixed-point logic MLFP can define all properties in Grandjean’s linear-time complexity class DLIN, and addition-invariant MSO precisely corresponds to the linear-time hierarchy LinH [1612].

References

1. Anderson, M., van Melkebeek, D., Schweikardt, N., Segoufin, L.: Locality of Queries Definable in Invariant First-Order Logic with Arbitrary Built-in Predicates. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part II. LNCS, vol. 6756, pp. 368–379. Springer, Heidelberg (2011)
2. Anderson, M., van Melkebeek, D., Schweikardt, N., Segoufin, L.: Locality from circuit lower bounds. To appear in SIAM Journal on Computing (2012)
3. Benedikt, M., Segoufin, L.: Towards a characterization of order-invariant queries over tame structures. *Journal on Symbolic Logic* 74(1), 168–186 (2009)
4. Courcelle, B.: The Monadic Second-Order Logic of Graphs X: Linear Orderings. *Theoretical Computer Science* 160(1&2), 87–143
5. Ganzow, T., Rubin, S.: Order-invariant MSO is stronger than Counting MSO in the finite. In: Proc. 25th Symposium on Theoretical Aspects of Computer Science (STACS 2008), pp. 313–324 (2008)
6. Grohe, M., Schwentick, T.: Locality of order-invariant first-order formulas. *ACM Transactions on Computational Logic* 1(1), 112–130 (2000)
7. Harwath, F., Schweikardt, N.: Regular tree languages, cardinality predicates, and addition-invariant FO. In: Proc. 29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012). LIPIcs, vol. 14, pp. 489–500 (2012)
8. Lapoire, D.: Recognizability Equals Monadic Second-Order Definability for Sets of Graphs of Bounded Tree-Width. In: Meinel, C., Morvan, M. (eds.) STACS 1998. LNCS, vol. 1373, pp. 618–628. Springer, Heidelberg (1998)
9. Libkin, L.: *Elements of Finite Model Theory*. Springer (2004)
10. Makowsky, J.A.: Invariant Definability (Extended Abstract). In: Gottlob, G., Leitsch, A., Mundici, D. (eds.) KGC 1997. LNCS, vol. 1289, pp. 186–202. Springer, Heidelberg (1997)
11. Makowsky, J.A.: Invariant Definability and $\mathbf{P}/poly$. In: Gottlob, G., Grandjean, E., Seyr, K. (eds.) CSL 1998. LNCS, vol. 1584, pp. 142–158. Springer, Heidelberg (1999)
12. More, M., Olive, F.: Rudimentary languages and second order logic. *Mathematical Logic Quarterly* 43, 419–426 (1997)
13. Otto, M.: Epsilon-logic is more expressive than first-order logic over finite structures. *Journal of Symbolic Logic* 65(4), 1749–1757 (2000)
14. Rossman, B.: Successor-invariant first-order logic on finite structures. *Journal of Symbolic Logic* 72(2), 601–618 (2007)
15. Rossman, B.: On the constant-depth complexity of k -clique. In: Proc. 40th Annual ACM Symposium on the Theory of Computing (STOC 2008), pp. 721–730. ACM (2008)
16. Schweikardt, N.: On the expressive power of monadic least fixed point logic. *Theoretical Computer Science* 350(2-3), 325–344 (2006)
17. Schweikardt, N., Segoufin, L.: Addition-invariant FO and regularity. In: Proc. 25th IEEE Symposium on Logic in Computer Science (LICS), pp. 273–282 (2010)

Logical Methods in Quantum Information Theory

Peter Selinger

Department of Mathematics and Statistics, Dalhousie University, Halifax, Nova
Scotia, Canada

`selinger@mathstat.dal.ca`

<http://www.mathstat.dal.ca/~selinger/>

Abstract. I will talk about some recent applications of logical methods to quantum information theory. In computing, a higher-order function is a function for which the input or output is another function. I will argue that many of the interesting phenomena of quantum information theory involve higher-order functions, although that is often not how they are presented. I'll talk about the quantum lambda calculus as a possible framework to describe such phenomena.

Quantifying Notes

Hans van Ditmarsch^{1,2}

¹ University of Seville, Spain
hvd@us.es

² IMSc, Chennai, India

Abstract. We review several logics with propositional quantification.

1 Introduction

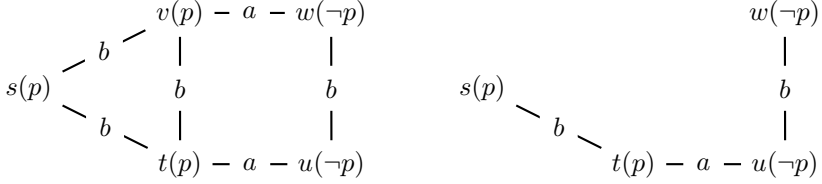
You are uncertain whether Cordoba is in Spain. Carlos is in a position to inform you. Let us assume for a moment that Cordoba is in fact in Spain (it's that town 250 kilometers east of Sevilla). This may of course be contested in Argentina, but we won't. Carlos has two possible answers, he can say: "Yes, Cordoba is in Spain," or he can say, politely: "I am also uncertain whether Cordoba is in Spain." In other words, there is an announcement after which you know that Cordoba is in Spain and there also is an announcement after which you do not know that Cordoba is in Spain. What is the logic for 'there is an announcement after which'? In this paper we present our various recent proposals for propositional quantification in modal logics. We focus on open problems. Different ways to quantify over information change include (where φ is a formula and G is a subset of the set of agents):

- there is a public announcement after which φ ;
- there is a public announcement by the agents in group G after which φ ;
- there is an action after which φ ;
- there is a refinement after which φ .

By 'action' we mean epistemic/informative action. A public announcement is such an epistemic action, but there are also other epistemic actions, e.g., private announcements. By 'refinement' we mean the dual of simulation. From the bisimulation requirements, a refinement relation satisfies **atoms** and **back**. We will see that there are subtle differences between 'there is an action' and 'there is a refinement', but that the two come quite close.

The original publication on propositional quantifiers is Fine's [13]. Such an operator quantifies over subsets of the domain of a given structure. Fine distinguishes three different options: quantification (i) over subsets definable by boolean combinations of propositional variables, (ii) over modally definable subsets, and (iii) over any subset.

Example 1



The above (left) structure depicts that agent b knows whether p and that agent a knows that p or is uncertain whether p . Agent b cannot distinguish states s , t , and v (and in that case he knows that p) and he also cannot distinguish states u and w (and in that case he knows that $\neg p$). Agent a knows p in s , or else is uncertain about p . This is a multi-agent Kripke structure where accessibility relations are equivalences, i.e., the ‘arrows’ are assumed to be transitive, symmetric, and reflexive: they are mere links between indistinguishable states. States t and v are also indistinguishable in the stronger sense of being *bisimilar*: they cannot be distinguished in the modal logical language. States u and w are also bisimilar.

The boolean definable subsets of the domain are $\{s, t, u, v, w\}$, $\{s, t, v\}$, and $\{u, w\}$. The modally definable subsets are those and $\{s\}$ (namely by $\Box_a p$) and $\{t, v\}$ (by $p \wedge \neg \Box_a p$). The singleton $\{v\}$ is not modally definable. If we restrict the model to the domain minus v (on the right in the figure), state w has become modally different from state u : w is now the unique state where a knows $\neg p$, and u the unique one where b knows $\neg p$ but a is uncertain about p .

Quantification of type (iii) is undesirable in a setting for information change, as states where agents have indistinguishable beliefs could then after all after become different. Avoiding this is a general requirement for all proposals that we discuss.

An established area of different explorations in propositional quantification are bisimulation quantified logics [34,20], that have also been studied in combination with epistemic logics [15]. That is not the topic of this survey. One might say that we restrict ourselves to quantification over the formula parameters of dynamic modalities (such as ‘ φ ’ in a public announcement $!\varphi$), i.e., quantification over information change. There is overlap. The refinement modal logic of Section 7 can be seen as a refinement quantified logic.

2 Dynamic Epistemic Logic

We present multi-agent epistemic logic, on the general level of a multimodal logic. Its language, structures, and semantics are as follows.

Language

$$\begin{aligned} \mathcal{L} &\ni \varphi ::= p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \Diamond_a \varphi \\ \mathcal{L}(!) &\ni \varphi ::= p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \Diamond_a \varphi \mid \langle !\varphi \rangle \varphi \\ \mathcal{L}(!!) &\ni \varphi ::= p \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid \Diamond_a \varphi \mid \langle !M_s \rangle \varphi \end{aligned}$$

where *propositional variable* p is in a countable set P , *agent* a is in a finite set A , and M_s is a finite *action model* to be defined below: we can see this as an

inductively defined operator with a finite number of arguments of type formula, where this operator is an element of a countable set of action model frames (as in automata PDL). Other propositional connectives are defined by abbreviation and we also define $\Box_a\varphi$ by abbreviation as $\neg\Diamond_a\neg\varphi$ and similarly $[\!|\varphi]\psi$ and $[\!|M_s]\psi$. For $\Diamond_a\varphi$ we read ‘agent a considers φ possible’ and for $\Box_a\varphi$, ‘agent a knows φ ’. For $[\!|\varphi]\psi$ we read ‘after announcement of φ , ψ (is true)’.

Structures. An *epistemic model* $M = \langle S, R, V \rangle$ consists of a *domain* S of *states* (or ‘worlds’), an *accessibility function* $R : A \rightarrow \mathcal{P}(S \times S)$, where each $R(a)$, for which we write R_a (and we may write R_ast for $(s, t) \in R_a$), is an accessibility relation, and a *valuation* $V : P \rightarrow \mathcal{P}(S)$, where each $V(p)$ represents the set of states where p is true. For $s \in S$, a pair (M, s) , for which we write M_s , is an *epistemic state*, also known as a pointed Kripke model. The model class without any restrictions is \mathcal{K} . The class of models where all accessibility relations are equivalence relations is $\mathcal{S5}$.

Let models $M = \langle S, R, V \rangle$ and $M' = \langle S', R', V' \rangle$ be given. A non-empty relation $\mathfrak{R} \subseteq S \times S'$ is a *bisimulation* between M and M' , notation $\mathfrak{R} : M \Leftrightarrow M'$, if for all $(s, s') \in \mathfrak{R}$ and $a \in A$:

atoms $s \in V(p)$ iff $s' \in V'(p)$ for all $p \in P$;

forth if $R_a(s, t)$, then there is a $t' \in S'$ such that $R'_a(s', t')$ and $(t, t') \in \mathfrak{R}$;

back if $R'_a(s', t')$, then there is a $t \in S$ such that $R_a(s, t)$ and $(t, t') \in \mathfrak{R}$.

We write $M_s \Leftrightarrow M'_s$, if there is a bisimulation between M and M' linking s and s' .

Semantics. We first give the semantics for truthful public announcement logic (PAL) [26]. The semantics for action models (the construct $[\!|M_s]\psi$) follows later. Assume an epistemic model $M = \langle S, R, V \rangle$.

$$\begin{aligned} M_s \models p & \quad \text{iff} \quad s \in V_p \\ M_s \models \neg\varphi & \quad \text{iff} \quad M_s \not\models \varphi \\ M_s \models \varphi \wedge \psi & \quad \text{iff} \quad M_s \models \varphi \text{ and } M_s \models \psi \\ M_s \models \Diamond_a\varphi & \quad \text{iff} \quad \text{there is a } t \in S : R_ast \text{ and } M_t \models \varphi \end{aligned}$$

$$M_s \models \langle \!|\psi \rangle \varphi \quad \text{iff} \quad M_s \models \psi \text{ and } (M|\psi)_s \models \varphi$$

where $M|\psi := \langle S', R', V' \rangle$ such that $S' := \{s \in S \mid M_s \models \psi\}$, $R'_a := R_a \cap (S' \times S')$, and $V'(p) := V(p) \cap S'$.

Example 2

$$\begin{array}{ccc} \underline{1} & \xrightarrow{a} & 0 & \xRightarrow{!\!p} & \underline{1} \\ M & & & & M' \end{array}$$

Agent a is uncertain about p (she cannot distinguish state 1 where p is true from state 0 where p is false). The actual state 1 is underlined. After truthful announcement $!\!p$, she knows that p . We have that $M_1 \models \langle \!|\!p \rangle \Box_a p$ because $M_1 \models p$ and $(M|\!p)_1 \models \Box_a p$. Note that we also have, e.g., $M_1 \models p \wedge \neg\Box_a p$, so, strangely, as $\Box_a p$ implies $\neg p \vee \Box_a p$ which is equivalent to $\neg(p \wedge \neg\Box_a p)$ we have that $M_1 \models \langle \!|(p \wedge \neg\Box_a p) \rangle \neg(p \wedge \neg\Box_a p)$.

Axiomatization. The well-known axiomatization of minimal modal logic K contains axiom $\Box_a(\varphi \rightarrow \psi) \rightarrow \Box_a\varphi \rightarrow \Box_a\psi$ and derivation rule ‘From φ infer $\Box_a\varphi$.’ There are no multimodal interaction axioms. One of the axioms involving announcements is $[\!|\psi]\Box_a\varphi \leftrightarrow (\psi \rightarrow \Box_a[\!|\psi]\varphi)$.

Action Models. An action model [7](#) (or event model) is a structure like a Kripke model but with a precondition function instead of a valuation function. An *action model* $M = \langle S, R, \text{pre} \rangle$ consists of a *domain* S of *actions*, an *accessibility function* $R : A \rightarrow \mathcal{P}(S \times S)$, where each R_a is an accessibility relation, and a *precondition function* $\text{pre} : S \rightarrow \mathcal{L}$, where \mathcal{L} is a logical language. A pointed action model M_s is an *epistemic action*.

Performing an epistemic action in an epistemic state means computing their restricted modal product. This product encodes the new state of information. It is defined as follows.

Given an epistemic state M_s where $M = \langle S, R, V \rangle$ and an epistemic action M_s where $M = \langle S, R, \text{pre} \rangle$. Let $M_s \models \text{pre}(s)$. The update $(M \otimes M, (s, s))$ is an epistemic state where $(M \otimes M) = \langle S', R', V' \rangle$ and

$$\begin{aligned} S' &= \{(t, \mathbf{t}) \mid M_t \models \text{pre}(\mathbf{t})\} \\ ((t, \mathbf{t}), (t', \mathbf{t}')) \in R'_a &\text{ iff } R_a t t' \text{ and } R_a \mathbf{t} \mathbf{t}' \\ (t, \mathbf{t}) \in V'(p) &\text{ iff } t \in V(p) \end{aligned}$$

In other words: the domain consists of the product but restricted to state/action pairs (t, \mathbf{t}) such that $M_t \models \text{pre}(\mathbf{t})$, i.e., such that the action can be executed in that state; an agent considers a pair (t, \mathbf{t}) possible in the next epistemic state if she considered the previous state t possible, and the execution of action \mathbf{t} in that state; and the valuations do not change after action execution. (Example [7](#) on page [105](#) illustrates action model execution.)

The action model for truthful public announcement is a singleton action model, with as precondition the announcement formula φ , accessible to all agents. So, public announcement logic is a specific action model logic.

In the language $\mathcal{L}(\!|)$ with action models we associate a dynamic modal operator $\langle \!| M_s \rangle$ to each finite epistemic action M_s . The clause $\langle \!| M_s \rangle \varphi$ is in fact inductive, if we realize that the preconditions of all actions in M (this includes s) are also of type formula. The interpretation of such modal operators for epistemic actions is then as follows.

$$M_s \models \langle \!| M_s \rangle \psi \quad \text{iff} \quad M_s \models \text{pre}(s) \text{ and } (M \otimes M)_{(s,s)} \models \psi$$

Axiomatization. Two crucial axioms of action model logic (AML) are

$$\begin{aligned} [\!| M_s] p &\leftrightarrow (\text{pre}(s) \rightarrow p) \\ [\!| M_s] \Box_a \psi &\leftrightarrow (\text{pre}(s) \rightarrow \bigwedge_{(s,\mathbf{t}) \in R_a} \Box_a [\!| M_{\mathbf{t}}] \psi) \end{aligned}$$

The axiom for knowledge after public announcement in the previous paragraph is a special case of the above. Public announcement logic and also action model logic are equally expressive as multi-agent epistemic logic. Epistemic actions can

be eliminated from formulas by axioms such as the above. They function as reduction rules. The dynamic modality is pushed ever more inward until it gets eliminated by an application of the first axiom.

3 Arbitrary Announcement

Arbitrary public announcement logic (APAL) contains a quantifier over announcements. First, we introduce the languages that will serve us throughout this contribution.

Definition 1 (Language) *To the languages \mathcal{L} , $\mathcal{L}(!)$, and $\mathcal{L}(!!)$ we can add inductive clauses (either) $\blacklozenge\varphi$ or $\blacklozenge_B\varphi$, where $B \subseteq A$. For $\blacklozenge_{\{a\}}\varphi$ we write $\blacklozenge_a\varphi$. We thus get $\mathcal{L}(\blacklozenge)$, $\mathcal{L}(!, \blacklozenge)$, $\mathcal{L}(!!, \blacklozenge)$, $\mathcal{L}(\blacklozenge_B)$, etc.*

The language of APAL is $\mathcal{L}(!, \blacklozenge)$.

Definition 2 (Semantics of arbitrary announcement) *Given are M_s (in \mathcal{K}) and $\varphi \in \mathcal{L}(!, \blacklozenge)$.*

$$M_s \models \blacklozenge\varphi \quad \text{iff} \quad \text{there is a } \psi \in \mathcal{L}(!) \text{ such that } M_s \models \langle !\psi \rangle \varphi$$

The restriction $\psi \in \mathcal{L}(!)$ is important: \blacklozenge quantifies over formulas in the language *without* the \blacklozenge operator, i.e., the ‘ \blacklozenge -free formulas’. Given that public announcement logic and epistemic logic are equally expressive, this means that we quantify over epistemically definable subsets of the given model.

Example 3

$$\begin{array}{ccccc} \underline{1} & \xrightarrow{a} & 0 & \xleftarrow{! \top} & \underline{1} & \xrightarrow{a} & 0 & \xrightarrow{! p} & \underline{1} \\ M'' & & & & M & & & & M' \end{array}$$

Agent a can either make a truly informative announcement $!p$ or a trivial announcement $!\top$. We have that $M_1 \models \blacklozenge\Box_a p$ because $M_1 \models \langle !p \rangle \Box_a p$. On the other hand we have that $M_1 \models \blacklozenge\neg\Box_a p$ because $M_1 \models \langle !\top \rangle \neg\Box_a p$. Of course we do not have $M_1 \models \blacklozenge(\Box_a p \wedge \neg\Box_a p)$.

Validities. On the class $\mathcal{S5}$ an illustrative validity is: $\blacklozenge(\Box_a p \vee \Box_a \neg p)$. This formalizes that the agent a can always learn the value of an atomic proposition. Either p is true, in which case the agent knows it after its announcement (or, in case it already knew that p , *still* knows it after its announcement), or it is false, in which case the agent knows that it is false after the announcement that p is false. Some schematic validities of interest are

$$- \quad \blacklozenge\blacklozenge\varphi \rightarrow \blacklozenge\varphi \quad (\mathbf{4})$$

This expresses that a sequence of two announcements $!\psi$ and $!\chi$ is again an announcement, namely $!\langle !\psi \rangle \chi$ (or $!(\psi \wedge [!\psi]\chi)$).

$$- \quad \blacksquare\varphi \rightarrow \varphi \quad (\mathbf{T})$$

If φ is true after any announcement, it is true after the trivial announcement.

– $\blacklozenge\blacksquare\varphi \rightarrow \blacksquare\lozenge\varphi$ (Church-Rosser)

Given an epistemic state M_s and two (truthful) announcements $!\psi$ and $!\chi$, there are, respectively, two consecutive announcements $!\psi'$ and $!\chi'$ such that the same (or bisimilar) epistemic state results: $(M|\varphi|\varphi')_s \stackrel{\sim}{\leftrightarrow} (M|\psi|\psi')_s$.

– $\blacksquare\lozenge\varphi \rightarrow \lozenge\blacksquare\varphi$ (McKinsey)

In combination with **4** this formalizes a property known as atomicity [8].

The operator \lozenge seems therefore to behave like *S4*. However, this is not the case. It is not even a normal modal operator. For example, $p \rightarrow \lozenge\Box_a p$ is valid but $(p \wedge \neg\Box_a p) \rightarrow \lozenge\Box_a(p \wedge \neg\Box_a p)$ is invalid (see Example [2]).

Axiomatization and Theory. The logic APAL has a complete axiomatization (for the class $\mathcal{S}5$, generalizable to the class \mathcal{K}), is not compact [6], is undecidable [16] (the usual tiling argument applies), and the model checking problem is PSPACE-complete (an application of a result obtained in [2] for the similar logic GAL, group announcement logic). A crucial semantic result is as follows.

Whenever $M_s \models \lozenge\psi$, then there exists a model M' only differing from M in the valuation of atoms not occurring in ψ (including a p) such that $M'_s \models \langle !p \rangle \psi$.

This is not trivial, because the value of a formula $\lozenge\psi$ does not merely depend on the variables occurring in ψ but on the entire set of propositional variables, as \lozenge implicitly quantifies over all propositional variables. This property is also used to prove that APAL is more expressive than epistemic logic. An elegant proof (due to Kooi) is as follows.

Let a formula in $\mathcal{L}(!, \lozenge)$ be given. We may as well assume that it has form $\lozenge\psi$. Suppose there is a $\chi \in \mathcal{L}$ that is logically equivalent to $\lozenge\psi$. This χ has a certain modal depth. Now take two epistemic states identical up to that (finite!) depth but different in some detail beyond there, and a formula φ expressing that difference. Then $\langle !\varphi \rangle \psi$ is true in the one and false in the other epistemic state, and therefore $\lozenge\psi$ as well. On the other hand, χ cannot see that far ahead and must be either true in both epistemic states or false in both epistemic states, in contradiction with the assumption.

The axiomatization for the logic contains an axiom and derivation rule:

$$\begin{array}{ll} \blacksquare\varphi \rightarrow [!\psi]\varphi & \text{where } \psi \text{ is } \lozenge\text{-free} \\ \text{From } \psi \rightarrow [!\chi][!p]\varphi \text{ infer } \psi \rightarrow [!\chi]\blacksquare\varphi & \text{where } p \text{ is not in } \varphi, \chi, \psi \end{array}$$

This derivation rule is the convenient form of a more intuitive infinitary axiom. The infinitary axiom says that if for all ψ you can derive a formula $\eta([!\psi]\varphi)$ (of a shape called ‘necessity form’ [17], containing a unique occurrence $[!\psi]\varphi$), then you can also derive $\eta(\blacksquare\varphi)$. A intermediary finitary axiom infers $\eta(\blacksquare\varphi)$ from $\eta([!p]\varphi)$, where p does not occur in φ — a technique pioneered by Gabbay. The soundness of this intermediate rule follows from the semantic result above that satisfiability of $\lozenge\psi$ implies that of $\langle !p \rangle \psi$ for a p not in ψ . The derivation rule above then simplifies the intermediate rule.

Knowability. The schema $\blacklozenge\Box_a\varphi$, for ‘there is an announcement after which the agent knows φ ’, forms a specific interpretation of ‘ φ is knowable,’ a suggestion made by van Benthem in [28], an interesting setting for Fitch’s knowability paradox [14]. Fitch addressed the question whether what is true can become known. It is considered problematic (paradoxical even) that the existence of unknown truths (there is a φ for which $\varphi \wedge \neg\Box_a\varphi$) is inconsistent with the requirement that all truths are knowable (for any ψ , $\psi \rightarrow \blacklozenge\Box_a\psi$). The inconsistency appears by substituting $p \wedge \neg\Box_a p$ for ψ . Given the Fitch setting, an interesting validity in APAL is $\blacklozenge(\Box_a\varphi \vee \Box_a\neg\varphi)$ [33], because this states that ‘everything is knowable’. This comes at a price: the Moore-sentence $p \wedge \neg\Box_a p$ is knowable in this ‘get to know if true or get to know if false’ sense, because after this is being truthfully announced, agent a knows that it is false (see Example 2).

Open Questions. The reason for the syntax restriction in the semantics of $\blacklozenge\varphi$ is that otherwise the definition is circular. Consider an ‘unlimited’ \blacksquare version: $\blacksquare\varphi$ is true in a state if $[\psi]\varphi$ is true there for all ψ — but that includes $[\blacksquare\varphi]\varphi$. So we need to go down the scale on some complexity measure. The solution in APAL is to go down all the way, ψ has to be \blacklozenge -free. But one could imagine a hierarchy of ever more expressive arbitrary announcement logics, e.g., $\blacksquare\varphi$ is true in a state if $[\psi]\varphi$ is true there for all ψ ‘with strictly less \blacklozenge operators than $\blacksquare\varphi$.’ This suggestion by Baltag was not followed up.

In her PhD thesis [12] Economou proposed various logics with propositional quantification that are at least as expressive as APAL. We conjecture that they are equally expressive. Her work has unfortunately not been published.

The semantics of $\blacklozenge\varphi$ uses public announcement operators. Given the semantics for public announcement it could therefore also be

$$M_s \models \blacklozenge\varphi \text{ iff there is a } \psi \in \mathcal{L} \text{ such that } (M|\psi)_s \models \varphi$$

We call this logic, for the language $\mathcal{L}(\blacklozenge)$ without announcements, *knowability logic*. What is its axiomatization? It is not known. The axiom and rule for \blacklozenge in APAL need to be replaced by something not using announcements. Possibly, this may be syntactic relativization. That would make matters not simpler but harder and even less elegant. An alternative road to success may be an axiomatization without the occurrence of the fresh variable in the derivation rule, where inspiration may be found in the — elegant, we think — axiomatization of refinement modal logic (Section 7).

4 Group Announcement

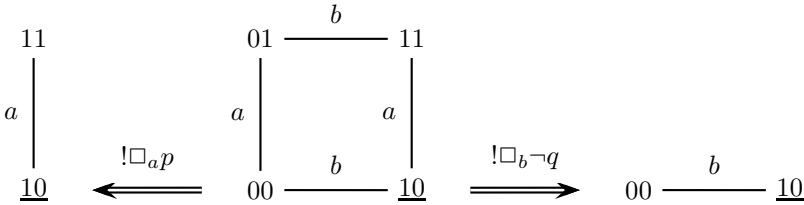
The announcements in truthful public announcement logic are supposed to be made by an outside observer. For that case, ‘truthful’ simply means ‘true’. The outside observer is not modelled as an agent and does not appear in the logical language. If we wish to formalize a truthful public announcement of φ made by an agent a that is modelled in the system, it is common to see this as the announcement $!\Box_a\varphi$. Now, there is a difference between true and truthful. Truthful means

that the agent believes what it announces. A truthful but false announcement by a that φ satisfies $\neg\varphi \wedge \Box_a\varphi$. In *group announcement logic* (GAL) we investigate what can be achieved by simultaneous truthful announcements by a subset of the set of all agents. This includes communication protocols where agents take turns in saying something. If a announces φ and subsequently b announces ψ , we can also see this as: simultaneously a announces φ and b announces \top (i.e., ‘nothing’), and then, simultaneously a announces \top and b announces ψ .

Definition 3 (Semantics of group announcement). Let $B \subseteq A, \varphi \in \mathcal{L}(!, \blacklozenge_B)$.

$$M_s \models \blacklozenge_B\varphi \quad \text{iff} \quad \text{there are } \psi_1 \dots \psi_{|B|} \in \mathcal{L}(!) \text{ such that } M_s \models \langle ! \bigwedge_B \Box_i \psi_i \rangle \varphi$$

Example 4. Given two agents a, b such that a knows whether p and b knows whether q (and this is common knowledge), and let in fact p be true and q be false (the underlined state). Anne (a) can achieve that Bill knows whether p (namely by informing him of the value of p), and Bill (b) can achieve that Anne knows whether q , but *neither* agent can achieve both outcomes at the same time. However, together they can achieve that.



We can evaluate in the square model in the middle that

- $10 \models \blacklozenge_a \Box_b p$ but $10 \not\models \blacklozenge_b \Box_a p$
- $10 \models \blacklozenge_b \Box_a \neg q$ but $10 \not\models \blacklozenge_a \Box_b \neg q$
- $10 \models \blacklozenge_{ab} (\Box_b p \wedge \Box_a \neg q)$ but $10 \not\models \blacklozenge_a (\Box_b p \wedge \Box_a \neg q)$ and $10 \not\models \blacklozenge_b (\Box_b p \wedge \Box_a \neg q)$

Whatever the actual state, a and b can get to know it by ‘collaborating’ in the \blacklozenge_{ab} sense.

A number of validities are ($B, C \subseteq A$)

- $\blacklozenge_B \blacklozenge_C \varphi \rightarrow \blacklozenge_{B \cup C} \varphi$
If an announcement by group B is followed by an announcement by group C , then B and C could have made a joint announcement with the same informative content.
- $\blacklozenge_B \blacklozenge_B \varphi \rightarrow \blacklozenge_B \varphi$
A corollary of the previous.
- $\blacklozenge_B \blacksquare_C \varphi \rightarrow \blacksquare_C \blacklozenge_B \varphi$
Church-Rosser for different groups of agents: let those in B announce something and those in C announce something else, then there is a subsequent C announcement in the first case and a subsequent B announcement in the second case to reach the same epistemic state again, modulo bisimilarity.

This group announcement logic (GAL) has been reported on in [312]. With GAL one can formalize communication protocols, such as security protocols. Let Alice be a sender a , Bob a receiver b , and Eve a spy / eavesdropper e . Let φ be some *information goal*. For example, suppose Alice wishes to inform Bob of the latest transatlantic scandal p , then an information goal could be that Alice, Bob, and Eve commonly know that either Alice and Bob share knowledge of p or Alice and Bob share knowledge of $\neg p$. (This can be more succinctly formalized with common knowledge operators but we bypass these in this paper.) The requirement that not only Alice and Bob but also they and Eve commonly know this, is usual in a security setting. It formalizes that the protocol is known to have terminated: we may assume that everything is public about the protocol except the message (and private keys). There is also a *security goal* ψ that needs to be preserved throughout protocol execution, e.g., Alice, Bob, and Eve commonly know that Eve is ignorant about p (or some more involved aspect of p , such as the identity of those involved in the scandal). A finite protocol for a and b to learn the secret safely should observe

$$\psi \rightarrow \blacklozenge_{ab}(\varphi \wedge \psi)$$

The logic GAL shares various properties with APAL, e.g., the axiomatization is similar and the method to prove completeness, and the model checking complexity is PSPACE-complete.

Agency. The expression $\blacklozenge_G \varphi$ has the smell of ‘group of agents G is able to achieve φ ’, such that, taking a single agent, $\Box_a \blacklozenge_a \varphi$ (on $\mathcal{S5}$ models) seems to formalize that agent a knows that she is able to achieve φ , as in logics combining agency and knowledge [211,30]. These are tricky issues in the setting for group announcements. For example:

- $\Box_a \blacklozenge_a \varphi \rightarrow \blacklozenge_a \varphi$ is valid
‘If you know that you can do something, you can do it.’
- $\blacklozenge_a \Box_a \varphi \rightarrow \Box_a \blacklozenge_a \varphi$ is valid
This is known as ‘knowledge de re implies knowledge de dicto’.
- $\Box_a \blacklozenge_a \varphi \rightarrow \blacklozenge_a \Box_a \varphi$ is not valid (‘knowl. de dicto does not imply knowl. de re’)
The problem is that in different states different announcements may be required to make φ true. As you do not know what the actual state is, you therefore do not know what announcement makes φ true in the actual state. You only know that in all states that you consider possible there is an announcement that makes φ true. For example, in state s formula φ is true after you announce p , but not after you announce q ; in indistinguishable state t formula φ is true after you announce q , but not after you announce p . Should you announce p or should you announce q ? You are not able to achieve φ !

Open Problems. The exact shape of group announcement specifications such as the finite protocol specification $\psi \rightarrow \blacklozenge_{ab}(\varphi \wedge \psi)$ are of some interest.

Group announcement logic GAL seems only a version of APAL. With group announcements one can only get model restrictions that are (for the $S5$ case) an intersection of unions of equivalence classes. E.g., in Example 4 one cannot get the submodel with domain $\{00, 10, 11\}$. This can be used in an expressivity argument to show that APAL is strictly more expressive than GAL. But it is unknown if GAL is strictly more expressive than APAL. For example, it is not known if the existence of a finite two-agent protocol specification as above is formalizable in APAL.

The logic GAL is probable also undecidable, but the tiling argument used for APAL does not work with announcements known by agents.

5 Coalition Announcement

Another variation on APAL is coalition announcement logic (CAL). In group announcement logic (GAL), we investigate the consequences of the simultaneous announcement (joint public event) by G . The agents not in G do not take part in the action. In CAL we quantify over what the agents in G can achieve by their joint announcement, no matter what the other agents simultaneously announce. The semantics therefore should be clear. The language is the same as for GAL: $\mathcal{L}(!, \blacklozenge_G)$ — to distinguish the two \blacklozenge_G operators we write $\blacklozenge_G^{\text{cal}}$ for the coalitional version. Properties of CAL are again similar to those of APAL.

Definition 4 (Semantics of coalition announcement). *Let $B \subseteq A$.*

$$M_s \models \blacklozenge_B^{\text{cal}} \varphi \quad \text{iff} \quad \text{there are } \psi_1, \dots, \psi_{|B|} \text{ such that for all } \chi_1, \dots, \chi_{|A \setminus B|}, \\ M_s \models \bigwedge \square_i \psi_i \text{ and } M_s \models [!(\bigwedge \square_i \psi_i \wedge \bigwedge \square_j \chi_j)] \varphi$$

Example 5. Consider the four-state model of Example 4. Although we have that $10 \models \blacklozenge_a \neg \square_a \neg q$ (namely by simply doing nothing / announcing \top), it is **not** the case that $10 \models \blacklozenge_a^{\text{cal}} \neg \square_a \neg q$: agent b can prevent a from remaining ignorant by announcing $\neg q$. We still have that $10 \models \blacklozenge_{ab}^{\text{cal}} (\square_b p \wedge \square_a \neg q)$ — but this is trivial, as there are no other agents around to say something to prevent it. The power of group announcements by all agents is the same as that of an announcement by the grand coalition.

The logic CAL is summarily discussed in [3], but no complete axiomatization is given. A complete axiomatization is given for a much related logic in [24], that also deals with many other aspects of such agency. Instead of a dynamic epistemic setting, [24] is an epistemic PDL-style dynamic setting that also involves factual change. Such PDL action labels have preconditions that are the otherwise announced formulas. Another difference is that simultaneous announcements by the agents in group B need not be known by those agents.

The logics GAL and CAL are related in interesting ways. If the grand coalition A can achieve φ in GAL, then obviously as well in CAL, as there are no remaining agents to counteract it (Example 5):

$$\blacklozenge_A \varphi \rightarrow \blacklozenge_A^{\text{cal}} \varphi$$

On the other hand, if the empty coalition \emptyset can achieve φ in CAL, then φ will be true after any announcement (APAL):

$$\blacklozenge_{\emptyset}^{\text{cal}}\varphi \rightarrow \blacksquare\varphi$$

Pauly's Coalition Logic [25] can be embedded in CAL. This includes the principles:

$$\begin{aligned} (S) \quad & (\blacklozenge_B^{\text{cal}}\varphi \wedge \blacklozenge_C^{\text{cal}}\varphi') \rightarrow \blacklozenge_{B \cup C}^{\text{cal}}(\varphi \wedge \varphi') && \text{if } B \cap C = \emptyset \\ (N) \quad & \blacksquare_{\emptyset}^{\text{cal}}\varphi \rightarrow \blacklozenge_A^{\text{cal}}\varphi \end{aligned}$$

and [3] also contains a summary but original interpretation of neighbourhood semantics for a public announcement setting.

Open Problems. The relative expressivity of APAL, GAL and CAL remains unclear. The only obvious result is that APAL is strictly more expressive than GAL and than CAL. Of particular interest is whether CAL is definable in GAL. The *forcing* operator $\{B\}$ proposed by van Benthem in [27] (and in [25]) quantifies over sequences of moves/actions. Schema $\blacklozenge_B^{\text{cal}}\blacklozenge_B^{\text{cal}}\varphi \rightarrow \blacklozenge_B^{\text{cal}}\varphi$ should be valid in CAL (similar to GAL). Therefore $\blacklozenge_B^{\text{cal}}$ models what coalition B can achieve/force in any finite sequence of moves: the extensive game setting of van Benthem. A quick way to that goal would be to show that coalition announcement $\blacklozenge_B^{\text{cal}}\varphi$ is definable in GAL. The conjectured definition is

$$\blacklozenge_B^{\text{cal}}\varphi \leftrightarrow \blacklozenge_B \blacksquare_{A \setminus B} \varphi$$

In other words: if coalition B can achieve φ no matter what the other agents announce *at the same time*, then coalition B can achieve φ no matter what the other agents announce *afterwards*.

6 From Arbitrary Announcements to Arbitrary Events

In the logic APAL we quantify over announcements. How about generalizations? As the effect of an announcement is a model restriction (take a subdomain of the domain, and restrict the valuation and accessibility relations to that subdomain), we can in the first place think of *other* restrictions of the model. In [35] a quantification is proposed over restrictions of the accessibility relation. We will get to that below.

A similar path but towards an even further generalization is as follows. In our multi-agent setting, an announcement is a *public* event. By this we mean that all agents observe the event similarly. Carlos' announcement 'Cordoba is in Spain' is assumed to be heard by all agents present, and they can all assume that they all know that they all hear this, and so on.

A non-public event is Carlos privately saying to you 'Cordoba is in Spain' while I am in another room. Now, there are complications. Firstly, the beliefs of the other agents may become incorrect. I still believe that you were not informed that Cordoba is in Spain, and therefore incorrectly believe that you do not know that. Secondly, lack of synchronization is an issue. In dynamic epistemic logic it

is usual to identify the observed execution of an action with a tick of the clock. So the unobservant agents are therefore running behind.

In events that are sometimes called semi-public all agents know that something happened, but they may have a different perspective on the action. Carlos writes on a piece of paper whether Cordoba is in Spain (or not), folds it, puts it in a letter, and hands me the letter in public telling me it contains the truth about p , and then I open it in public. Now everybody knows that I know the truth about p , but they still do not know if p is true or false. (If the alternatives in semi-public events are mutually exclusive, this can be modelled as restriction of accessibility relations, the first mentioned variation.)

Increasingly more complex scenarios can be worked out. Instead of quantifying over announcements, public events, we can quantify over a more general class of events, that includes announcements as a special case. The usual suspect is then to quantify over the action models of Section 2.

Unlike the previous sections, this section mainly consists of partial results and open problems.

Restriction of Accessibility Relations. In [35], Wen *et al.* assume that accessibility relations R_a are equivalence relations \sim_a (and where \sim is a function that assigns such an equivalence relation to each agent). A restriction of an accessibility relation that is an equivalence relation is also known as a refinement of the partition induced by the equivalence relation. In other words, $\sim'_a \subseteq \sim_a$ means that \sim'_a is a refinement of \sim_a . We write $\sim' \subseteq \sim$ for ‘for all agents a , $\sim'_a \subseteq \sim_a$ ’.

Definition 5 (Semantics of quantification over restr. of access. rel.).

Let $M = \langle S, \sim, V \rangle$ and $\varphi \in \mathcal{L}(!, \blacklozenge)$ be given.

$$M, s \models \blacklozenge\varphi \text{ iff there is a } M' \text{ that is as } M \text{ but with } \sim' \subseteq \sim, \text{ s.t. } M', s \models \varphi$$

Some of the semi-public actions mentioned above are partition refining, e.g., the action in the envelope example. The authors discuss alternative solutions to the knowability paradox in this setting.

We recall how Fine distinguished different forms of quantification, as presented in Section 1, and that quantifying over all submodels allows us to distinguish bisimilar states. This is also major disadvantage of mere restriction of accessibility. The \blacklozenge operator of Definition 5 does not preserve bisimilarity of epistemic states. We think this can easily be repaired by making the deletion of arrows (refinement of equivalence classes) dependent on logical conditions. For example, only delete arrows that start in states where some formula φ is true, or only delete arrows that finish in states where some such φ is true, as in [23].

Quantifying over Action Models. The logic AAML (arbitrary action model logic), interpreted over $\mathcal{L}(!, \blacklozenge)$, has a clear semantics (as stipulated in [6, p.329-330]).

Definition 6 (Semantics of arbitrary events).

$$M_s \models \blacklozenge\varphi \text{ iff there is a } M_s \text{ with } \blacklozenge\text{-free preconditions, such that } M_s \models \langle !M_s \rangle \varphi$$

In a manuscript by Balbiani and van Ditmarsch an axiomatization is proposed. We recall the axiom and derivation rule for \blacklozenge in APAL:

$$\begin{array}{ll} \blacksquare\varphi \rightarrow [!\psi]\varphi & \text{where } \psi \text{ is } \blacklozenge\text{-free} \\ \text{From } \psi \rightarrow [!\chi][!p]\varphi \text{ infer } \psi \rightarrow [!\chi]\blacksquare\varphi & \text{where } p \text{ is not in } \varphi, \chi, \psi \end{array}$$

The axiomatization for AAML consists of ‘the usual suspects’ for the reduction and composition of action models (see Section 2), plus the following axiom and — tentatively proposed — rule. There is no completeness proof.

$$\begin{array}{ll} \blacksquare\varphi \rightarrow [!M_s]\varphi & \text{where all preconditions in } M \text{ are } \blacklozenge\text{-free} \\ \text{From } \psi \rightarrow [!M_s][!p]\varphi \text{ infer } \psi \rightarrow [!M_s]\blacksquare\varphi & \text{where } p \text{ not in } \varphi, \psi \text{ or precondition of } M \end{array}$$

The axiom is obviously correct. The rule could well be another (necessity) form wherein an occurrence of $[!p]\varphi$ is replaced by an occurrence of $\blacksquare\varphi$. Assuming the rule is correct, the surprising aspect is that *public announcement* of variable p is then sufficient to ‘witness’ the execution of *any* action model. This applies a result from [31]: the execution of an action model in a given model M_s has the same effect as the announcement of fresh propositional variable p in a model that is bisimilar to M_s except for p (and such that the value of p does not affect the value of the φ evaluated in M_s). On the level of the proof system, this might mean that it is sufficient to have a derivation from a fresh atom (for any model) in order to get it from any action model, no matter how complex.

Any Action That Satisfies φ . A public announcement $!\varphi$ is an event similarly perceived by all agents with execution precondition φ . The corresponding action model is a singleton, accessible to all agents, with, obviously, precondition φ . Consider *any* epistemic action with precondition φ . For example, any epistemic action such that Cordoba is in Spain. *Any* communicative act is fine, such as lying, deceiving, and private announcement. As long as Cordoba is in Spain. This seems an interesting form of quantification. (It is somewhat similar to the quantification described by de Lima in [24, p.110]: what holds if the agents in group B make a simultaneous announcement $\varphi_1 \wedge \dots \wedge \varphi_n$, no matter what the remaining agents simultaneously announce.) We reuse the notation $!\varphi$ for this φ -satisfying epistemic action operator. It should obey that $M_s \models \langle !\varphi \rangle \psi$ iff there is a M_s such that $M_s \models \langle !M_s \rangle \psi$ (from $M_s \models \langle !M_s \rangle \psi$ also follows that $M_s \models \text{pre}(s)$). More succinctly, where $\blacklozenge\varphi$ is the quantification over action models of this section:

Definition 7 (Quantifying over actions that satisfy φ)

$$M_s \models \langle !\varphi \rangle \psi \quad \text{iff} \quad M_s \models \varphi \wedge \blacklozenge\psi$$

Such a quantification was suggested by Aucher in [5], based on the approach presented in [4] where instead of action models an action language is presented, on a par with the static language (see Section 8). An intriguing question is if we can axiomatize this logic in the language $\mathcal{L}(!)$, without action models and without \blacklozenge .

More Open Problems. The logic for arbitrary action models lacks an axiomatization, but another open problem is whether this logic is decidable. The expectation of the reader might be that, as APAL is undecidable, AAML is also undecidable. But the mere fact that AAML quantifies over a larger set of objects does not mean that that logic is therefore also undecidable. Refinement modal logic (RML), presented in Section 7 is *on finite models* equivalent to AAML. And RML is decidable. So why not AAML as well?

Group announcement logic GAL and coalition announcement logic CAL can be generalized to action models — such very similar logics would clearly benefit from a complete axiomatization of AAML. We should then require that the preconditions of all actions in the action models take the form $\bigwedge_{a \in B} \Box_a \varphi_a$ (where all φ_a are \blacklozenge -free). This would give us a more general meaning to ‘ability of a group B ’. For example, consider agents Anne, Bill, Cath, and Dave. Anne and Bill know p but Cath and Dave don’t. The group consisting of Anne and Bill has the ability to achieve that:

Cath knows p but is uncertain if Anne knows p , and Dave knows p but is uncertain if Bill knows p .

The way to achieve that is:

Anne privately informs Dave that p and Bill privately informs Cath that p .

We cannot achieve the stipulated postcondition by a public announcement! Then, Cath and Dave would have common knowledge that Anne and Bill know p .

7 Refinement Modal Logic

In arbitrary public announcement logic the quantification \blacklozenge is over announcements $!\varphi$, on the level of the syntax. But as we have seen there is a corresponding structural operation: model restriction to the φ -states. Another structural operation used to quantify was restriction of the accessibility relation. The particular quantification we presented was deficient, because it was not bisimulation preserving. But we can make it bisimulation preserving again by making the restrictions dependent on logical (formula) conditions. Arbitrary action model logic quantifies over pointed action models M_s . As such action models are parameters in the language, this is again a syntactic way to define the quantification (their execution as the restricted modal product is relative to action preconditions, that are formulas). Wouldn’t there be a more purely semantic way to define quantification over information change? Somewhat surprisingly, there is. It is called refinement.

Bisimulation determines when two structures have the same informative content. Bisimilar states have the same valuation (**atoms**), and for all agents every step (arrow in the accessibility relation for that agent) that you do from one state, can be matched by a step that you do from the other state (**forth**), and vice versa (**back**). *Simulation* is widely used in computer science. We now only

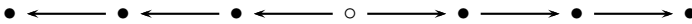
require **atoms** and **forth**. (Typically, simulation only requires inclusion of propositional variables: all variables true in the origin should be true in the image. We require that for all literals.) We can think of a structure M'_s , that is simulating a structure M_s as containing all the structural properties of M *but maybe more*: in M' we can match all steps we do in M but there may be unmatched steps in M' — **back** is not required. Now consider doing simulation in the other direction. We require only **atoms** and **back**. This direction is called *refinement*. It corresponds to structural loss instead of structural gain. But more structure means more uncertainty and less structure means less uncertainty. So, less uncertainty is more information. (Only for certain ‘positive’ aspects of information it is real gain.) This suggests that existentially quantifying over informative change means that there is a refinement of an epistemic model.

A model restriction due to an announcement is an example: given an M and some restriction $M|\varphi$, every state in the restriction has an original in M , and every step one does in the restriction can be matched by the same step in M . But not vice versa! Steps in M to $\neg\varphi$ -states cannot be matched. So we have **back**, but not **forth**. And trivially we have **atoms**, because states in the model restriction do not change the value of atoms.

Example 6. Consider the following structure, for an anonymous agent (unlabeled modality). The \circ state is the designated point. The arrows can be associated with a modality.



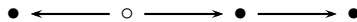
E.g., $\diamond\diamond\diamond\square\perp$ is true in the point. From the point of view of the modal language, this structure is essentially the same structure (it is bisimilar) as



This one also satisfies $\diamond\diamond\diamond\square\perp$, and any other modal formula that is true in the first one as well, for that matter. A more radical structural transformation would be to consider submodels, such as



A distinguishing formula between the two is $\diamond\diamond\square\perp$, which is true here and false above. Can we consider other ‘submodel-like’ transformations that are *neither* bisimilar structures *nor* strict submodels? Yes, we can. Consider



It is neither a submodel of the initial structure, nor is it bisimilar. It satisfies the formula $\diamond\square\perp \wedge \diamond\diamond\square\perp$ that certainly is false in any submodel. It is a refinement of the initial structure.

There still is a ‘submodel-like’ relation with the original structure. Look at its bisimilar duplicate, the one with seven states. The last structure is a submodel

of that copy. Such a relation always holds: a refinement of a given structure can always be seen as the model restriction of a bisimilar copy of that structure.

We can think of the restriction as the result of the public announcement of a variable p . But that variable has then first to be made true exactly in the four states in the restriction and false elsewhere! This we can achieve, with respect to the original structure, by choosing a duplicate that is bisimilar *except for p* , and such that p is made true on the four states of the later restriction. In other words: refinement is bisimulation except for p , followed by model restriction to p . In syntactic terms: refinement quantification is bisimulation quantification followed by relativization.

The logic based on the refinement operation has been presented in [31,32,9,19,18]. It is also known as ‘future event logic’. The language of this logic is again $\mathcal{L}(\blacklozenge_B)$ (but in an entirely different usage of this parameter B than that in GAL and CAL). For \blacklozenge_A we write \blacklozenge .

Definition 8 (Refinement). *Given are epistemic states M_s and $M_{s'}$, and let $B \subseteq A$. A relation between the domains of M and M' that satisfies (i) **atoms**, (ii) **back** for all agents in B , and (iii) **forth** and **back** all agents in $A \setminus B$ is a B -refinement. Epistemic state $M_{s'}$ is then (also called) a B -refinement of M_s (i.e., given that (s, s') is in the relation), and we write $M_s \leq_B M_{s'}$. An A -refinement we call a refinement (plain and simple) and for $\{a\}$ -refinement we write a -refinement.*

Definition 9 (Semantics of the refinement modality). *Assume an epistemic model $M = (S, R, V)$. Let $B \subseteq A$.*

$$M_s \models \blacklozenge_B \varphi \text{ iff for all } M_{s'} : M_s \leq_B M_{s'} \text{ implies } M_{s'} \models \varphi$$

We have the validities $\blacklozenge_a \blacklozenge_b \varphi \leftrightarrow \blacklozenge_b \blacklozenge_a \varphi$ and also $\blacklozenge_a \blacklozenge_b \varphi \leftrightarrow \blacklozenge_{ab} \varphi$. It is therefore sufficient to give results for a single agent \blacklozenge_a . In this refinement modal logic (RML) we see some by now familiar validities.

- $\blacksquare_a \varphi \rightarrow \varphi$ (reflexivity)
- $\blacklozenge_a \blacklozenge_a \varphi \rightarrow \blacklozenge_a \varphi$ (transitivity)
- $\blacklozenge_a \blacksquare_a \varphi \rightarrow \blacksquare_a \blacklozenge_a \varphi$ (Church-Rosser)

The \blacklozenge_a operator can also be seen as implicit quantification over a propositional variable, just as in bisimulation quantified logics we have explicit quantification over propositional variables. The bisimulation variation except for one variable followed by a model restriction in Example 6 is a general result. Refinement quantification is bisimulation quantification plus relativization (modulo an inductively defined translation), for \blacklozenge_A (\blacklozenge) we get:

$$\blacklozenge \varphi \text{ is equivalent to } \exists p \varphi^p$$

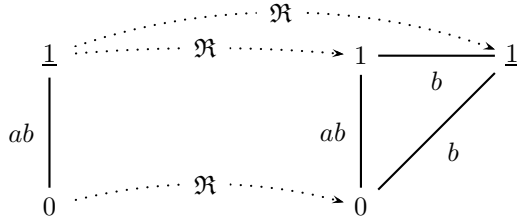
The chosen notion of relativization also matters: given the primitive $\blacklozenge_a \varphi$, the result had to be obtained using the arrow elimination semantics for announcement, not the state elimination semantics.

Action Models and Refinement. There is a strong link between AAML (arbitrary action model logic) and RML: executing an epistemic action in an epistemic state produces a refinement of that epistemic state, and, dually, for every refinement of a *finite* epistemic state there is an epistemic action such that the result of its execution in that pointed model is a model bisimilar to the refinement [31, Prop.4,5]. It is instructive to outline the proof of these results.

Given pointed model M_s and epistemic action M_s , the resulting $(M \otimes M)_{(s,s)}$ is a refinement of M_s by way the relation \mathfrak{R} consisting of all pairs $(t, (t, \mathbf{t}))$ such that $M_t \models \text{pre}(\mathbf{t})$. Some states of the original model may get lost in the modal product, namely if there is no action whose precondition can be executed there. But all ‘surviving’ (state,action)-pairs simply can be traced back to their first argument: clearly a refinement.

For the other direction, construct an epistemic action $M_{s'}$ that is isomorphic to a given refinement $N_{s'}$ of a model M_s , but wherein valuations (determining the value of propositional variables) in states $t \in N$ are replaced by preconditions for action execution of the corresponding action points (also called) t . Precondition $\text{pre}(t)$ should be satisfied in exactly those states $s \in M$ such that $(s, t) \in \mathfrak{R}$, where \mathfrak{R} is the refinement relation linking M_s and $N_{s'}$. Now in a *finite* model, we can single out these states by a distinguishing formula [10]. One then shows that $(M \otimes M)_{(s,s')}$ is bisimilar to $N_{s'}$. It is unknown if the finiteness restriction can be lifted.

Example 7. We illustrate that action model execution is refinement and vice versa. Two agents a, b are uncertain about the value of a (true) fact p . An informative event is possible after which a knows that p but b does not know that. The initial state of information is on the left, and its refinement validating the postcondition is on the right. (Actual states are underlined.)



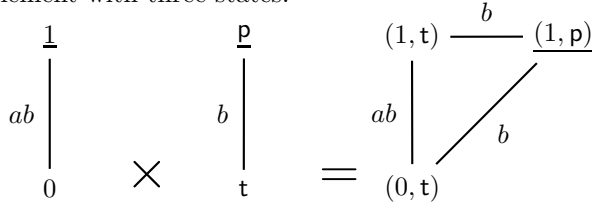
On the left, the formula $\blacklozenge(\Box_a p \wedge \neg \Box_b \Box_a p)$ is true, because $\Box_a p \wedge \neg \Box_b \Box_a p$ is true on the right. On the right, in the actual state there is no alternative for agent a (only the actual state itself is considered possible by a), so $\Box_a p$ is true, whereas agent b also considers another state possible, wherein agent a considers it possible that p is false. Therefore, $\neg \Box_b \Box_a p$ is also true in the actual state on the right.

The model on the right in the figure is neither an a -refinement of the model on the left, nor a b -refinement of it, but an $\{a, b\}$ -refinement. The right model is *not* bisimilar to the left model, e.g., from the underlined 1-state on the right a cannot access a state where p is false.

Now we produce the right model as the execution of an epistemic action! The epistemic action consists of two action points \mathbf{t} and \mathbf{p} , they can be distinguished

by agent a but not by agent b . What really happens is p ; it has precondition p . Agent b cannot distinguish this from t with precondition \top . In words: agent a learns that p is true, but agent b is uncertain if that has happened or not (i.e., or if nothing has happened).

The execution of this action is below. The point of the structure is the one with precondition p : in fact, a is learning that p , but b is uncertain between that action and the ‘trivial’ action wherein nothing is learnt. The trivial action has precondition \top . It can be executed in both states of the initial model. The actual action can only be executed in the state where p is true. Therefore, the resulting structure is the refinement with three states.



Axiomatization and Theory. The axiomatization RML has the following axioms involving \blacklozenge_a .

$$\begin{array}{l}
 \blacklozenge_a \nabla_a \Phi \leftrightarrow \bigwedge_{\varphi \in \Phi} \blacklozenge_a \blacklozenge_a \varphi \\
 \blacklozenge_a \nabla_b \Phi \leftrightarrow \nabla_b \{ \varphi \in \Phi \mid \blacklozenge_a \varphi \} \\
 \blacklozenge_a \bigwedge_{b \in B} \nabla_b \Phi^b \leftrightarrow \bigwedge_{b \in B} \blacklozenge_a \nabla_b \Phi^b
 \end{array}
 \quad \text{where } a \neq b$$

This uses the *cover* operator ∇_a . By abbreviation, $\nabla_a \Phi$ is defined as $\bigwedge_{\varphi \in \Phi} \blacklozenge_a \varphi \wedge \square_a \bigvee_{\varphi \in \Phi} \varphi$. The single-agent version of the axiomatization only contains the first axiom. Consider the case where $\Phi = \{\varphi_1, \varphi_2\}$. The first axiom then says that if there is a refinement with two accessible states, of which one validates φ_1 and the other φ_2 (so that we have $\blacklozenge_a \varphi_1$ and $\blacklozenge_a \varphi_2$ and $\square_a(\varphi_1 \vee \varphi_2)$), then already in the initial model there should be two accessible states such that after a refinement φ_1 is true in one and φ_2 is true in the other (see also Example 8, next).

All these axioms are equivalences. This seems a recipe to eliminate by rewriting the refinement quantifiers from the epistemic language. And that is indeed the case! In other words, the axiomatization is complete, the logic RML is equally expressive as the base epistemic (or rather: multi-agent modal) logic, and therefore it is evidently also decidable, unlike APAL (and unlike AAML?). Such nice results do not generalize straightforwardly to other model classes than \mathcal{K} . The problem is that the ‘implicit’ bisimulation aspect in the semantics makes it very relevant over what class of models the quantification is. The above axioms may be invalid for other model classes.

Example 8. On the class $S5$ the first axiom is invalid! Consider $\blacklozenge_a \nabla_a \{ \square_a p, \neg \square_a p \}$ and $\blacklozenge_a \blacklozenge_a \square_a p \wedge \blacklozenge_a \blacklozenge_a \neg \square_a p$. The first is inconsistent for $S5$. It implies that there is a refinement in which $\blacklozenge_a \square_a p$ and $\blacklozenge_a \neg \square_a p$ are both true. In $S5$ this is equivalent to both $\square_a p$ and $\neg \square_a p$ being true, an inconsistency: you cannot know and not know p at the same time! But the second is very conceivable, if you are uncertain about p , you consider it possible that you are informed about p , after

which you know p , but you also consider it possible that you are kept in the dark, after which you still do not know that p . So, there is no equivalence there, this axiom does not hold for class $S5$.

Open Problems. There are a number of *results* on complexity and succinctness for a version of refinement modal logic for the modal μ -calculus, for this see [9] (refinement μ -calculus is non-elementary). The complexity of RML is reported complete for $AEXP_{pol}$ in a manuscript by Bozzelli, van Ditmarsch and Pinchinat ($AEXP_{pol}$ is the class of problems solvable by alternating Turing machines running in single exponential time but only with a polynomial number of alternations). French and Hales are investigating refinement modal logic for various other classes than the results now available (for \mathcal{K} , $\mathcal{KD45}$, and, recently, $S5$): there are worthwhile results to obtain for axiomatization, complexity, and expressivity. It seems also worthwhile to investigate refinement in other modal settings, e.g., refinement CTL and refinement PDL.

8 Other Variations and Conclusions

We presented a fair variety of quantifiers over information change. Fair, we hope, but clearly not all. In the course of our investigations we came across other options of independent interest, and of which the relation to the proposals presented in this contribution, that are mainly *our* proposals, is unclear. Consider the following.

Difference Operator. In the *sabotage games* proposed by van Benthem [29] a network with multiple connections between nodes is given, a player called Runner attempts to travel between two given nodes in the network and a player Blocker attempts to sabotage the first player by removing nodes in the network after every move of Runner. If Runner cannot reach his destination, Blocker wins. Otherwise, Runner wins. The focus of [29] is on the complexities of solving such games (determine the winner). Van Benthem also proposes a modal logical setting [29, p.271], as follows. We can think of removing a link between states/destinations as the elimination of a state/world in a corresponding Kripke model, such that we have:

$$M_s \models \blacklozenge\varphi \quad \text{iff} \quad \text{there is a } t \neq s \text{ such that } (M - t)_s \models \varphi$$

where $M - t$ is the model restriction of M to the domain minus t . As far as we know, the axiomatization of this logic is an open question. Surely there is a relation with the difference operator proposed by de Rijke [11].

Action Language Quantifiers. In works as [4,5] Aucher proposes a dynamic epistemic logic with an action language instead of action models. The idea is that the standard modalities can have both a static and dynamic interpretation. Similar ideas were proposed by Kooi in [22, Section 4.4.1], the ‘action language logic’ ALL, and the roots go back to [7]. Instead of having pointed action models

as epistemic actions, consider having action language expressions as epistemic actions. The base language is again \mathcal{L} , multi-agent epistemic logic, but now interpreted on action models. The action formula p is true on all action models M_s such that $\text{pre}(s) = p$, but $\diamond_a q$ is true if agent a considers an alternative action possible, accessible from s in M , with precondition q . The quantifier proposed in Definition 7, over all action models with precondition φ , is the action language formula satisfying φ . Just like that. Consider $\neg p \wedge \Box_a p$, as an action language expression. This stands for all actions satisfying $\neg p$ where a incorrectly believes that the action satisfies p is executed. That formula holds in a two-action action model where you are lying to a that p , but also in any action model that contains that as a substructure (so, in a way, in any action model simulating the two-action minimal one). The relation to operators like \blacklozenge , that quantify in a way unrelated to any specific formula, is as yet unclear.

More and more quantifiers over information change... As Ramanujam says: “These are all submodel operators. What are their properties?” Well, the ones in this contribution are fairly $S4$ -like, and satisfy $\blacklozenge\blacklozenge\varphi \rightarrow \blacklozenge\varphi$. And also $\blacksquare\varphi \rightarrow \varphi$. But if we were to quantify over protocol execution logics, 4 would not hold. It is a bit unclear what properties should always hold. The complexity picture for model checking and for satisfiability is rather incomplete. Overviews of complexities of dynamic epistemic logics such as the notes by Yanjing Wang are eagerly expected. We encourage any reader to contribute to the further development and completion of this new frontier in dynamic epistemic logics. This story is to be continued.

References

1. Ågotnes, T.: Action and knowledge in alternating-time temporal logic. *Synthese* 149(2), 377–409 (2006)
2. Ågotnes, T., Balbiani, P., van Ditmarsch, H., Seban, P.: Group announcement logic. *Journal of Applied Logic* 8, 62–81 (2010)
3. Ågotnes, T., van Ditmarsch, H.: Coalitions and announcements. In: *Proceedings of 7th AAMAS*, pp. 673–680. IFAAMAS (2008)
4. Aucher, G.: Characterizing updates in dynamic epistemic logic. In: *Proceedings of Twelfth KR*. AAAI Press (2010)
5. Aucher, G.: DEL-sequents for regression and epistemic planning. *Forthcoming in Journal of Applied Non-Classical Logics* (2012)
6. Balbiani, P., Baltag, A., van Ditmarsch, H., Herzog, A., Hoshi, T., De Lima, T.: Knowable as known after an announcement. *Review of Symbolic Logic* 1(3), 305–334 (2008)
7. Baltag, A., Moss, L., Solecki, S.: The logic of public announcements, common knowledge, and private suspicions. In: *Proceedings of the 7th TARK*, pp. 43–56 (1998)
8. Blackburn, P., de Rijke, M., Venema, Y.: *Modal Logic*. Cambridge Tracts in Theoretical Computer Science, vol. 53. Cambridge University Press, Cambridge (2001)
9. Bozzelli, L., van Ditmarsch, H., French, T., Hales, J., Pinchinat, S.: Refinement modal logic, <http://arxiv.org/abs/1202.3538>
10. Browne, M., Clarke, E., Grümberg, O.: Characterizing Kripke Structures in Temporal Logic. In: Ehrig, H., Levi, G., Montanari, U. (eds.) *CAAP 1987 and TAPSOFT 1987*. LNCS, vol. 249, pp. 256–270. Springer, Heidelberg (1987)

11. de Rijke, M.: The modal logic of inequality. *Journal of Symbol Logic* 57, 566–584 (1992)
12. Economou, P.: *Extensions and Applications of Dynamic Epistemic Logic*. PhD thesis, Oxford University (2010)
13. Fine, K.: Propositional quantifiers in modal logic. *Theoria* 36(3), 336–346 (1970)
14. Fitch, F.B.: A logical analysis of some value concepts. *The Journal of Symbolic Logic* 28(2), 135–142 (1963)
15. French, T.: *Bisimulation quantifiers for modal logic*. PhD thesis, University of Western Australia (2006)
16. French, T., van Ditmarsch, H.: Undecidability for arbitrary public announcement logic. In: *Advances in Modal Logic* 7, pp. 23–42. College Publications (2008)
17. Goldblatt, R.: *Axiomatising the Logic of Computer Programming*. Springer (1982)
18. Hales, J.: *Refinement quantifiers for logics of belief and knowledge*. Honours Thesis, University of Western Australia (2011)
19. Hales, J., French, T., Davies, R.: Refinement quantified logics of knowledge. *Electr. Notes Theor. Comput. Sci.* 278, 85–98 (2011)
20. Hollenberg, M.: *Logic and bisimulation*. PhD thesis, University of Utrecht (1998)
21. Jamroga, W., van der Hoek, W.: Agents that know how to play. *Fundamenta Informaticae* 63, 185–219 (2004)
22. Kooi, B.: *Knowledge, Chance, and Change*. PhD thesis, University of Groningen, ILLC Dissertation Series DS-2003-01 (2003)
23. Kooi, B., Renne, B.: Arrow update logic. *Review of Symbolic Logic* 4, 536–559 (2011)
24. de Lima, T.: *Alternating-Time Temporal Announcement Logic*. In: Leite, J., Torroni, P., Ågotnes, T., Boella, G., van der Torre, L. (eds.) CLIMA XII 2011. LNCS, vol. 6814, pp. 105–121. Springer, Heidelberg (2011)
25. Pauly, M.: A modal logic for coalitional power in games. *Journal of Logic and Computation* 12(1), 149–166 (2002)
26. Plaza, J.A.: *Logics of public communications*. In: *Proceedings of the 4th International Symposium on Methodologies for Intelligent Systems: Poster Session Program*, pp. 201–216. Oak Ridge National Laboratory (1989)
27. van Benthem, J.: Games in dynamic epistemic logic. *Bulletin of Economic Research* 53(4), 219–248 (2001)
28. van Benthem, J.: What one may come to know. *Analysis* 64(2), 95–105 (2004)
29. van Benthem, J.: An Essay on Sabotage and Obstruction. In: Hutter, D., Stephan, W. (eds.) *Mechanizing Mathematical Reasoning*. LNCS (LNAI), vol. 2605, pp. 268–276. Springer, Heidelberg (2005)
30. van der Hoek, W., Wooldridge, M.J.: Tractable multiagent planning for epistemic goals. In: *Proceedings of the First AAMAS*, pp. 1167–1174. ACM (2002)
31. van Ditmarsch, H., French, T.: Simulation and Information: Quantifying over Epistemic Events. In: Meyer, J.-J.C., Broersen, J.M. (eds.) KRAMAS 2008. LNCS (LNAI), vol. 5605, pp. 51–65. Springer, Heidelberg (2009)
32. van Ditmarsch, H., French, T., Pinchinat, S.: Future event logic - axioms and complexity. In: *Advances in Modal Logic*, vol. 8, pp. 77–99. College Publications (2010)
33. van Ditmarsch, H., van der Hoek, W., Iliev, P.: Everything is knowable – how to get to know whether a proposition is true. *Theoria* 78(2), 93–114 (2012)
34. Visser, A.: *Bisimulations, model descriptions and propositional quantifiers*. Logic Group Preprint Series 161, Department of Philosophy, Utrecht University (1996)
35. Wen, X., Liu, H., Huang, F.: An Alternative Logic for Knowability. In: van Ditmarsch, H., Lang, J., Ju, S. (eds.) LORI 2011. LNCS (LNAI), vol. 6953, pp. 342–355. Springer, Heidelberg (2011)

Linearizing Bad Sequences: Upper Bounds for the Product and Majoring Well Quasi-orders

Sergio Abriola¹, Santiago Figueira^{2,3,*}, and Gabriel Senno²

¹ Dto. Matemática, FCEN, Universidad de Buenos Aires, Argentina

² Dto. Computación, FCEN, Universidad de Buenos Aires, Argentina

³ CONICET, Argentina

Abstract. Well quasi-orders (wqo's) are an important mathematical tool for proving termination of many algorithms. Under some assumptions upper bounds for the computational complexity of such algorithms can be extracted by analyzing the length of controlled bad sequences.

We develop a new, self-contained study of the length of bad sequences over the product ordering of \mathbb{N}^n , which leads to known results but with a much simpler argument.

We also give a new tight upper bound for the length of the longest controlled descending sequence of multisets of \mathbb{N}^n , and use it to give an upper bound for the length of controlled bad sequences in the majoring ordering of sets of tuples. We apply this upper bound to obtain complexity upper bounds for decision procedures of automata over data trees.

In both cases the idea is to *linearize* bad sequences, i.e. transform them into a descending one over a well-order for which upper bounds can be more easily handled.

1 Introduction

A quasi-order is a binary relation \leq over a given set A that is reflexive and transitive. A sequence $\mathbf{X} = x_0, x_1, x_2, \dots$ of elements of A is called *good* if there are $i < j$ such that $x_i \leq x_j$. A sequence is *bad* if it is not good. A *well quasi-order* (wqo) is a quasi-order where all infinite sequences are good, or, equivalently, all bad sequences are finite.

Wqo's are widely used in termination proofs of algorithms in constraint solving, automated deduction, program analysis, verification and model checking, logic, etc. From the analysis of a termination proof of a given algorithm \mathcal{S} , whose correctness is grounded in the analysis of certain wqo, one may extract a computational complexity upper bound for \mathcal{S} . Roughly, the idea is that any sequence of successive configurations of \mathcal{S} (with a given input) is transformed into a bad sequence in the wqo. Thus, having an upper bound for the length of the bad sequence entails an upper bound for the number of steps that the algorithm needs to terminate.

* Figueira was partially supported by UBA (UBACyT 20020090200116), ANPCyT (PICT-2010-0688) and CONICET (PIP 370).

However, in principle, a bad sequence over a wqo can be arbitrarily large. For instance, the lexicographic ordering \leq_{lex} over \mathbb{N}^n is a well-order, and hence a wqo. Observe that for \mathbb{N}^2 and any N , the sequence

$$\langle 1, 0 \rangle, \langle 0, N \rangle, \langle 0, N - 1 \rangle, \langle 0, N - 2 \rangle, \dots, \langle 0, 1 \rangle, \langle 0, 0 \rangle \tag{1}$$

is \leq_{lex} -bad (which in a total order is equivalent to say that it is *decreasing*) and has length greater than N . Therefore, in general there is no bound to the length of a bad sequence starting with a given element: bad sequences in a wqo are finite but could be *arbitrarily* large.

In practice, in the analysis of termination proofs, one has two additional assumptions of a wqo (A, \leq) . First, one has some effective way of measuring the *size* of each element $x \in A$, notated $|x|_A$ or simply $|x|$.

Definition 1. [13] *A norm function $|\cdot|_A$ over a set A is a mapping $|\cdot|_A : A \rightarrow \mathbb{N}$ that provides every element of A with a positive integer, its norm. The norm function is said to be proper if $\{x \in A \mid |x|_A < n\}$ is finite for every n .*

Second, we may restrict ourselves to bad sequences $\mathbf{x} = x_0, x_1, x_2 \dots$ with a *controlled behavior*, which means that there is an effective way of computing, given i , an upper bound for $|x_i|$.

Definition 2. *Let $g : \mathbb{N} \rightarrow \mathbb{N}$ be a computable increasing function and let (A, \leq) be a wqo with a proper norm. A bad sequence $\mathbf{x} = x_0, x_1, x_2 \dots$ is g, t -controlled if for all i , $|x_i|_A < g(t + i)$. We say that g is the control function for \mathbf{x} .*

As a consequence of König’s Lemma, controlled bad sequences over wqos cannot be *arbitrarily* large: given a control, there exist upper bounds for their lengths. Let us go back to the example of the \leq_{lex} -decreasing sequence in (1). If we further impose that the sequence is $g, 0$ -controlled, where $g(0) = 2$ and we fix $|x|_{\mathbb{N}^2}$ to be the infinity norm of x then the reader may verify that the *longest* $g, 0$ -controlled decreasing sequence is

$$\langle 1, 1 \rangle, \langle 1, 0 \rangle, \langle 0, g(2) - 1 \rangle, \langle 0, g(2) - 2 \rangle, \dots, \langle 0, 1 \rangle, \langle 0, 0 \rangle. \tag{2}$$

In this paper we give upper bounds for the length of g, t -controlled bad sequences, when t is a parameter. That is, given a well (quasi) order under study (we address lexicographic, product, multiset and majoring) (A, \leq) , we define $L_g^A(t)$ as the length of the longest g, t -controlled bad sequence in (A, \leq) , and we study upper bounds for L_g^A , which are classified in the Fast Growing Hierarchy $(\mathfrak{F}_\alpha)_{\alpha < \epsilon_0}$ of Löb and Wainer [10].

For a more detailed introduction to some topics of this paper, see [1].

Linearizing

Our technique to obtain an upper bound for L_g^A is to *linearize* the wqo (A, \leq_A) with a proper norm $|\cdot|_A$ into a suitable well linear order (B, \leq_B) with a proper norm $|\cdot|_B$. This means to find a function $h : A^+ \rightarrow B$ such that for every

$\mathbf{a} \in A^+$ and $a \in A$, if $\mathbf{a} \frown a$ is a bad sequence in (A, \leq_A) then $h(\mathbf{a}) >_B h(\mathbf{a} \frown a)$. So if $\mathbf{a} = a_0, \dots, a_k$ is bad in (A, \leq_A) then

$$\mathbf{b} = h(a_0), h(a_0, a_1), h(a_0, a_1, a_2), \dots, h(\mathbf{a})$$

is descending in (B, \leq_B) . Furthermore, for any control function g we seek a control function \tilde{g} such that if \mathbf{a} is g, t -controlled then $|h(\mathbf{a})|_B < \tilde{g}(|\mathbf{a}| + t - 1)$ —here $|\mathbf{a}|$ denotes the length of \mathbf{a} . Hence if \mathbf{a} is g, t -controlled then \mathbf{b} is \tilde{g}, t -controlled and therefore from a g, t -controlled bad sequence in (A, \leq_A) one can get a \tilde{g}, t -descending sequence in (B, \leq_B) of the same length. Hence $L_g^A \leq L_{\tilde{g}}^B$, and the task is now to find an upper bound for L_g^B . In practice, these upper bounds are easier to devise for well-orders than for wqo's.

Our Contributions

Product and lexicographic ordering. For some dimension n , let $(\mathbb{N}^n, \leq_{\text{pr}})$ be the set of n -tuples of \mathbb{N} ordered with the natural product ordering. Dickson's Lemma is the statement that $(\mathbb{N}^n, \leq_{\text{pr}})$ is a wqo. We denote $L_{n,g}^{\text{pr}}(t)$ the length of the longest g, t -controlled bad sequence over $(\mathbb{N}^n, \leq_{\text{pr}})$. Here we take $|x|_{\mathbb{N}^n}$ to be $|x|_{\infty}$.

McAloon [11] shows an upper bound for $L_{n,g}^{\text{pr}}$ when g is linear, and places it at the level \mathfrak{F}_{n+1} of the Fast Growing Hierarchy. Later Clote [2] simplifies McAloon's argument and finds an upper bound in \mathfrak{F}_{n+6} . Neither of these proofs are self contained and both are quite complex. In [5] D. and S. Figueira, Schmitz and Schnoebelen show an improved upper bound of \mathfrak{F}_n with a simpler proof, relying in a mathematical more general setting of disjoint unions of powers of \mathbb{N} . In fact, the main result is both more general and more precise than those of McAloon and Clote: if $g \in \mathfrak{F}_\gamma$ then $L_{n,g}^{\text{pr}}$ is bounded by a function in $\mathfrak{F}_{\gamma+n-1}$. Although this proof is markedly simpler than those of [11] and [2], there are still some technical lemmas regarding this richer setting.

In Thm. 4 we give an even shorter, elementary and self-contained proof of the result of [5] which only uses a linearization of $(\mathbb{N}^n, \leq_{\text{pr}})$ into $(\mathbb{N}^n, \leq_{\text{lex}})$. As a side result, in Prop. 3 we obtain a tight upper bound for the length of the longest decreasing sequence in $(\mathbb{N}^n, \leq_{\text{lex}})$.

Majoring and multiset ordering. Informally, if A and B are finite subsets of \mathbb{N}^n then $A \leq_{\text{maj}} B$ iff every element of A is majorized (with respect to \leq_{pr}) in B . It is well-known that \leq_{maj} over subsets of \mathbb{N}^n is a wqo, and this fact is used in a number of decidability results.

In Cor. 19 we show an upper bound for $L_{n,g}^{\text{maj}}(t)$, the length of the longest g, t -controlled \leq_{maj} -bad sequence of finite subsets of \mathbb{N}^n . To obtain this upper bound, we linearize the wqo into the multiset ordering over $(\mathbb{N}^n, \leq_{\text{lex}})$, which is a well-order. In Thm. 12 and Thm. 14 we show a tight upper bound for the longest decreasing sequence of multisets.

We also give some applications on how our upper bound for $L_{n,g}^{\text{maj}}(t)$ can be used in some decision procedures of some types of automata over data trees.

Outline

In §2 we give the formal definitions of all the involved orders and the definition of the Fast Growing Hierarchy. In §3, §4, §5, §6 we study the lexicographic, product, multiset and majoring ordering, respectively. In §7 we mention some applications of our upper bounds in concrete decision procedures. We close with some conclusions and future work in §8.

2 Basic Definitions

If A is a set then $|A|$ denotes the cardinality of A . If $x \in A^n$ then the i -th coordinate of x is denoted $x[i]$, so $x = \langle x[1], \dots, x[n] \rangle$. Sequences are always in boldface and if \mathbf{x} is a finite sequence then $|\mathbf{x}|$ denotes its length. The concatenation of the sequence \mathbf{x} and the element x at the rightmost place is denoted $\mathbf{x} \hat{\ } x$. We fix $g : \mathbb{N} \rightarrow \mathbb{N}$ to be a computable and increasing function.

Given a set X provided with a total order \leq , (X, \leq) is called a *well-order* if every non-empty subset of X has a minimum.

We work with the following wqo's:

Lexicographic ordering. If $x, y \in \mathbb{N}^n$ then it is the well-order defined as

$$x <_{\text{lex}} y \stackrel{\text{def}}{\iff} x[1] < y[1] \vee (x[1] = y[1] \wedge \langle x[2], \dots, x[n] \rangle <_{\text{lex}} \langle y[2], \dots, y[n] \rangle).$$

Product ordering. If $x, y \in \mathbb{N}^n$ then it is the wqo defined as

$$x \leq_{\text{pr}} y \stackrel{\text{def}}{\iff} (\forall i \in \{1, \dots, n\}) x[i] \leq y[i].$$

Multiset ordering. A multiset M over a set X is a function $X \rightarrow \mathbb{N}$. Intuitively a multiset is a generalization of a set, where elements may be repeated. For $x \in X$, $M(x)$ is called the *multiplicity* of x . A multiset is finite if the set of elements with positive multiplicity is finite. We notate $x \in M$ for $M(x) > 0$. Let $\mathcal{M}_{<\infty}(X)$ denote the class of finite multisets over X .

Let (X, \leq) be a poset and let $M, N \in \mathcal{M}_{<\infty}(X)$. We define

$$N <_{\text{ms}}^{(\leq)} M \stackrel{\text{def}}{\iff} M \neq N \wedge (\forall x \in X)[N(x) > M(x) \Rightarrow (\exists y \in X)[y > x \wedge M(y) > N(y)]].$$

Intuitively, this says that N can be obtained from M by replacing some elements by finitely many (possibly zero) smaller (with respect to \leq) elements. If (X, \leq) is a well-order then $(\mathcal{M}_{<\infty}(X), \leq_{\text{ms}}^{(\le)})$ is also a well-order. See §3 for more details.

We will study $(\mathcal{M}_{<\infty}(\mathbb{N}^n), \leq_{\text{ms}}^{(\leq_{\text{lex}})})$, the multiset ordering of finite multisets of tuples with the underlying lexicographic ordering. In this context, we write \leq_{ms} for $\leq_{\text{ms}}^{(\leq_{\text{lex}})}$. Observe that it is a well-order because $(\mathbb{N}^n, \leq_{\text{lex}})$ is so.

Majoring ordering. Let $\mathcal{P}_{<\infty}(X)$ denote the finite and non-empty parts of X . For a wqo (X, \leq) and $A, B \in \mathcal{P}_{<\infty}(X)$, the *majoring ordering* is defined as

$$A \leq_{\text{maj}}^{(\leq)} B \stackrel{\text{def}}{\iff} (\forall x \in A)(\exists y \in B) x \leq y.$$

We will study $(\mathcal{P}_{<\infty}(\mathbb{N}^n), \leq_{\text{maj}}^{(\leq_{\text{pr}})})$, the majoring ordering of finite sets of tuples with the underlying product ordering. In this context, we write \leq_{maj} for $\leq_{\text{maj}}^{(\leq_{\text{pr}})}$. Observe that it is a wqo because $(\mathbb{N}^n, \leq_{\text{pr}})$ is so (see for instance [4, Prop. 2.15]).

The Fast Growing Hierarchy $(F_\alpha)_{\alpha < \epsilon_0}$. Let ϵ_0 be the least infinite ordinal α such that $\omega^\alpha = \alpha$. The Fast Growing Hierarchy is defined as

$$F_0(x) \stackrel{\text{def}}{=} x + 1 \quad F_{\alpha+1}(x) \stackrel{\text{def}}{=} F_\alpha^{x+1}(x) \quad F_\lambda \stackrel{\text{def}}{=} F_{\lambda_x}(x),$$

where in general g^k denotes the k -th iteration of g (i.e. $g^1 = g$ and $g^{k+1} = g \circ g^k$), $\alpha < \epsilon_0$ is an ordinal, $\lambda < \epsilon_0$ is a limit ordinal and $(\lambda_x)_{x < \omega}$ is an increasing sequence of ordinals with limit λ (a *fundamental sequence*), which we fix to be:

$$(\gamma + \omega^{\beta+1})_x \stackrel{\text{def}}{=} \gamma + \omega^\beta \cdot (x + 1) \quad (\gamma + \omega^\lambda)_x \stackrel{\text{def}}{=} \gamma + \omega^{\lambda_x}.$$

The class \mathfrak{F}_α of the Fast Growing Hierarchy is the closure under substitution and limited recursion of the constant, sum, projections, and the functions F_α . $\mathfrak{F}_0 = \mathfrak{F}_1$ contains all linear functions, \mathfrak{F}_2 contains all the elementary functions, \mathfrak{F}_3 contains all the tetration functions. $\bigcup_{n < \omega} \mathfrak{F}_n$ is the class of all primitive recursive functions and in general $\bigcup_{\alpha < \omega^k} \mathfrak{F}_\alpha$ is the class of k -recursive functions [12]. There are a number of important *monotonicity* results regarding the Fast Growing Hierarchy: for ordinals $\alpha < \beta < \epsilon_0$, the function F_α is strictly increasing, $F_{\alpha+1} \geq F_\alpha$, F_α is eventually majorized by F_β , and then $\mathfrak{F}_\alpha \subsetneq \mathfrak{F}_\beta$ (except $\alpha = 0$ and $\beta = 1$), etc. For more results on the Fast Growing Hierarchy, cf. [10].

3 The Lexicographic Ordering

We denote by $L_{n,g}^{\text{lex}}(t)$ the length of the longest g , t -controlled decreasing sequence in $(\mathbb{N}^n, \leq_{\text{lex}})$. In [5, Section VI], it is shown that

$$L_{1,g}^{\text{lex}}(t) = g(t), \quad L_{n+1,g}^{\text{lex}}(t) = \sum_{j=1}^{g(t)} L_{n,g}^{\text{lex}}(o_{n,g}^{j-1}(t)), \quad o_{n,g}(t) \stackrel{\text{def}}{=} t + L_{n,g}^{\text{lex}}(t). \quad (3)$$

Proposition 3. *For any ordinal $\gamma \geq 1$, if $g \in \mathfrak{F}_\gamma$ then $L_{n,g}^{\text{lex}}$ has an upper bound in $\mathfrak{F}_{\gamma+n-1}$.*

Proof. We proceed by induction on n . If $n = 1$ then $L_{1,g}^{\text{lex}}(t) = g(t)$, and by hypothesis $g \in \mathfrak{F}_\gamma$. Now suppose $L_{n,g}^{\text{lex}} \leq h \in \mathfrak{F}_{\gamma+n-1}$. We have $L_{n+1,g}^{\text{lex}}(t) \leq g(t) \cdot L_{n,g}^{\text{lex}}(o_{n,g}^{g(t)-1}(t)) \leq g(t) \cdot o_{n,g}^{g(t)}(t)$ where the first inequality follows from (3), since $o_{n,g}$ is growing, and the second one because $L_{n,g}^{\text{lex}} \leq o_{n,g}$.

Since $L_{n,g}^{\text{lex}} \leq h \in \mathfrak{F}_{\gamma+n-1}$ then $o_{n,g}(t) \leq h(t) + t$ and so $o_{n,g} \in \mathfrak{F}_{\gamma+n-1}$. By [10, Thm. 2.10], there is p such that $F_{\gamma+n-1}^p$ majorizes $o_{n,g}$. Therefore

$$\begin{aligned} L_{n+1,g}^{\text{lex}}(t) &< g(t) \cdot F_{\gamma+n-1}^{p \cdot g(t)}(t) \\ &\leq g(t) \cdot F_{\gamma+n-1}^{p \cdot g(t)+1}(p \cdot g(t)) && \text{(by mononicity of } F_{\gamma+n-1}) \\ &= g(t) \cdot F_{\gamma+n}(p \cdot g(t)), \end{aligned}$$

which lies in $\mathfrak{F}_{\gamma+n}$, since it is the composition and product of functions in $\mathfrak{F}_{\gamma+n}$ (and since $\gamma + n \geq 2$, $\mathfrak{F}_{\gamma+n}$ is closed by products). \square

In [5, Prop. VI.3] it is shown that if $g = F_\gamma$ then $L_{n,g}^{\text{lex}} \geq F_{\gamma+n-1}$. Hence our upper bound is tight.

4 The Product Ordering

In this section we linearize the wqo $(\mathbb{N}^n, \leq_{\text{pr}})$ into the well-order $(\mathbb{N}^n, \leq_{\text{lex}})$ and derive an upper bound for $L_{n,g}^{\text{pr}}(t)$, the length of the longest g, t -controlled bad sequence over $(\mathbb{N}^n, \leq_{\text{pr}})$.

The next result follows the idea of Harwood, Moller and Setzer [7] adapted to *controlled* bad sequences. For the sake of completeness we include the full proof.

First, let us mention the intuition behind the proof. For $x \in \mathbb{N}^n$, define $\uparrow x \stackrel{\text{def}}{=} \{z \in \mathbb{N}^n \mid x \leq_{\text{pr}} z\}$. Let $n = 2$, and suppose

$$\mathbf{x} = \langle x_0, y_0 \rangle, \langle x_1, y_1 \rangle, \langle x_2, y_2 \rangle, \dots, \langle x_k, y_k \rangle$$

is a bad sequence in $(\mathbb{N}^2, \leq_{\text{pr}})$. Let $a(\mathbf{x}) = \min_{0 \leq i < |\mathbf{x}|} x_i$, $b(\mathbf{x}) = \min_{0 \leq i < |\mathbf{x}|} y_i$ and $C(\mathbf{x}) = \uparrow(a(\mathbf{x}), b(\mathbf{x})) \setminus \bigcup_{0 \leq i < |\mathbf{x}|} \uparrow \langle x_i, y_i \rangle$. It is easy to see that $C(\mathbf{x})$ is finite.

Here is how we can linearize $(\mathbb{N}^2, \leq_{\text{pr}})$ into $(\mathbb{N}^2, \leq_{\text{lex}})$: Define $h(\mathbf{x}) \stackrel{\text{def}}{=} \langle a(\mathbf{x}) + b(\mathbf{x}), |C(\mathbf{x})| \rangle \in \mathbb{N}^2$ and suppose that $\mathbf{x} \frown \langle x, y \rangle$ is bad. If $x < a(\mathbf{x}) \vee y < b(\mathbf{x})$ then $h(\mathbf{x} \frown \langle x, y \rangle)[1] < h(\mathbf{x})[1]$; in case $x \geq a(\mathbf{x}) \wedge y \geq b(\mathbf{x})$ then $C(\mathbf{x} \frown \langle x, y \rangle) \subseteq C(\mathbf{x})$. In this last case, since $\langle x, y \rangle \in C(\mathbf{x}) \setminus C(\mathbf{x} \frown \langle x, y \rangle)$, we have $|C(\mathbf{x} \frown \langle x, y \rangle)| < |C(\mathbf{x})|$. Therefore $h(\mathbf{x} \frown \langle x, y \rangle) <_{\text{lex}} h(\mathbf{x})$. Furthermore, if \mathbf{x} is g, t -controlled then $C(\mathbf{x})$ has at most $g(t + |\mathbf{x}| - 1)^2$ elements, and $a(\mathbf{x}) + b(\mathbf{x}) < 2g(t + |\mathbf{x}| - 1)$. Hence if \mathbf{x} is g, t -controlled, then the sequence

$$\mathbf{y} = h(\langle x_0, y_0 \rangle), h(\langle x_0, y_0 \rangle, \langle x_1, y_1 \rangle), \dots, h(\bar{\mathbf{x}}),$$

is $<_{\text{lex}}$ -descending and \tilde{g}, t -controlled, where $\tilde{g}(x) = 2g(x)^2$.

The argument for any $n > 2$ cannot be generalized straightforwardly, obtaining a linearization into $(\mathbb{N}^n, \leq_{\text{lex}})$. For instance, for $n = 3$ and $\mathbf{x} = \langle 0, 0, 1 \rangle, \langle 0, 1, 0 \rangle$, we would have $C(\mathbf{x}) = \uparrow \langle 0, 0, 0 \rangle \setminus (\uparrow \langle 0, 0, 1 \rangle \cup \uparrow \langle 0, 1, 0 \rangle)$ and this set is infinite ($(N, 0, 0) \in C(\mathbf{x})$ for any N). However, by an inductive argument $(\mathbb{N}^n, \leq_{\text{pr}})$ can be linearized into $(\mathbb{N}^n, \leq_{\text{lex}})$.

Theorem 4. *There is a function $h_n : (\mathbb{N}^n)^+ \rightarrow \mathbb{N}^n$ such that if $\mathbf{x} \frown x$ is bad in $(\mathbb{N}^n, \leq_{\text{pr}})$ and \mathbf{x} is nonempty, then $h_n(\mathbf{x} \frown x) <_{\text{lex}} h_n(\mathbf{x})$. Furthermore if \mathbf{x} is g, t -controlled then $|h_n(\mathbf{x})|_\infty < \tilde{g}(|\mathbf{x}| - 1 + t)$, for $\tilde{g}(x) = n! g(nx)^n$.*

Proof. We define the functions h_n by induction in n . If $\mathbf{x} = x_0, x_1, x_2, \dots, x_k$ is a bad sequence in \mathbb{N} then define $h_1(x_0, x_1, x_2, \dots, x_k) \stackrel{\text{def}}{=} x_k$. Since in \mathbb{N} the product order and the lexicographic order coincide, we have $h_1(\mathbf{x} \frown x) <_{\text{lex}} h_1(\mathbf{x})$.

For the inductive construction of h_n , let $n > 1$ and assume the statement of the theorem for dimension $n - 1$. For $1 \leq i \leq n$ and $x \in \mathbb{N}^n$ we define

$$\text{DEL}_i(x) \stackrel{\text{def}}{=} \langle x[1], \dots, x[i - 1], x[i + 1], \dots, x[n] \rangle,$$

i.e. $\text{DEL}_i(x)$ deletes the i -th component of the n -tuple x . Given a finite and nonempty bad sequence $\mathbf{x} = x_0, x_1, \dots, x_k$ of n -tuples, we define the set

$$\text{BAD}_i(\mathbf{x}) \stackrel{\text{def}}{=} \{ \text{DEL}_i(x_{j_0}), \dots, \text{DEL}_i(x_{j_p}) \mid p \geq 0, 0 \leq j_0 < \dots < j_p \leq k, \text{ and } \text{DEL}_i(x_{j_0}), \dots, \text{DEL}_i(x_{j_p}) \text{ is bad} \},$$

i.e. $\text{BAD}_i(\mathbf{x})$ consists of the bad subsequences of $(n-1)$ -tuples of \mathbf{x} in which the i -th components of the n -tuples have been deleted. Finally we define

$$\text{MIN}_i(\mathbf{x}) \stackrel{\text{def}}{=} \min_{<_{\text{lex}}} \{ h_{n-1}(\mathbf{y}) \mid \mathbf{y} \in \text{BAD}_i(\mathbf{x}) \} \quad \text{and}$$

$$\text{EXT}_n(\mathbf{x}) \stackrel{\text{def}}{=} \{ x \in \mathbb{N}^n \mid (\forall i \in \{1, \dots, n\}) \text{MIN}_i(\mathbf{x}) = \text{MIN}_i(\mathbf{x} \hat{\ } x), \text{ and } (\forall j \in \{0, \dots, k\}) x_j \not\leq_{\text{pr}} x_j \},$$

which consists of the n -tuples with which the sequence \mathbf{x} can be extended without altering the MIN_i values and yet while maintaining badness.

Fact 1. $|\text{EXT}_n(\mathbf{x})| < \infty$, and if \mathbf{x} is g, t -controlled, then $|\text{EXT}_n(\mathbf{x})| < g(k+t)^n$.

Proof. Let $\mathbf{z} = \text{DEL}_i(x_{j_0}), \dots, \text{DEL}_i(x_{j_p}) \in (\mathbb{N}^{n-1})^+$ be a bad sequence, suppose $\text{MIN}_i(\mathbf{x}) = h_{n-1}(\mathbf{z})$, and suppose that $s \in \text{EXT}_n(\mathbf{x})$. If the sequence $\mathbf{z} \hat{\ } \text{DEL}_i(s)$ were bad, then by the ind. hyp. we would get that $\text{MIN}_i(\mathbf{x} \hat{\ } s) \leq_{\text{lex}} h_{n-1}(\mathbf{z} \hat{\ } \text{DEL}_i(s)) <_{\text{lex}} h_{n-1}(\mathbf{z}) = \text{MIN}_i(\mathbf{x})$, contradicting $s \in \text{EXT}_n(\mathbf{x})$. Therefore, since \mathbf{z} is bad but $\mathbf{z} \hat{\ } \text{DEL}_i(s)$ is not, we have $\text{DEL}_i(x_{j_m}) \leq_{\text{pr}} \text{DEL}_i(s)$ for some m . But since $s \in \text{EXT}_n(\mathbf{x})$ we have that $x_{j_m} \not\leq_{\text{pr}} s$, and therefore $s[i] < x_{j_m}[i]$. Now, since this goes for all i , we conclude that $|\text{EXT}_n(\mathbf{x})|$ is finite.

Now if \mathbf{x} is g, t -controlled, then $x_j[i] < g(k+t)$ for all j , because g is increasing. By the above argument $|\text{EXT}_n(\mathbf{x})| \leq g(k+t)^n$, but since \mathbf{x} was nonempty and $x_0 \notin \text{EXT}_n(\mathbf{x})$, we conclude $|\text{EXT}_n(\mathbf{x})| < g(k+t)^n$. \square

We finally define

$$h_n(\mathbf{x}) \stackrel{\text{def}}{=} \left\langle \sum_{i=1}^n \text{MIN}_i(\mathbf{x}), |\text{EXT}_n(\mathbf{x})| \right\rangle \in \mathbb{N}^n,$$

where the sum is taken componentwise and thus results in a tuple in \mathbb{N}^{n-1} . We conclude the proof with the following two facts:

Fact 2. If $\mathbf{x} \hat{\ } x$ is bad then $h_n(\mathbf{x} \hat{\ } x) <_{\text{lex}} h_n(\mathbf{x})$.

Proof. Suppose that $\mathbf{y} = \mathbf{x} \hat{\ } x$ is bad. Since for any $i \in \{1, \dots, n\}$, $\text{BAD}_i(\mathbf{x}) \subseteq \text{BAD}_i(\mathbf{y})$, then $\text{MIN}_i(\mathbf{y}) \leq_{\text{lex}} \text{MIN}_i(\mathbf{x})$; and if $\text{MIN}_i(\mathbf{y}) = \text{MIN}_i(\mathbf{x})$ for all i then $\text{EXT}_n(\mathbf{y}) \subsetneq \text{EXT}_n(\mathbf{x})$, since $\text{EXT}_n(\mathbf{y}) \subseteq \text{EXT}_n(\mathbf{x})$ but $x \in \text{EXT}_n(\mathbf{x}) \setminus \text{EXT}_n(\mathbf{y})$. Thus $|\text{EXT}_n(\mathbf{y})| < |\text{EXT}_n(\mathbf{x})|$. \square

Fact 3. If \mathbf{x} is g, t controlled then $|h_n(\mathbf{x})|_{\infty} < \tilde{g}(|\mathbf{x}| - 1 + t)$, where $\tilde{g}(x) = n! g(nx)^n$.

Proof. By induction in $n \geq 1$. If $n = 1$ then If $\mathbf{x} = x_0, \dots, x_k$ is g, t -controlled, then $h_1 \mathbf{x} = x_k < g(t+k) = g(t+|\mathbf{x}|-1) = \tilde{g}(t+|\mathbf{x}|-1)$.

Since any $\mathbf{y} \in \text{BAD}_i(\mathbf{x})$ is a $g, (t+k)$ -controlled bad sequence of \mathbb{N}^{n-1} , by inductive hypothesis we get

$$\begin{aligned} |h_{n-1}(\mathbf{y})|_\infty &< (n-1)! g((n-1)(|\mathbf{y}|-1) + t+k)^{n-1} \\ &\leq (n-1)! g((n-1)k + t+k)^{n-1} \\ &= (n-1)! g(nk+t)^{n-1}. \end{aligned}$$

In particular, for \mathbf{y} such that $\text{MIN}_i(\mathbf{x}) = h_{n-1}(\mathbf{y})$, we conclude $|\text{MIN}_i(\mathbf{x})|_\infty < (n-1)! g(nk+t)^{n-1}$, and so the first $n-1$ coordinates of $h_n(\mathbf{x})$ are strictly bounded by $n! g(nk+t)^{n-1}$ (the factor n comes from the n additions). By Fact [1](#), the last coordinate of $h_n(\mathbf{x})$ is strictly bounded by $g(k+t)^n$. Therefore, $|h_n(\mathbf{x})|_\infty < \max\{n! g(nk+t)^{n-1}, g(k+t)^n\} \leq n! g(nk+t)^n \leq \tilde{g}(|\mathbf{x}|-1+t)$. \square

Let $L_{n,g}^{\text{pr}}(t)$ denote the length of the longest g, t -controlled bad sequence in $(\mathbb{N}^n, \leq_{\text{pr}})$, and let $L_{n,g}^{\text{lex}}(t)$ denote the length of the longest g, t -controlled decreasing sequence in $(\mathbb{N}^n, \leq_{\text{lex}})$. We arrive to the same result as in [5](#):

Corollary 5. $L_{n,g}^{\text{pr}} \leq L_{n,\tilde{g}}^{\text{lex}}$, for \tilde{g} as in Thm. [4](#). Hence if $g \in \mathfrak{F}_\gamma$, and $\gamma \geq 2$ is an ordinal, then $L_{n,g}^{\text{pr}}$ has an upper bound in $\mathfrak{F}_{\gamma+n-1}$.

Proof. The function \tilde{g} is defined through finite substitution from g and product. Since \mathfrak{F}_2 and higher levels are closed under finite products, we have $\tilde{g} \in \mathfrak{F}_\gamma$. By Prop. [3](#), there is a function $h \in \mathfrak{F}_{\gamma+n-1}$ such that $h \geq L_{n,\tilde{g}}^{\text{lex}}$. \square

5 The Multiset Ordering

We need a notion of g, t -controlled sequence of (multi)sets. By Def. [2](#) it suffices to give a proper norm:

Definition 6 (A proper norm of sets and multisets of tuples). Given $X \in \mathcal{M}_{<\infty}(\mathbb{N}^n)$, we define $|X|$, the norm of X , as the maximum between $\max_{x \in \mathbb{N}^n} X(x)$ and $\max\{|x|_\infty \mid x \in \mathbb{N}^n \wedge X(x) > 0\}$. For $X \in \mathcal{P}_{<\infty}(\mathbb{N}^n)$, $|X|$ is defined analogously, as any set is a multiset.

We denote by $L_{g,n}^{\text{ms}}(t)$ the length of the longest g, t -controlled decreasing sequence in $(\mathcal{M}_{<\infty}(\mathbb{N}^n), \leq_{\text{ms}}^{\leq_{\text{lex}}})$, i.e. a sequence of finite multisets of \mathbb{N}^n , with the underlying lexicographic ordering. In this section we give a tight upper bound for $L_{g,n}^{\text{ms}}(t)$ in terms of the Fast Growing Hierarchy.

5.1 Maximizing Strategy

To study the longest g, t -controlled \leq_{ms} -descending sequence of multisets we define the maximizing strategy which, given a nonempty g, t -controlled multiset M , determines the greatest $g, (t+1)$ -controlled multiset N which is smaller than M .

The strategy says that to obtain N one should take out one of the minimum elements of M , say m , (i.e. decrement in one the multiplicity of m) and add as many elements smaller than m as the control function permits.

For the rest of this subsection, assume (X, \leq) is a well-order. We write $<_{\text{ms}}$ instead of $<_{\text{ms}}^{(\leq)}$. Let $M \in \mathcal{M}_{<\infty}(X)$ which is g, t -controlled and a proper norm $|\cdot|_X = |\cdot|$ for X . We define the g, t -predecessor of M as follows: For $x \in X$,

$$\text{PRED}_t^g(M)(x) \stackrel{\text{def}}{=} \begin{cases} g(t+1) - 1 & x < \min M \wedge |x| < g(t+1); \\ M(x) - 1 & x = \min M; \\ M(x) & \text{otherwise.} \end{cases}$$

where $\min M \stackrel{\text{def}}{=} \min\{x \mid M(x) > 0\}$.

Lemma 7. *Let M be a nonempty finite multiset over a totally ordered set P , which is g, t -controlled and let $N = \text{PRED}_t^g(M)$. Then (1) N is $g, (t+1)$ -controlled; (2) $N <_{\text{ms}} M$; and (3) if N' is $g, (t+1)$ -controlled and $N' <_{\text{ms}} M$ then $N' \leq_{\text{ms}} N$.*

Proof. (1) is clear from the definition of N and the fact that g is monotone increasing. For (2), it is obvious that $M \neq N$. By definition, if $N(x) > M(x)$ then $x < m = \min M$ and $M(m) > N(m)$.

For (3), assume $N' < M$ is $g, (t+1)$ -controlled. We show that if $N'(x) > N(x)$ then there is $z > x$ such that $N(z) > N'(z)$. Suppose $N'(x) > N(x)$. First, if $x < \min M$ then $N(x) = g(t+1) - 1 \geq N'(x)$, contradicting $N'(x) > N(x)$. Second, suppose $x > \min M$. Then $N(x) = M(x)$ and therefore $N'(x) > M(x)$. Since $N' <_{\text{ms}} M$ there is $z > x$ such that $N(z) = M(z) > N'(z)$. Third, suppose $x = \min M$. Then $N(x) = M(x) - 1$, and so $N'(x) \geq M(x)$. If $N'(x) > M(x)$ then, since $M <_{\text{ms}} N'$, there is $z > x$ with $M(z) > N'(z)$. For such z , by definition of N , we have $N(z) = M(z) > N'(z)$. If $N'(x) = M(x)$ then, since $N' \neq M$, there is y such that $N'(y) \neq M(y)$. Any such y must be different from x . Suppose that all such y 's were smaller than $x = \min M$. In this case $M \leq_{\text{ms}} N'$ and this contradicts the hypothesis. Hence there is $y > x$ such that $N'(y) \neq M(y)$. If $N'(y) > M(y)$, there is $z > y > x$ such that $N'(z) < M(z) = N(z)$. If $N'(y) < M(y)$, since $M(y) = N(y)$, we conclude $N'(y) < N(y)$. \square

We represent a finite multiset M such that $\{x \mid M(x) > 0\} = \{x_1, \dots, x_n\}$ as $M \stackrel{\text{def}}{=} M(x_1) \cdot x_1 + \dots + M(x_n) \cdot x_n$.

For a finite multiset M , let $L_{g,M}(t)$ denote the length minus one of the longest g, t -controlled and $<_{\text{ms}}$ -decreasing sequence of multisets starting with the multiset M . For $x \in X$, let $o_{g,x}(t) = t + L_{g,1 \cdot \{x\}}(t)$.

Lemma 8. *If $k \geq 1$ then $L_{g,k \cdot \{x\}}(t) = \sum_{i=0}^{k-1} L_{g,1 \cdot \{x\}}(o_{g,x}^i(t))$.*

Proof (Sketch). We write L_k for $L_{g,k \cdot \{x\}}$ and o for $o_{g,x}$. First show by induction in i that $o^i(t) = t + \sum_{j=0}^{i-1} L_1(o^j(t))$. Then show the statement of the Lemma by

induction in $k \geq 1$. Observe that the longest g, t -controlled decreasing sequence of multisets beginning with $M_1 = (k + 1) \cdot \{x\}$ is

$$M_1 >_{\text{ms}} M_2 >_{\text{ms}} \dots >_{\text{ms}} M_{l_1} >_{\text{ms}} N_2 >_{\text{ms}} N_3 >_{\text{ms}} \dots >_{\text{ms}} N_{l_2},$$

of length $l_1 + l_2 - 1$ and where $l_1 = L_k(t) + 1$, $M_{l_1} = 1 \cdot \{x\}$, $l_2 = L_1(t + L_k(t)) + 1$ and $N_{l_2} = \emptyset$. Use straightforwardly the inductive hypothesis. \square

Corollary 9. For $k \geq 1$, $L_{g,k \cdot \{x\}} \geq L_{g,1 \cdot \{x\}}^k$.

Corollary 10. For $k \geq 1$, $L_{g,k \cdot \{x\}}(t) \leq k \cdot L_{g,1 \cdot \{x\}}(o_{g,x}^{k-1}(t))$.

In the sequel we fix (X, \leq) to be $(\mathbb{N}^n, \leq_{\text{lex}})$. If $M \in \mathcal{M}_{< \infty}(\mathbb{N}^n)$ then $P_{g,n}(M, t)$ denotes the length minus one of the longest g, t -controlled $<_{\text{ms}}$ -decreasing sequence of multisets starting with M . If M consists of one copy of (x_1, \dots, x_n) , we simply write $P_{g,n}(x_1, \dots, x_n, t)$ instead of $P_{g,n}(1 \cdot \{x_1, \dots, x_n\}, t)$. Observe that, having fixed (X, \leq) , we have $L_{g,M}(t) = P_{g,n}(M, t)$.

5.2 Lower Bound

Define $G_{g,n} : \mathbb{N}^{n+1} \setminus \{(0, \dots, 0)\} \rightarrow \mathbb{N}$ by multiple recursion as:

$$G_{g,n}(0, \dots, 0, 1, t) \stackrel{\text{def}}{=} g(t + 1) \tag{4}$$

$$G_{g,n}(\bar{x}, x_n + 1, t) \stackrel{\text{def}}{=} G_{g,n}^{g(t+1)-1}(\bar{x}, x_n, t), \text{ for } \bar{x} = x_1, \dots, x_{n-1} \tag{5}$$

$$G_{g,n}(\bar{x}, x_j + 1, \bar{0}, t) \stackrel{\text{def}}{=} G_{g,n}(\bar{x}, x_j, g(t + 1) - 1, \bar{0}, t), \text{ for } \bar{x} = x_1, \dots, x_{j-1} \tag{6}$$

Equation (5) applies when $x_i > 0$ for some i , and (6) when $j < n$. $G_{g,n}^k(\bar{a}, b)$ denotes the k -th iteration of $G_{g,n}$ in the last component, i.e. $G_{g,n}^1(\bar{a}, b) = G_{g,n}(\bar{a}, b)$ and $G_{g,n}^{k+1}(\bar{a}, b) = G_{g,n}(\bar{a}, G_{g,n}^k(\bar{a}, b))$.

Lemma 11. If $g(x) \geq x + 1$ then $P_{g,n} \geq G_{g,n}$.

Proof (Sketch). By induction in the lexicographic order of x_1, \dots, x_n . For (4), the longest g, t -controlled $<_{\text{ms}}$ -decreasing sequence starting with $1 \cdot \{(\bar{0}, 1)\}$ is

$$1 \cdot \{(\bar{0}, 1)\} >_{\text{ms}} (g(t + 1) - 1) \cdot \{(\bar{0}, 0)\} >_{\text{ms}} \dots >_{\text{ms}} 0 \cdot \{(\bar{0}, 0)\} = \emptyset,$$

which has length $g(t + 1) + 1$ and then $P_{g,n}(0, \dots, 0, 1, t) = g(t + 1)$. For (5), the longest g, t -controlled $<_{\text{ms}}$ -decreasing sequence of multisets starting with $1 \cdot \{(\bar{x}, x_n + 1)\}$ contains the multiset $M = (g(t + 1) - 1) \cdot \{(\bar{x}, x_n)\}$, so $P_{g,n}(\bar{x}, x_n + 1, t) \geq P_{g,n}(M, t + 1)$. Now apply Cor. 9, monotonicity of $G_{g,n}$ and ind. hyp. For (6), the longest g, t -controlled $<_{\text{ms}}$ -decreasing sequence of multisets starting with $1 \cdot \{(\bar{x}, x_j + 1, \bar{0})\}$ contains $1 \cdot \{x_1, \dots, x_j, g(t + 1) - 1, \bar{0}\}$ as one of its terms, so $P_{g,n}(\bar{x}, x_j + 1, \bar{0}, t) \geq P_{g,n}(\bar{x}, x_j, g(t + 1) - 1, \bar{0}, t)$. Then apply ind. hyp. \square

Theorem 12. If $g \geq F_1$ and $g(x) \geq x + 2$, then $L_{g,n}^{\text{ms}} \geq F_{\omega^n}$.

Proof (Sketch). Show that if $x_i > 0$ for some i then $G_{g,n}(x_{n-1}, \dots, x_0, t) \geq F_\alpha(t)$, where $\alpha = \omega^{n-1} \cdot x_{n-1} + \dots + x_0 \cdot \omega^0$ by induction in (x_{n-1}, \dots, x_0) . Use monotonicity of $G_{g,n}$ and the fact that $g(x) \geq x + 2$. Finally, for all t we have

$$\begin{aligned} L_{g,n}^{\text{ms}}(t) &\geq P_{g,n}(g(t) - 1, \bar{0}, t) \\ &\geq P_{g,n}(t + 1, \bar{0}, t) \\ &\geq G_{g,n}(t + 1, \bar{0}, t) \\ &\geq F_{\omega^{n-1} \cdot (t+1)}(t) = F_{\omega^n}(t). \end{aligned}$$

The second inequality follows from the monotonicity of $P_{g,n}$ and $g(x) \geq x + 2$; the third one from Lem. [11](#). \square

5.3 Upper Bound

Define $U_{g,n} : \mathbb{N}^{n+1} \setminus \{(0, \dots, 0)\} \rightarrow \mathbb{N}$ by multiple recursion as:

$$U_{g,n}(0, \dots, 0, 1, t) \stackrel{\text{def}}{=} g(t + 1) \quad (7)$$

$$U_{g,n}(\bar{x}, x_n + 1, t) \stackrel{\text{def}}{=} g(t + 1) \cdot U_{g,n}(\bar{x}, x_n, o_{x_1, \dots, x_n}^{g(t+1)-1}(t + 2)) \quad (8)$$

$$U_{g,n}(\bar{x}, x_j + 1, \bar{0}, t) \stackrel{\text{def}}{=} U_{g,n}(\bar{x}, x_j, g(t + 1), \bar{0}, t + 2) \quad (9)$$

where $o_{x_1, \dots, x_n}(t) = t + U_{g,n}(x_1, \dots, x_{n-1}, x_n, t)$; equation [\(8\)](#) applies when $x_i > 0$ and $\bar{x} = x_1, \dots, x_{n-1}$; and equation [\(9\)](#) applies when $j < n$ and $\bar{x} = x_1, \dots, x_{j-1}$.

Lemma 13. $P_{g,n} \leq U_{g,n}$.

Proof. By induction in the lexicographic order of x_1, \dots, x_n . For [\(7\)](#), as in the proof of Lem. [11](#), the longest g, t -controlled $<_{\text{ms}}$ -decreasing sequence starting with $1 \cdot \{(\bar{0}, 1)\}$ has length $g(t + 1) + 1$ and then $P_{g,n}(\bar{0}, 1, t) = g(t + 1) = U_{g,n}(\bar{0}, 1, t)$. For [\(8\)](#) the longest g, t -controlled $<_{\text{ms}}$ -decreasing sequence starting with $M_0 = 1 \cdot \{(\bar{x}, x_n + 1)\}$ continues with a multiset M_1 whose $<_{\text{lex}}$ -maximum element is (\bar{x}, x_n) , of multiplicity $g(t + 1) - 1$. Therefore if $N = g(t + 1) \cdot \{(\bar{x}, x_n)\}$ then $M_0 >_{\text{ms}} N >_{\text{ms}} M_1$ and N is $g, (t + 2)$ -controlled. Hence

$$\begin{aligned} P_{g,n}(\bar{x}, x_n + 1, t) &\leq P_{g,n}(g(t + 1) \cdot \{(\bar{x}, x_n)\}, t + 2) \\ &\leq g(t + 1) \cdot P_{g,n}(\bar{x}, x_n, \tilde{o}_{x_1, \dots, x_n}^{g(t+1)-1}(t + 2)) \\ &\leq g(t + 1) \cdot U_{g,n}(\bar{x}, x_n, o_{x_1, \dots, x_n}^{g(t+1)-1}(t + 2)) = U_{g,n}(\bar{x}, x_n + 1, t) \end{aligned}$$

where $\tilde{o}_{x_1, \dots, x_n}(t) = t + P_{g,n}(x_1, \dots, x_n, t)$, the second inequality follows from Cor. [10](#), and the third one from ind. hyp. and monotonicity of $U_{g,n}$. For [\(9\)](#) the longest g, t -controlled $<_{\text{ms}}$ -decreasing sequence of multisets starting with $M'_0 = 1 \cdot \{(\bar{x}, x_j + 1, \bar{0})\}$ continues with a multiset M'_1 whose $<_{\text{lex}}$ -maximum element is $(\bar{x}, x_j, g(t + 1) - 1, \dots, g(t + 1) - 1)$, of multiplicity $g(t + 1) - 1$. Then $M'_0 >_{\text{ms}} N' >_{\text{ms}} M'_1$, where $N' = 1 \cdot \{(\bar{x}, x_j, g(t + 1), \bar{0})\}$, and hence N' is $g, (t + 2)$ -controlled. Therefore by inductive hypothesis we have

$$\begin{aligned} P_{g,n}(\bar{x}, x_j + 1, \bar{0}, t) &\leq P_{g,n}(\bar{x}, x_j, g(t + 1), \bar{0}, t + 2) \\ &\leq U_{g,n}(\bar{x}, x_j, g(t + 1), \bar{0}, t + 2) = U_{g,n}(\bar{x}, x_j + 1, \bar{0}, t), \end{aligned}$$

and this concludes the proof. \square

Theorem 14. *If g is primitive recursive and $g(t) \geq t+1$ then $L_{g,n}^{\text{ms}}$ has an upper bound in \mathfrak{F}_{ω^n} . Also, this bound is tight.*

Proof. The fact that the bound is tight follows from Thm. 12. Without loss of generality suppose, $t > 2$ and let $2 \leq e < \omega$ such that $g(t+1) \leq F_e(t)$. By $(\forall^\infty x)\varphi(x)$ we mean that φ holds for almost every x , i.e. $(\exists k)(\forall x > k)\varphi(x)$.

Fact 4. *If $x \neq 0$ then $(\forall^\infty t)(\forall x)U_{g,n}(\bar{0}, x, t) \leq F_{3(x-1)+e}(t)$.*

Proof. By induction in $x \neq 0$. For $x = 1$, observe that $U_{g,n}(\bar{0}, 1, t) = g(t+1) \leq F_e(t)$. For the inductive step, $\sigma_{\bar{0},x} = t + U_{g,n}(\bar{0}, x, t) \leq t + F_{3(x-1)+e}(t) \leq F_{3(x-1)+e+1}(t)$. Now

$$\begin{aligned}
 U_{g,n}(\bar{0}, x+1, t) &= g(t+1) \cdot U_{g,n}(\bar{0}, x, o_{\bar{0},x}^{g(t+1)-1}(t+2)) \\
 &\leq g(t+1) \cdot F_{3(x-1)+e}(o_{\bar{0},x}^{g(t+1)-1}(t+2)) && \text{(ind. hyp.)} \\
 &\leq F_e(t) \cdot F_{p(x)}(F_{p(x)+1}^{g(t+1)-1}(t+2)) && (p(x) \stackrel{\text{def}}{=} 3(x-1)+e) \\
 &\leq F_e(t) \cdot F_{p(x)}(F_{p(x)+1}^{g(t+1)+1}(g(t+1))) \\
 &= F_e(t) \cdot F_{p(x)}(F_{p(x)+2}(g(t+1))) \\
 &\leq F_{p(x)+2}(F_{p(x)+2}(F_{p(x)+2}(F_{p(x)+2}(t)))) \\
 &= F_{p(x)+2}^4(t) \leq F_{p(x)+3}(t) = F_{3x+e}. && (t \geq 3)
 \end{aligned}$$

This concludes the proof of the Fact \square

Fact 5. *If $x_0 > 0$ then $(\forall^\infty t)(\forall x_{n-1}, \dots, x_0)[U_{g,n}(x_{n-1}, \dots, x_0, t) \leq F_\gamma(t) \Rightarrow U_{g,n}(x_{n-1}, \dots, x_0+1, t) \leq F_{\gamma+3}(t)]$.*

Proof. Same idea as in Fact 4. \square

Fact 6. *If $x_i > 0$ for some $i \geq 1$ then $(\forall^\infty t)(\forall \bar{x} = x_{n-1}, \dots, x_1)U_{g,n}(\bar{x}, 0, t) \leq F_\alpha(t)$, where $\alpha = x_{n-1} \cdot \omega^{n-1} + x_{n-2} \cdot \omega^{n-2} + \dots + x_2 \cdot \omega^2 + x_1 \cdot \omega + 1$.*

Proof. By induction in $\bar{x} \neq \bar{0}$. $U_{g,n}(\bar{0}, 1, 0, t) = U_{g,n}(\bar{0}, g(t+1), t+2) \leq F_{d(t)}(t+2) \leq F_{d(t)+1}(d(t)) = F_\omega(d(t)) \leq F_{\omega+1}(t)$ where $d(t) \stackrel{\text{def}}{=} 3(g(t+1)-1)+e$, the first inequality follows from Fact 4 and the last one is true for all $t \geq k_1$.

Next, $U_{g,n}(\bar{0}, x_1+1, 0, t) = U_{g,n}(\bar{0}, x_1, g(t+1), t+2) \leq F_{x_1 \cdot \omega+1+r(t)}(t+2) \leq F_{x_1 \cdot \omega+1+r(t)}(r(t)) = F_{(x_1+1) \cdot \omega}(r(t)) \leq F_{(x_1+1) \cdot \omega+1}(t)$ where $r(t) \stackrel{\text{def}}{=} 3g(t+1)$, the first inequality follows from ind. hyp. and Fact 5 and the last one is true for all $t \geq k_2 \geq k_1$ (independently of x_1).

Finally, let $\bar{x} = x_{n-1}, \dots, x_{j-1}$ and let $\beta = x_{n-1} \cdot \omega^{n-1} + \dots + x^{j-1} \cdot \omega^{j-1}$.

$$\begin{aligned}
 U_{g,n}(\bar{x}, x_j+1, \bar{0}, t) &= U_{g,n}(\bar{x}, x_j, g(t+1), \bar{0}, t+2) \\
 &\leq F_{\beta+x_j \cdot \omega^j+g(t+1) \cdot \omega^{j-1}+1}(t+2) && \text{(ind. hyp.)} \\
 &\leq F_{\beta+x_j \cdot \omega^j+(g(t+1)+1) \cdot \omega^{j-1}}(t+2) \\
 &\leq F_{\beta+x_j \cdot \omega^j+(g(t+1)+1) \cdot \omega^{j-1}}(g(t+1)) \\
 &\leq F_{\beta+(x_j+1) \cdot \omega^j}(g(t+1)) \leq F_{\beta+(x_j+1) \cdot \omega^j+1}(t),
 \end{aligned}$$

where the last inequality is true for all $t \geq k_3 \geq k_2$ (independently of \bar{x}, x_j). \square

Now, let t be sufficiently large. If $n = 1$ then $L_{g,n}^{\text{ms}}(t) \leq P_{g,n}(g(t), t + 1) \leq U_{g,n}(g(t), t + 1) \leq F_{3(g(t)-1)+e}(t + 1) \leq F_{3(g(t)-1)+e+1}(3(g(t)-1)+e) = F_{\omega}(3(g(t)-1) + e) \in \mathfrak{F}_{\omega}$, where the second inequality follows from Lem. 1.3 and the third one from Fact 4. If $n > 1$ we have:

$$\begin{aligned} L_{g,n}^{\text{ms}}(t) &\leq P_{g,n}(g(t), \bar{0}, t + 1) \\ &\leq U_{g,n}(g(t), \bar{0}, t + 1) \\ &\leq F_{g(t) \cdot \omega^{n-1} + 1}(t + 1) \\ &\leq F_{(g(t)+1) \cdot \omega^{n-1}}(g(t)) = F_{\omega^n}(g(t)) \in \mathfrak{F}_{\omega^n}. \end{aligned}$$

The second inequality follows from Lem. 1.3 and the third one from Fact 6. \square

6 The Majoring Ordering

Recall from §2 that the underlying order of \leq_{maj} is \leq_{pr} and the underlying order of \leq_{ms} is \leq_{lex} . We linearize the wqo $(\mathcal{P}_{< \infty}(\mathbb{N}^n), \leq_{\text{maj}})$ into the well-order $(\mathcal{M}_{< \infty}(\mathbb{N}^n), \leq_{\text{ms}})$ and derive an upper bound for $L_{n,g}^{\text{maj}}(t)$, the length of the longest g, t -controlled bad sequence of finite sets of n -tuples with respect to the majoring ordering \leq_{maj} . To do this, we use the results of §5.

Our linearization will be done in two steps. Given a \leq_{maj} -bad sequence $\mathbf{X} = X_0, X_1, \dots, X_k$ of finite and nonempty sets of n -tuples we define an intermediate sequence T_0, T_1, \dots, T_k of trees whose nodes are decorated with n -tuples. From these trees we define a sequence of finite and nonempty multisets of n -tuples $\mathbf{M} = M_0, M_1, \dots, M_k$. We show that if \mathbf{X} is \leq_{maj} -bad then \mathbf{M} is $<_{\text{ms}}$ -decreasing. Furthermore, given a control for \mathbf{X} , we find a control for \mathbf{M} . Using the results of §5 we give an answer to the question of the maximum possible length of a controlled \leq_{maj} -bad sequence of finite sets of n -tuples.

Let $X \subseteq \mathbb{N}^n$. We say X avoids x if for all $y \in X$ we have $x \not\leq_{\text{pr}} y$. Since $\mathbf{X} = X_0, X_1, \dots, X_k$ is bad, then for any $i < j$, X_j avoids some tuple of X_i . In particular for all $j \in \{1, \dots, k\}$, X_j avoids some tuple of X_0 . If a is the \leq_{pr} -supremum of X_0 then $\tilde{\mathbf{X}} = \{a\}, X_1, \dots, X_k$ is also a bad sequence. Furthermore, if \mathbf{X} was g, t -controlled then $\tilde{\mathbf{X}}$ also is, and in this case $a \leq_{\text{pr}} \langle g(t) - 1, \dots, g(t) - 1 \rangle$. Even more, if \mathbf{X} is the longest such sequence then $a = \langle g(t) - 1, \dots, g(t) - 1 \rangle$. Therefore, without loss of generality we may assume that all \leq_{maj} -bad sequences of sets analyzed here have a singleton as the first element.

Construction of the trees T_i . Without loss of generality suppose $X_0 = \{a_0\}$. Define the following sequence of finite trees of n -tuples. By *path* we always refer to a path from the root to a leaf. See Fig. 1 for an example of this construction.

- T_0 is a_0 , the root.
- T_{i+1} is formed by extending T_i as follows. For any path a_0, \dots, a_m in T_i do the following: if for all $j = 0, \dots, m$, X_{i+1} avoids a_j then add all the elements of X_{i+1} as new children of a_m .

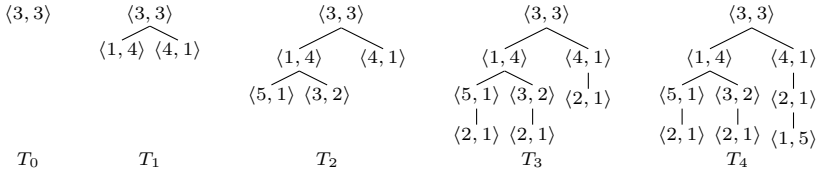


Fig. 1. Construction of the trees for the bad sequence X_0, X_1, X_2, X_3, X_4 , where $X_0 = \{(3, 3)\}$; $X_1 = \{(1, 4), (4, 1)\}$; $X_2 = \{(5, 1), (3, 2)\}$; $X_3 = \{(2, 1)\}$; $X_4 = \{(1, 5)\}$

Proposition 15. *At least one path of T_i is strictly extended in T_{i+1} .*

Proof. Recall that $X_j \neq \emptyset$ for all j . It is clear that if all internal nodes of T_i have a child which is avoided by X_{i+1} then there is a path a_0, \dots, a_m in T_i such that X_{i+1} avoids a_j for all j .

If $T_{i+1} = T_i$ then, by construction, there is no path a_0, \dots, a_m with all of its elements avoided by X_{i+1} . Then there is an internal node of T_i , say a , with none of its children avoided by X_{i+1} . But this contradicts the badness of \mathbf{X} since by construction the set of children of a is X_j for some $j \leq i$. \square

As the example in Fig. 1 shows, the height of T_{i+1} is not necessarily greater than the height of T_i . The following follows by construction:

Proposition 16. *Any path in T_i is a bad sequence of n -tuples with respect to the product ordering. Furthermore if \mathbf{X} is g, t -controlled then any such path is $g, (t + i)$ -controlled.*

Construction of the multisets M_i . Let $M_i \in \mathcal{M}_{<\infty}(\mathbb{N}^n)$ be defined as: $M_i(y) \stackrel{\text{def}}{=} d$ iff there are exactly d paths in T_i , say p_1, \dots, p_d , such that $h_n(p_j) = y$ for all j . In other words, M_i is the multiset where we put $h_n(p)$ for every path in T_i .

If the path $\bar{a} = a_1, \dots, a_m$ in T_i is extended to \bar{a}, x in T_{i+1} then by Thm. 4, $h_n(\bar{a}, x) <_{\text{lex}} h_n(\bar{a})$. Then $M_{i+1} <_{\text{ms}} M_i$. The need for working with multisets and not simply with sets resides in the fact that h is not injective.

Proposition 17. *If $\mathbf{X} = X_0, \dots, X_k$ is g, t -controlled then $|M_k| < \tilde{g}(t + k)$, for $\tilde{g}(x) = n! g(nx)^{n(x+1)} + 1$.*

Proof. Observe that the maximum multiplicity of an element in M_k is bounded by $\prod_{j=1}^k g(t + j)^n \leq g(t + k)^{nk} < \tilde{g}(t + k)$. By Prop. 16 each of such path is $g, (t + k)$ -controlled and by the second part of Thm. 4 we have that if $x \in M_k$ then $|x|_\infty < n! g(n(k + t))^n < \tilde{g}(t + k)$. \square

Altogether we have shown:

Theorem 18. *There is a function $f_n : (\mathcal{P}_{<\infty}(\mathbb{N}^n))^+ \rightarrow \mathcal{M}_{<\infty}(\mathbb{N}^n)$ such that if $\mathbf{X} \hat{\ } X$ is a bad sequence in $(\mathcal{P}_{<\infty}(\mathbb{N}^n), \leq_{\text{maj}})$, \mathbf{X} is nonempty and X is a nonempty set, then $f_n(\mathbf{X} \hat{\ } X) <_{\text{ms}} f_n(\mathbf{X})$. Furthermore if \mathbf{X} is g, t -controlled then $|f_n(\mathbf{X})| < \tilde{g}(|\mathbf{X}| - 1 + t)$, for \tilde{g} as in Prop. 17.*

Proof. Take $f_n(\mathbf{X}) = M_{|\mathbf{X}|-1}$ as in the above construction. □

Let $L_{n,g}^{\text{maj}}(t)$ denote the length of the longest g, t -controlled bad sequence in $(\mathbb{N}^n, \leq_{\text{maj}})$, and let $L_{n,g}^{\text{ms}}(t)$ denote the length of the longest g, t -controlled decreasing sequence in $(\mathbb{N}^n, <_{\text{ms}})$.

Corollary 19. *For any primitive recursive g there is a primitive recursive \tilde{g} such that $L_{n,g}^{\text{maj}} \leq L_{n,\tilde{g}}^{\text{ms}}$. Hence there is an upper bound of $L_{n,g}^{\text{maj}}$ in \mathfrak{F}_{ω^n} .*

Proof. It follows from Thm. 18 and Thm. 14. □

7 Applications

In Jurdziński and Lazić [9] it is shown that for the class of incrementing tree counter automata (ITCA) as well as the class of alternating top-down tree one register automata (ATRA), the emptiness problem —i.e. whether the language accepted by an automaton of such classes is empty— is decidable over finite data trees. Figueira [4] later showed that for some extensions of ATRA decidability still holds. All these proofs go along the lines of interpreting the automata execution as a downward well-structured transition system, then showing that it is reflexive-downward-compatible with respect to a wqo between sets of configurations, and finally applying Finkel and Schnoebelen results [6] (mainly Prop. 5.4). That wqo is precisely the majoring order.

From [9], we know that the computational complexity of such decision procedures is lower-bounded by a non-primitive recursive function. For the upper-bound for ITCA’s, an algorithm can be given in a manner analogous to [5, §VII.B.] for finding the *levels* (a finite set of configurations) reachable from the initial level —the emptiness problem is then reduced to testing whether the empty level is amongst them. The complexity of such an algorithm is mainly determined by the length of a bad sequence of levels $\mathbf{V} = V_0, V_1, \dots, V_m$. In more detail, suppose an ITCA \mathcal{C} has k counters and a finite set of states Q . Then a level of \mathcal{C} is a finite set of tuples of the form $\langle q, v \rangle$, where $q \in Q$ and $v = \langle a_1, \dots, a_k \rangle \in \mathbb{N}^k$ is the current values of the k counters. The levels are ordered by the majoring ordering with the following underlying order

$$\langle p, u \rangle \leq \langle q, v \rangle \stackrel{\text{def}}{\iff} p = q \wedge u \leq_{\text{pr}} v,$$

which is a wqo. The complexity of the emptiness problem can be bounded by the length of the longest bad sequence in $(\mathcal{P}_{<\infty}(Q \times \mathbb{N}^k), \leq_{\text{maj}}^{(\le)})$. As one can see, the application of Cor. 19 is not entirely straightforward because it applies to the majoring ordering of finite sets of tuples of \mathbb{N} with the underlying \leq_{pr} and not to levels with the underlying \leq . We reduce bad sequences of levels to bad sequences of finite sets of tuples as follows. Suppose $Q = \{q_0, \dots, q_{s-1}\}$ and let $q'_i \stackrel{\text{def}}{=} (i, s - i) \in \mathbb{N}^2$. Clearly if $p' \leq_{\text{pr}} q'$ then $p' = q'$ and so $p = q$. Let $V \in \mathcal{P}_{<\infty}(Q \times \mathbb{N}^k)$ be a level. Define $V' \stackrel{\text{def}}{=} \{\langle p', u \rangle \in \mathbb{N}^{k+2} \mid \langle p, u \rangle \in V\}$. The reader can verify that if V and W are levels then $V' \leq_{\text{maj}}^{(\le_{\text{pr}})} W'$ implies

$V \leq_{\text{maj}}^{(\leq)} W$. Hence $\mathbf{V} = V_0, V_1, \dots, V_m$, a bad sequence of levels of an ITCA with k counters, can be seen as a bad sequence of the same length $\mathbf{V}' = V'_0, V'_1, \dots, V'_m$ in $\mathcal{P}_{<\infty}(\mathbb{N}^{k+2})$ with the majoring ordering studied in §6. Regarding how \mathbf{V}' is controlled, the analysis is almost the same as in [5, §VII.B.]. Let $V'_0 = \{\langle 0, |Q| - 1, \bar{0} \rangle\}$ and $V'_i = \{c_1, \dots, c_{p_i}\}$. From Def. 6 we have that $|V'_i| = \max_j \{ |c_j|_\infty \}$. The change from V'_i to V'_{i+1} may involve a change of state or increment of c_j 's counters' values by one. The 'state part' of c_j is controlled by the constant $|Q|$ and the 'counters part' is controlled by the successor function. Hence, the bad sequence of sets is $g, 0$ -controlled by $g(t) = t + 1 + |Q|$. Now we can finally apply Cor. 19 to conclude that *the complexity of the emptiness problem for an ITCA with k counters is upper bounded by a function in $\mathfrak{F}_{\omega^{k+2}}$* .

This immediately gives us an upper bound for the emptiness problem for ATRA. From [9, Thm. 3.1] we have that emptiness for ATRA follows from a PSPACE-reduction to emptiness for ITCA. If the ATRA \mathcal{A} has s states then the ITCA \mathcal{C} constructed in the reduction has $k(s) \stackrel{\text{def}}{=} 2^s - 1 + 2^{4s}$ counters.¹ Hence *the complexity of the emptiness problem for an ATRA with s states is upper bounded by a function in $\mathfrak{F}_{\omega^{k(s)+2}}$* .

The above complexities are obtained by the straightforward codification of levels V into V' . This increases the dimension of tuples from n to $n + 2$, and this might be too wasteful. It seems plausible to work directly with levels (i.e. sets of $Q \times \mathbb{N}^n$) and obtain better upper bounds.

8 Conclusions

Upper bounds for controlled descending sequences in a well-order are easier to obtain than for controlled bad sequences in a wqo's. We studied upper bounds for the length of controlled bad sequences of two wqo's by linearizing them into well-orders. Such bounds were placed in the Fast Growing Hierarchy.

For the product ordering of tuples, we gave a straightforward elementary proof for an upper bound of controlled bad sequences, and we arrived to the same general result as [5] but avoiding the "sum of powers of \mathbb{N} " approach. This last approach —being noticeably more understandable than previous proofs, and also leading to a more general result— still needs some rather technical lemmas. Our proof simply relies on a linearization of controlled bad sequences of tuples in the *product* ordering into controlled descending sequences of tuples in the *lexicographic* ordering, for which upper bounds can be easily obtained.

For the *majoring* ordering of sets of tuples, we gave an upper bound of controlled bad sequences over such wqo by linearizing to controlled and descending sequences of multisets with the natural *multiset* ordering. For the latter we also gave a tight upper bound, which is of interest by itself. As applications we showed complexity upper bounds for the emptiness problem for two types counter automata: ITCA and ATRA.

¹ In [9] there is typo in the number of counters in the auxiliary array c' . Where it says $2^{|Q|^4}$, it should read $2^{4|Q|}$.

The fact that $(\mathcal{P}_{<\infty}(\mathbb{N}^n), \leq_{\text{maj}})$ is a wqo follows from reducing finite sets of tuples to finite strings over \mathbb{N}^n and then applying Higman's Lemma. Schmitz and Schnoebelen [13] developed an algebraic framework for handling normed wqo's where upper bounds for controlled bad sequences when using Higman's Lemma on finite alphabets are derived. Hence, another approach to obtain upper bounds for the majoring ordering would be to try to extend this framework to deal with strings over infinite alphabets.

As future research we will study lower bounds for the majoring ordering, and upper bounds for the bad sequences over the dual of the majoring ordering, the *minoring* ordering:

$$A \leq_{\min}^{(\leq)} B \stackrel{\text{def}}{\Leftrightarrow} (\forall y \in B)(\exists x \in A) x \leq y.$$

This is not in general a wqo: one needs the underlying \leq to be an ω^2 -wqo [8, Thm. 1]. It would also be interesting to investigate how far one can generalize this idea of linearization. Are there general ways in which one can relate the length of a bad sequence over a wqo into the length of a linearization of it?

References

1. Abriola, S.A.: Sobre la longitud de las secuencias malas controladas en cuasi-órdenes buenos. MSc thesis, Universidad de Buenos Aires, Argentina (2011)
2. Clote, P.: On the finite containment problem for Petri nets. *Theoretical Computer Science* 43, 99–105 (1986)
3. Dershowitz, N., Manna, Z.: Proving termination with multiset orderings. *Communications of the ACM* 22(8), 465–476 (1979)
4. Figueira, D.: Reasoning on Words and Trees with Data. PhD thesis, Laboratoire Spécification et Vérification, ENS Cachan, France (December 2010)
5. Figueira, D., Figueira, S., Schmitz, S., Schnoebelen, P.: Ackermannian and primitive-recursive bounds with Dickson's lemma. In: LICS, pp. 269–278 (2011)
6. Finkel, A., Schnoebelen, P.: Well-structured transition systems everywhere? *Theoretical Computer Science* 256(1-2), 63–92 (2001)
7. Harwood, W., Moller, F., Setzer, A.: Weak Bisimulation Approximants. In: Ésik, Z. (ed.) CSL 2006. LNCS, vol. 4207, pp. 365–379. Springer, Heidelberg (2006)
8. Jancar, P.: A note on well quasi-orderings for powersets. *Information Processing Letters* 72(5-6), 155–160 (1999)
9. Jurdziński, M., Lazić, R.: Alternating automata on data trees and XPath satisfiability. *ACM Transactions on Computational Logic (TOCL)* 12(3), 19 (2011)
10. Löb, M.H., Wainer, S.S.: Hierarchies of number theoretic functions, I. *Archive for Mathematical Logic* 13, 39–51 (1970)
11. McAloon, K.: Petri nets and large finite sets. *Theoretical Computer Science* 32(1-2), 173–183 (1984)
12. Péter, R.: Recursive functions. Academic Press (1967)
13. Schmitz, S., Schnoebelen, P.: Multiply-Recursive Upper Bounds with Higman's Lemma. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part II. LNCS, vol. 6756, pp. 441–452. Springer, Heidelberg (2011)

Initiality for Typed Syntax and Semantics

Benedikt Ahrens

Université Nice Sophia Antipolis, France

Abstract. We give an algebraic characterization of the syntax and semantics of a class of simply-typed languages, such as the language PCF: we characterize simply-typed binding syntax equipped with reduction rules via a universal property, namely as the initial object of some category. For this purpose, we employ techniques developed in two previous works: in [2], we model syntactic translations between languages *over different sets of types* as initial morphisms in a category of models. In [1], we characterize untyped syntax *with reduction rules* as initial object in a category of models. In the present work, we show that those techniques are modular enough to be combined: we thus characterize simply-typed syntax with reduction rules as initial object in a category. The universal property yields an operator which allows to specify translations — that are semantically faithful by construction — between languages over possibly different sets of types.

We specify a language by a *2-signature*, that is, a signature on two levels: the *syntactic* level specifies the types and terms of the language, and associates a type to each term. The *semantic* level specifies, through *inequations*, reduction rules on the terms of the language. To any given 2-signature we associate a category of models. We prove that this category has an initial object, which integrates the types and terms freely generated by the 2-signature, and the reduction relation on those terms generated by the given inequations. We call this object the (*programming*) *language generated by the 2-signature*.

1 Introduction

We give a characterization, via a universal property, of the syntax and semantics of simply-typed languages with variable binding. More precisely, we characterize the terms and sorts associated to a signature equipped with reduction rules as the initial object in a category of models. Initiality in this category gives rise to an iteration principle (cf. [Rem. 45](#)) which allows to specify translations between languages in a convenient way as initial morphisms. The category of models is sufficiently large — and thus the iteration principle stemming from initiality is sufficiently general — to account for translations between languages over different sets of sorts. Furthermore, translations specified via this principle are ensured to be faithful with respect to reduction in the source and target languages, as well as compatible in a suitable sense with substitution on either side.

To illustrate the iteration operator stemming from initiality, we use it to specify a translation from PCF to the untyped lambda calculus ULC. We do

so in the proof assistant Coq [5]; for this purpose, we prove formally, in Coq, an instance of our main theorem for the 2-signature of PCF: the types and terms of PCF, equipped with their usual reductions, form an initial object in the category of models of PCF. We then use the iteration principle to obtain an initial morphism — a translation, faithful with respect to reductions — to ULC, as an executable Coq function. The Coq theory files as well as documentation are available online at <http://math.unice.fr/laboratoire/logiciels>.

Summary. We define a notion of *2-signature* which allows the specification of the *types and terms* of a language — via an underlying 1-signature — as well as its *semantics* in form of reduction rules. A 1-signature (S, Σ) is given by a pair of a signature S for types and a binding signature Σ for terms typed over the set of types associated to S . Reduction rules for terms generated by Σ are specified via a set A of inequations over (S, Σ) . A 2-signature $((S, \Sigma), A)$ is a pair of a 1-signature (S, Σ) and a set A of inequations over (S, Σ) . To such a 2-signature we associate a category of representations, for which the types and terms generated by (S, Σ) , equipped with reductions according to A , forms an initial object.

1-signatures are defined in [2]. There, we associate a category $\text{Rep}(S, \Sigma)$ of representations to any 1-signature (S, Σ) , and show that the types and terms freely generated by (S, Σ) form an initial object in this category. Representations there are built from monads on families of sets. In the present work, we build a different category $\text{Rep}^\Delta(S, \Sigma)$ of representations using *relative* monads from sets to preordered sets, which allows — in a second step, cf. below — the integration of reduction rules to account for semantic aspects. The two categories of representations, $\text{Rep}(S, \Sigma)$ and $\text{Rep}^\Delta(S, \Sigma)$, are connected through an adjunction which transports the initial object of the former to the latter category (cf. [Lem. 34](#)).

Inequations over untyped 1-signatures are considered in [1]. There, we define a notion of 2-signature for untyped syntax with semantics in form of reduction rules and show that its associated category of representations has an initial object. In the present work, we define inequations over *typed* 1-signatures as defined in [2]. Given a set A of inequations over a 1-signature (S, Σ) , the representations of (S, Σ) that *satisfy* each inequation of A , form a full subcategory of $\text{Rep}^\Delta(S, \Sigma)$, which we call the category of representations of (S, Σ, A) . Our main theorem (cf. [Thm. 44](#)) states that this category has an initial object, which integrates the types and terms freely generated by (S, Σ) , equipped with reduction rules generated by the inequations of A .

Related Work. Related work is reviewed extensively in [1,2], as well as in the author’s PhD thesis [3]. We give a brief overview: rewriting in nominal settings is examined by Fernández and Gabbay [6]. Ghani and Lüth [8] present rewriting for algebraic theories without variable binding; they characterize equational theories resp. rewrite systems as *coequalizers* resp. *coinserters* in a category of monads on the categories Set resp. Pre . Fiore and Hur [7] have extended Fiore’s work to integrate semantic aspects into initiality results. In particular, Hur’s thesis

[12] is dedicated to *equational* systems for syntax with variable binding. In a “Further research” section [12, Chap. 9.3], Hur suggests the use of preorders, or more generally, arbitrary relations to model *inequational* systems. Hirschowitz and Maggesi [9] prove initiality of the set of lambda terms modulo beta and eta conversion in a category of *exponential monads*. In an unpublished paper [10], they define a notion of *half-equation* and *equation* to express congruence between terms. We adopt their definition in this paper, but interpret a pair of half-equations as *inequation* rather than equation.

2 Relative Monads and Modules

Relative monads were defined by Altenkirch et al. [4] to overcome the restriction of (regular) monads to *endofunctors*. In an earlier work [1], we define morphisms of relative monads and *modules over relative monads*. In the following section we define a more general notion of *colax* morphism of relative monads — which we use in Sect. 3 to model translations between languages over different sets of types — and generalize constructions of [1] to such colax morphisms. Some definitions from [12] which we use in the present work, are recalled at the beginning.

We denote by \mathbf{Set} the category of sets and total maps of sets. We call \mathbf{Pre} the category of preordered sets and monotone maps between them.

Definition 1. We call $\Delta : \mathbf{Set} \rightarrow \mathbf{Pre}$ the left adjoint of the forgetful functor $U : \mathbf{Pre} \rightarrow \mathbf{Set}$. The functor Δ associates to each set X the set itself together with the smallest preorder, i.e. the diagonal of X , $\Delta(X) := (X, \delta_X)$.

Definition 2 (Category of Families). Let \mathcal{C} be a category and T be a set, i.e. a discrete category. We denote by \mathcal{C}^T the functor category, an object of which is a T -indexed family of objects of \mathcal{C} . We write $V_t := V(t)$ for objects and morphisms. Given a functor $F : \mathcal{C} \rightarrow \mathcal{D}$, we denote by $F^T : \mathcal{C}^T \rightarrow \mathcal{D}^T$ the induced functor.

Definition 3 (Relative Monad on Δ^T , enriched). We strengthen the definition of a relative monad P on Δ^T by requiring the substitution map $\sigma_{X,Y}$ to be monotone with respect to the preorders induced by the preorders on PY ,

$$\sigma_{X,Y} : \mathbf{Pre}^T(\Delta X, PY) \rightarrow \mathbf{Pre}^T(PX, PY) .$$

From now on, a relative monad on Δ^T is meant to be enriched in the sense of Def. 3, i.e. monotone in both the first- and the higher-order argument.

Example 4 (Lambda Calculus as Relative Monad on Δ^T). Let $T := T_{\text{TLC}}$ be the set of types of the simply-typed lambda calculus, built from a base type and a binary arrow constructor. Given a set family $V \in \mathbf{Set}^{T_{\text{TLC}}}$, we denote by $\text{TLC}(V) \in \mathbf{Set}^{T_{\text{TLC}}}$ the set family of simply-typed lambda terms over T_{TLC} in context V , which might be implemented in the proof assistant Coq as follows:

Inductive TLC ($V : T \rightarrow \text{Type}$) : $T \rightarrow \text{Type} :=$
 | **Var** : forall t, V t \rightarrow TLC V t
 | **Abs** : forall s t, TLC (V + s) t \rightarrow TLC V (s \sim > t)
 | **App** : forall s t, TLC V (s \sim > t) \rightarrow TLC V s \rightarrow TLC V t.

Here $V + s$ is a notation denoting the context V extended by a fresh variable of type s — the variable that is bound by the constructor `Abs s t`. We occasionally leave the object type arguments of the constructors implicit and write λM and $M(N)$ for `Abs s t M` and `App s t M N`, respectively. The set family of lambda terms is equipped with a structure of a monad TLC on the category $\text{Set}^{T_{\text{TLC}}}$ as follows [11]: the family η^{TLC} is given by the family of constructors `Var`, and the substitution map is given by capture-avoiding simultaneous substitution:

$$\sigma_{X,Y} : \text{Set}^T(X, \text{TLC}(Y)) \rightarrow \text{Set}^T(\text{TLC}(X), \text{TLC}(Y)) .$$

Similarly, with the same operations η and σ , we can consider it as a *relative* monad on the functor $\Delta^{T_{\text{TLC}}}$,

$$\text{TLC}\Delta : \text{Set}^{T_{\text{TLC}}} \rightarrow \text{Pre}^{T_{\text{TLC}}} .$$

The underlying object map $\text{TLC}\Delta$ associates, to each set family V , the family of lambda terms in context V , equipped with the diagonal preorder, corresponding to *syntactic equality*:

$$\text{TLC}\Delta : V \mapsto (\text{TLC}(V), \delta_{\text{TLC}(V)}) .$$

We equip each set $\text{TLC}(V)(t)$ of lambda terms over context V of object type t with a preorder taken as the reflexive-transitive closure of the relation generated by the *beta* rule

$$\lambda M(N) \leq M[* := N] \tag{\beta}$$

and its propagation into subterms:

$$\text{TLC}\beta : V \mapsto (\text{TLC}(V), \beta_{\text{TLC}(V)}^*) .$$

The beta rule in Disp. (β) states that the application of a lambda abstraction with body M to an argument N reduces to the term M in which the term N is substituted for the fresh variable of M — recall from above that M lives in an extended context — in a capture-avoiding manner. This assignment defines a relative monad $\text{TLC}\beta$ on the functor $\Delta^T : \text{Set}^T \rightarrow \text{Pre}^T$.

Modules over relative monads and their morphisms are defined in [11], together with several constructions of modules. Recall that modules over P with codomain \mathcal{E} and morphisms between them form a category called $\text{RMod}(P, \mathcal{E})$. We give some examples of modules and module morphisms over the monad $\text{TLC}\beta$, which hold analogously for the monad $\text{TLC}\Delta$:

Example 5 (Ex. 4 cont.). The map $\text{TLC}\beta : V \mapsto \text{TLC}\beta(V)$ yields a module over the relative monad $\text{TLC}\beta$, the *tautological* $\text{TLC}\beta$ -module $\text{TLC}\beta$. Given $V \in \text{Set}^T$ and $s \in T$, we denote by V^s the context V enriched by an additional variable of type s . The map $\text{TLC}\beta^s : V \mapsto \text{TLC}\beta(V^s)$ inherits the structure of a $\text{TLC}\beta$ -module from the tautological module $\text{TLC}\beta$. We call $\text{TLC}\beta^s$ the *derived module with respect to* $s \in T$ of the module $\text{TLC}\beta$. Given $t \in T$, the map $V \mapsto \text{TLC}\beta(V)(t) : \text{Set}^T \rightarrow \text{Pre}$ inherits a structure of a $\text{TLC}\beta$ -module, the *fibre module* $[\text{TLC}\beta]_t$ with respect

to $t \in T$. Given $s, t \in T$, the map $V \mapsto \text{TLC}\beta(V)(s \rightsquigarrow t) \times \text{TLC}\beta(V)(s)$ inherits a structure of a $\text{TLC}\beta$ -module. Finally, the constructors of abstraction $\text{Abs } s \ t$ and application $\text{App } s \ t$ are carriers of morphisms of $\text{TLC}\beta$ -modules:

$$\text{Abs}_{s,t} : [\text{TLC}\beta^s]_t \rightarrow [\text{TLC}\beta]_{s \rightsquigarrow t} , \quad \text{App}_{s,t} : [\text{TLC}\beta]_{s \rightsquigarrow t} \times [\text{TLC}\beta]_s \rightarrow [\text{TLC}\beta]_t .$$

Analogous remarks hold for the monad $\text{TLC}\Delta$ and modules over this monad.

As in [Ex. 4](#), we consider a language over a set T of types as a (relative) monad on Δ^T . Translations between languages are given by *colax morphisms of monads*:

Definition 6. *Suppose given two relative monads $P : \mathcal{C} \xrightarrow{F} \mathcal{D}$ and $Q : \mathcal{C}' \xrightarrow{F'} \mathcal{D}'$. A colax morphism of relative monads from P to Q is a quadruple $h = (G, G', N, \tau)$ of a functor $G : \mathcal{C} \rightarrow \mathcal{C}'$, a functor $G' : \mathcal{D} \rightarrow \mathcal{D}'$ as well as a natural transformation $N : F'G \rightarrow G'F$ and a natural transformation $\tau : PG' \rightarrow GQ$ such that the following diagrams commute for any objects c, d and any suitable morphism f :*

$$\begin{array}{ccc} G'Pc & \xrightarrow{G'\sigma^P(f)} & G'Pd \\ \tau_c \downarrow & & \downarrow \tau_d \\ QGc & \xrightarrow{\sigma^Q(\tau_d \circ G'f \circ Nc)} & QGd \end{array} \quad \begin{array}{ccc} F'Gc & \xrightarrow{Nc} & G'Fc & \xrightarrow{G'\eta_c^P} & G'Pc \\ & \searrow \eta_{Gc}^Q & & & \downarrow \tau_c \\ & & & & QGc \end{array}$$

Given a morphism of relative monads $h : P \rightarrow Q$ and a Q -module N with codomain \mathcal{E} , we define the *pullback P -module h^*N* , also with codomain \mathcal{E} :

Definition 7. *We define the pullback of N along h with object map $c \mapsto M(Gc)$ and with substitution map, for $f : Fc \rightarrow Pd$, as $\zeta^{h^*M}(f) := \zeta^M(\tau_d \circ G'f \circ Nc)$. The pullback extends to module morphisms and is functorial.*

Given two languages over different object types T and T' , modelled as relative monads P and Q on Δ^T and $\Delta^{T'}$, respectively, we model a translation from P to Q by a colax monad morphism whose underlying functors are *retyping functors*:

Definition 8 (Retyping Functor). *Let $g : T \rightarrow T'$ be a map of sets, and let \mathcal{C} be a cocomplete category. The map g induces a functor $g^* : \mathcal{C}^{T'} \rightarrow \mathcal{C}^T$ by postcomposition, $W \mapsto W \circ g$. The retyping functor $\bar{g} : \mathcal{C}^T \rightarrow \mathcal{C}^{T'}$ associated to $g : T \rightarrow T'$ is defined as the left Kan extension operation along g , that is, we have an adjunction $\bar{g} \dashv g^*$.*

Remark 9. We are going to use the following instance of [Def. 6](#): P and Q are monads — e.g., languages — on Δ^T and $\Delta^{T'}$, for sets T and T' of object types. The functors G and G' are the retyping functors (cf. [Def. 8](#)) associated to some translation of types $g : T \rightarrow T'$, and N is the identity transformation. Then τ denotes a translation of terms from P to Q :

$$\begin{array}{ccc} \text{Set}^T & \xrightarrow{\Delta^T} & \text{Pre}^T \\ \bar{g} \downarrow & \text{Id} \not\swarrow & \downarrow \bar{g} \\ \text{Set}^{T'} & \xrightarrow{\Delta^{T'}} & \text{Pre}^{T'} \end{array} \quad \begin{array}{ccc} \text{Set}^T & \xrightarrow{P} & \text{Pre}^T \\ \bar{g} \downarrow & \not\swarrow \tau & \downarrow \bar{g} \\ \text{Set}^{T'} & \xrightarrow{Q} & \text{Pre}^{T'} \end{array}$$

A family of constructors, such as the family $(\text{Abs}_{s,t})_{s,t \in T_{\text{TLC}}}$ of [Ex. 5](#), is modelled via a family of module morphisms of suitable domain and codomain. Equivalently, via *uncurrying*, we can consider such a family as *one* module morphism between two suitable modules: intuitively, the idea is to write $\text{Abs}(V, s, t) : \text{TLC}\beta(V^s)(t) \rightarrow \text{TLC}\beta(V)(s \rightsquigarrow t)$ instead of $\text{Abs}_{s,t}(V)$. For this to work, an object of the domain category of the source and target modules of Abs must be of the form (V, s, t) , where V is a context and $s, t \in T_{\text{TLC}}$. More generally:

Definition 10 (Pointed index sets). *Given a category \mathcal{C} , a set T and a natural number n , we denote by \mathcal{C}_n^T the category with, as objects, diagrams of the form $n \xrightarrow{\mathbf{t}} T \xrightarrow{V} \mathcal{C}$, written (V, t_1, \dots, t_n) with $t_i := \mathbf{t}(i)$. A morphism h to another such (W, \mathbf{t}) with the same pointing map \mathbf{t} is given by a morphism $h : V \rightarrow W$ in \mathcal{C}^T .*

Any functor $F : \mathcal{C}^T \rightarrow \mathcal{D}^T$ extends to $F_n : \mathcal{C}_n^T \rightarrow \mathcal{D}_n^T$ via $F_n(V, \mathbf{t}) := (FV, \mathbf{t})$. Also, any relative monad R over F induces a monad R_n over F_n .

Given a map of sets $g : T \rightarrow T'$, by postcomposing the pointing map with g , the retyping functor (cf. [Def. 8](#)) generalizes to the functor

$$\vec{g}(n) : \mathcal{C}_n^T \rightarrow \mathcal{C}_n^{T'} \quad , \quad (V, \mathbf{t}) \mapsto (\vec{g}V, g \circ \mathbf{t}) \quad .$$

The category \mathcal{C}_n^T consists of T^n copies of \mathcal{C}^T , which do not interact. Due to the “markers” (t_1, \dots, t_n) we can act differently on each copy, cf. [Defs. 12](#) and [13](#).

Two important constructions on modules over monads of [2](#), derivation and fibre modules, carry over to modules over monads on Δ^T . Intuitively, derivation corresponds to considering terms in an extended context, whereas the fibre corresponds to picking terms of a specific object type. Since we consider varying sets of types, the object type for context extension and fibre is chosen through a natural transformation, which picks an element of any set.

Given $u \in T$, we denote by $D(u) \in \text{Set}^T$ the context with $D(u)(u) = \{*\}$ and $D(u)(t) = \emptyset$ for $u \neq t$. For a context $V \in \text{Set}^T$ we set $V^{*u} := V + D(u)$.

Given a category \mathcal{C} and $n \in \mathbb{N}$, we denote by \mathcal{TC}_n the category an object of which is a triple (T, V, \mathbf{t}) of a set T , a T -indexed family V of objects of \mathcal{C} and a vector \mathbf{t} of length n of elements of T . Note that for a fixed set T , the category \mathcal{C}_n^T is the fibre over T of the forgetful functor $\mathcal{TU}_n : \mathcal{TC}_n \rightarrow \text{Set}$ which maps an object (T, V, \mathbf{t}) to its indexing set T . Let $1 : \mathcal{TC}_n \rightarrow \text{Set}$ be the constant functor mapping to the singleton set. For a natural transformation $\tau : 1 \Rightarrow \mathcal{TU}_n$, we write $\tau(T, V, \mathbf{t}) := \tau(T, V, \mathbf{t})(*) \in T$, i.e. we omit the argument from the singleton set. Intuitively, such τ picks an element of T of any object $(T, V, \mathbf{t}) \in \mathcal{TC}_n$.

Example 11. For $1 \leq k \leq n$, we denote by $k : 1 \Rightarrow \mathcal{TU}_n : \mathcal{TC}_n \rightarrow \text{Set}$ the natural transformation such that $k(T, V, \mathbf{t})(*) := \mathbf{t}(k)$.

Definition 12 (Context Extension). *Let τ be as above, and let T be a fixed set. Given a monad P on Δ_n^T and a P -module M with codomain \mathcal{E} , we define the derived module of M with respect to τ by setting $M^\tau(V, \mathbf{t}) := M(V^{*\tau(T, V, \mathbf{t})}, \mathbf{t})$.*

Definition 13 (Fibre). Let τ be as in [Def. 12](#). Let P be a monad on Δ_n^T and M be a P -module with codomain category \mathcal{E}_n^T . The fibre module $[M]_\tau$ of M with respect to τ has object map $(V, \mathbf{t}) \mapsto M(V, \mathbf{t})(\tau(T, V, \mathbf{t}))$, that is, the component $\tau(T, V, \mathbf{t})$ of M , forgetting also the pointing map \mathbf{t} .

Example 14 ([Ex. 5](#) cont.). Let $T := T_{\text{TLC}}$. According to [Def. 10](#), we have a relative monad — and its associated tautological module — $\text{TLC}\beta_2$ on the functor $\Delta_2^T : \text{Set}_2^T \rightarrow \text{Pre}_2^T$. Let $i : 1 \Rightarrow \text{TU}_2 : \text{TC}_2 \rightarrow \text{Set}$, for $i = 1, 2$, be a natural transformation as in [Ex. 11](#). Then we have a $\text{TLC}\beta_2$ -module

$$\text{TLC}\beta_2^1 : (V, s, t) \mapsto \text{TLC}\beta_2(V^s, s, t) .$$

We also have a $\text{TLC}\beta_2$ -module

$$[\text{TLC}\beta_2]_2 : (V, s, t) \mapsto \text{TLC}\beta_2(V, s, t)(t) .$$

Again, as in [Ex. 5](#), analogous remarks hold for $\text{TLC}\Delta$.

Remark 15 (Module of Higher Degree corresponds to a Family of Modules). Let T be a set and let R be a monad on the functor Δ^T . Then a module M over the monad R_n corresponds precisely to a family of R -modules $(M_{\mathbf{t}})_{\mathbf{t} \in T^n}$ by (un)currying. Similarly, a morphism $\alpha : M \rightarrow N$ of modules of degree n is equivalent to a family $(\alpha_{\mathbf{t}})_{\mathbf{t} \in T^n}$ of morphisms of modules of degree zero with $\alpha_{\mathbf{t}} : M_{\mathbf{t}} \rightarrow N_{\mathbf{t}}$.

3 Signatures, Representations, Initiality

We combine the techniques of [2](#) and [11](#) in order to obtain an initiality result for simple type systems with reductions on the term level. As an example, we specify, via the iteration principle stemming from the universal property, a semantically faithful translation from PCF with its usual reduction relation to the untyped lambda calculus with beta reduction. Analogously to [11](#), we define a notion of *2-signature* with two levels: a *syntactic* level specifying types and terms of a language, and, on top, a *semantic* level specifying reduction rules on the terms.

The syntactic level itself — given by a *1-signature* (S, Σ) , cf. [Def. 29](#) — specifies the *types* of the language, via an algebraic signature S , as well as terms that are typed over the types specified by S , via a signature Σ over S . In a first result (cf. [Lem. 34](#)) we characterize the language generated by a 1-signature, and equipped with the *equality preorder*, as an initial object of a category of representations. An instance of this theorem is given in [Ex. 22](#), where $\text{TLC}\Delta$, equipped with two module morphisms given by the constructors `Abs` and `App`, is characterized as the initial representation of a suitable 1-signature.

Afterwards we equip 1-signatures with *inequations*, yielding 2-signatures (cf. [Def. 42](#)). We prove an initiality result for those 2-signatures (cf. [Thm. 44](#)), an instance of which characterizes the simply-typed lambda calculus $\text{TLC}\beta$ with beta reduction as initial representation (cf. [Ex. 43](#)).

Signatures for Types. We consider sets of types that are specified by algebraic signatures, which are presented in [2]. We review briefly:

Definition 16 (Algebraic Signature). *An algebraic signature is given by a family of natural numbers.*

Intuitively, each natural number of such a family specifies the number of arguments of its associated type constructor.

Example 17. The types of the simply-typed lambda calculus are specified via the algebraic signature $S_{\text{TLC}} := \{* : 0, (\rightsquigarrow) : 2\}$.

Example 18. The language PCF [14] is a simply-typed lambda calculus with a fixed point operator and arithmetic constants. The signature of the types of PCF is given by $S_{\text{PCF}} := \{\iota : 0, o : 0, (\Rightarrow) : 2\}$. A representation T of S_{PCF} is given by a set T and three operations of suitable arities. A morphism of representations is a map of sets compatible with the operations on either side.

Lemma 19. *Let S be an algebraic signature. The category of representations of S has an initial object \hat{S} .*

Signatures for Terms. For the rest of the section, let S be a signature for types. Signatures for *terms over S* are *syntactically* defined as in [2]. We call *degree* of an arity the number of object type variables appearing in the arity. For instance, the signature Σ_{TLC} of simply-typed lambda terms over the signature S_{TLC} (cf. Ex. 17) is given by two arities of degree 2:

$$\Sigma_{\text{TLC}} := \{\text{abs} : [([1], 2)] \rightarrow (1 \rightsquigarrow 2), \text{ app} : [([1], 1 \rightsquigarrow 2), ([1], 1)] \rightarrow 2\}. \quad (1)$$

Intuitively, the numbers vary over object types. More precisely, for any representation of S_{TLC} in a set T , the numbers vary over elements of T .

In order to define *representations* of such a signature (S, Σ) , we need to consider set families where the indexing set is equipped with a representation of the type signature S :

Definition 20. *Given a category \mathcal{C} — e.g., the category Set of sets — we define the category SC_n to be the category an object of which is a triple (T, V, \mathbf{t}) where T is a representation of S , the object $V \in \mathcal{C}^T$ is a T -indexed family of objects of \mathcal{C} and \mathbf{t} is a vector of elements of T of length n . We denote by $SU_n : SC_n \rightarrow \text{Set}$ the functor mapping an object (T, V, \mathbf{t}) to the underlying set T .*

We have a forgetful functor $SC_n \rightarrow \mathcal{TC}_n$ which forgets the representation structure. On the other hand, any representation T of S in a set T gives rise to a functor $\mathcal{C}_n^T \rightarrow SC_n$, which “attaches” the representation structure.

Recall from [2] that $S(n)$ denotes terms of S with free variables in $\{1, \dots, n\}$. The meaning of a term $s \in S(n)$ as a natural transformation $s : 1 \Rightarrow SU_n : SC_n \rightarrow \text{Set}$ is given by recursion on the structure of s :

Definition 21 (Canonical Natural Transformation). *Let $s \in S(n)$ be a type of degree n . Then s denotes a natural transformation $s : 1 \Rightarrow SU_n : SC_n \rightarrow \mathbf{Set}$ defined recursively on the structure of s as follows: for $s = \alpha(a_1, \dots, a_k)$ the image of a constructor $\alpha \in S$ we set*

$$s(T, V, \mathbf{t}) = \alpha(a_1(T, V, \mathbf{t}), \dots, a_k(T, V, \mathbf{t}))$$

and for $s = m$ with $1 \leq m \leq n$ we define $s(T, V, \mathbf{t}) = \mathbf{t}(m)$. We call a natural transformation of the form $s \in S(n)$ canonical.

The natural transformations of [Ex. 11](#) yield examples of canonical transformations. We now define representations of the 1–signature $(S_{\text{TLC}}, \Sigma_{\text{TLC}})$ of the simply–typed lambda calculus. Afterwards we define general 1–signatures and their representations.

Example 22 (Ex. 14 cont.). Let $S := S_{\text{TLC}}$ be the signature for types of TLC as in [Ex. 17](#). We denote by $i : 1 \Rightarrow SU_2 : SC_2 \rightarrow \mathbf{Set}$, for $i = 1, 2$, the natural transformations defined analogously to those of [Ex. 14](#). We define the transformation $1 \rightsquigarrow 2 : 1 \Rightarrow SU_2$ as

$$(1 \rightsquigarrow 2)(V, s, t)(*) := s \rightsquigarrow t .$$

The constructors of the simply–typed lambda calculus thus constitute the carriers of two module morphisms,

$$\begin{aligned} \text{Abs} &: [\text{TLC}\Delta_2^1]_2 \rightarrow [\text{TLC}\Delta_2]_{1 \rightsquigarrow 2} \\ \text{App} &: [\text{TLC}\Delta_2]_{1 \rightsquigarrow 2} \times [\text{TLC}\Delta_2]_1 \rightarrow [\text{TLC}\Delta_2]_2 . \end{aligned} \quad (2)$$

Altogether we model the simply–typed lambda calculus with equality relation via the following categorical structure:

- the relative monad $\text{TLC}\Delta$ on Δ^{TLC} and
- two morphisms of $\text{TLC}\Delta_2$ –modules Abs and App of type as in [Disp. \(2\)](#).

We thus define a *representation* of the simply–typed lambda calculus, specified by the signature $(S_{\text{TLC}}, \Sigma_{\text{TLC}})$ (cf. [Disp. \(1\)](#)), as a representation T of S_{TLC} in a set T , a monad P on Δ^T and two morphisms of P_2 –modules

$$\text{Abs} : [P_2^1]_2 \rightarrow [P_2]_{1 \rightsquigarrow 2} \quad \text{and} \quad \text{App} : [P_2]_{1 \rightsquigarrow 2} \times [P_2]_1 \rightarrow [P_2]_2 .$$

Together with a suitable definition of *morphisms of representations*, this yields a category in which the triple $(\text{TLC}\Delta, \text{Abs}, \text{App})$ is the initial object.

In general, an arity over S of degree $n \in \mathbb{N}$ is given by a pair of functors, each of which associates, to any suitable monad P , a source $\text{dom}(s, P)$ and a target $\text{cod}(s, P)$ of a P_n –module morphism. Each such functor is called a *half–arity*. Representing an arity in the monad P then means specifying a module morphism $\text{dom}(s, P) \rightarrow \text{cod}(s, P)$.

We define the source and target categories of half–arities; an object of the source category is a pair of a representation of S in a set T and a monad on Δ^T .

Definition 23 (Relative S -Monad). *The category $S\text{-RMnd}$ of relative S -monads is the category whose objects are pairs (T, P) of a representation T of S and a relative monad P on Δ^T . A morphism from (T, P) to (T', P') is a pair (g, f) of a morphism of S -representations $g : T \rightarrow T'$ and a morphism of relative monads $f : P \rightarrow P'$ over \vec{g} as in [Rem. 9](#).*

Given $n \in \mathbb{N}$, we write $S\text{-RMnd}_n$ for the category whose objects are pairs (T, P) of a representation T of S and a relative monad P over Δ_n^T . A morphism from (T, P) to (T', P') is a pair (g, f) of a morphism of S -representations $g : T \rightarrow T'$ and a monad morphism $f : P \rightarrow P'$ over the retyping functor $\vec{g}(n)$ ([Def. 10](#)).

The target categories mix modules over different relative monads:

Definition 24. *Given $n \in \mathbb{N}$, an algebraic signature S and a category \mathcal{D} , we call $\text{LRMod}_n(S, \mathcal{D})$ the category an object of which is a pair (P, M) of a relative S -monad $P \in S\text{-RMnd}_n$ and a P -module with codomain \mathcal{D} . A morphism to another such (Q, N) is a pair (f, h) of a morphism of relative S -monads $f : P \rightarrow Q$ in $S\text{-RMnd}_n$ and a morphism of relative modules $h : M \rightarrow f^*N$.*

We sometimes just write the module — i.e. the second — component of an object or morphism of the large category of modules. Given $M \in \text{LRMod}_n(S, \mathcal{D})$, we thus write $M(V, \mathbf{t})$ or $M_{V, \mathbf{t}}$ for the value of the module on the object (V, \mathbf{t}) .

A *half-arity over S of degree n* is a functor from relative S -monads to the category of large modules of degree n .

Definition 25. *Given an algebraic signature S and $n \in \mathbb{N}$, a half-arity over S of degree n is a functor $\alpha : S\text{-RMnd} \rightarrow \text{LRMod}_n(S, \text{Pre})$ which is pre-inverse to the forgetful functor.*

The basic building brick for the half-arithies we consider is the *tautological* module:

Definition 26. *To any relative S -monad R we associate the tautological module of R_n (cf. [Def. 10](#)),*

$$\Theta_n(R) := (R_n, R_n) \in \text{LRMod}_n(S, \text{Pre}_n^T) .$$

From the tautological module, we build *classic* half-arithies using canonical natural transformations (cf. [Def. 21](#)); these transformations specify context extension (derivation) and selection of specific object types (fibre):

Definition 27 (Classic Half-Arity). *The following clauses define an inductive set of classic half-arithies, to which we restrict our attention:*

- The constant functor $*$: $R \mapsto \mathbf{1}_{\text{RMnd}(R, \text{Pre})}$ is a classic half-arity.
- Given any canonical natural transformation $\tau : \mathbf{1} \Rightarrow \text{SU}_n$ (cf. [Def. 21](#)), the point-wise fibre module with respect to τ (cf. [Def. 13](#)) of the tautological module $\Theta_n : R \mapsto (R_n, R_n)$ (cf. [Def. 26](#)) is a classic half-arity of degree n ,

$$[\Theta_n]_\tau : S\text{-RMnd} \rightarrow \text{LRMod}_n(S, \text{Pre}) , \quad R \mapsto [R_n]_\tau .$$

- Given any (classic) half-arity $M : S\text{-Mnd} \rightarrow \text{LMod}_n(S, \text{Pre})$ of degree n and a canonical natural transformation $\tau : 1 \Rightarrow SU_n$, the point-wise derivation of M with respect to τ is a (classic) half-arity of degree n ,

$$M^\tau : S\text{-RMnd} \rightarrow \text{LRMod}_n(S, \text{Pre}) \quad , \quad R \mapsto (M(R))^\tau \quad .$$

- Given two (classic) half-arithies M and N of degree n , their pointwise product of modules $M \times N$ is again a (classic) half-arity of degree n .

A half-arity of degree n thus associates, to any relative S -monad P over a set of types T , a family of P -modules indexed by T^n , cf. [Rem. 15](#).

An arity of degree $n \in \mathbb{N}$ for terms over an algebraic signature S is defined to be a pair of functors from relative S -monads to modules in $\text{LRMod}_n(S, \text{Pre})$. The degree n corresponds to the number of object type indices of its associated constructor. For instance, the arities of `Abs` and `App` of [Disp. \(1\)](#) are of degree 2.

Definition 28 (Term-Arity, Signature over S). A classic arity α over S of degree n is a pair $s = (\text{dom}(\alpha), \text{cod}(\alpha))$ of half-arithies over S of degree n such that $\text{dom}(\alpha)$ is classic and $\text{cod}(\alpha)$ is of the form $[\Theta_n]_\tau$ for some canonical transformation τ as in [Def. 21](#). We write $\text{dom}(\alpha) \rightarrow \text{cod}(\alpha)$ for the arity α , and $\text{dom}(\alpha, R) := \text{dom}(\alpha)(R)$ and similar for the codomain and morphisms of relative S -monads. A term-signature Σ over S is a family of classic arities (of varying degree) over S .

Definition 29 (1-Signature). A 1-signature is a pair (S, Σ) consisting of an algebraic signature S for sorts and a term-signature Σ over S .

Example 30 ([Ex. 22](#) cont.). The terms of the simply typed lambda calculus over the type signature of [Ex. 17](#) are given by the arities

$$\text{abs} : [\Theta]_2^1 \rightarrow [\Theta]_{1 \rightsquigarrow 2} \quad , \quad \text{app} : [\Theta]_{1 \rightsquigarrow 2} \times [\Theta]_1 \rightarrow [\Theta]_2 \quad ,$$

both of which are of degree 2 — we leave the degree implicit. The outer lower index and the exponent are to be interpreted as de Bruijn variables, ranging over types. They indicate the fibre (cf. [Def. 13](#)) and derivation (cf. [Def. 12](#)), respectively, in the special case where the corresponding natural transformation is given by a natural number as in [Def. 21](#).

Example 31 ([Ex. 18](#) cont.). The term-signature of PCF consists of an arity for abstraction and an arity for application, each of degree 2, an arity (of degree 1) for the fixed point operator, and one arity of degree 0 for each logic and arithmetic constant — some of which we omit:

$$\begin{aligned} \text{abs} : [\Theta]_2^1 \rightarrow [\Theta]_{1 \Rightarrow 2} \quad , \quad \text{app} : [\Theta]_{1 \Rightarrow 2} \times [\Theta]_1 \rightarrow [\Theta]_2 \quad , \quad \mathbf{Fix} : [\Theta]_{1 \Rightarrow 1} \rightarrow [\Theta]_1 \quad , \\ \mathbf{n} : * \rightarrow [\Theta]_\iota \quad \text{for } n \in \mathbb{N} \quad , \quad \mathbf{Succ} : * \rightarrow [\Theta]_{\iota \Rightarrow \iota} \quad , \quad \mathbf{Zero}? : * \rightarrow [\Theta]_{\iota \Rightarrow o} \end{aligned}$$

Definition 32 (Representation of an Arity, of a 1-Signature over S). A representation r of an arity α over S in an S -monad R is a morphism of relative modules $r : \text{dom}(\alpha, R) \rightarrow \text{cod}(\alpha, R)$. A representation R of a signature over S is given by a relative S -monad — called R as well — and a representation α^R of each arity α of S in R .

Representations of (S, Σ) are the objects of a category $\text{Rep}^\Delta(S, \Sigma)$, whose morphisms are defined as follows:

Definition 33 (Morphism of Representations). *Given representations P and R of a typed signature (S, Σ) , a morphism of representations $f : P \rightarrow R$ is given by a morphism of relative S -monads $f : P \rightarrow R$, such that for any arity α of Σ the following diagram of module morphisms commutes:*

$$\text{cod}(\alpha, f) \circ \alpha^P = \alpha^R \circ \text{dom}(\alpha, f) .$$

Lemma 34. *For any 1-signature (S, Σ) , the category of representations of (S, Σ) has an initial object.*

Proof. The initial object is obtained, analogously to the untyped case (cf. [1]), via an adjunction $\Delta_* \dashv U_*$ between the categories of representations of (S, Σ) in relative monads and those in monads as in [2]. We use that left adjoints are cocontinuous, and thus preserve initial objects.

Inequations & 2-Signatures. An inequation associates, to any representation of (S, Σ) in a relative monad P , two parallel morphisms of P -modules. Similarly to arities, an inequation (of higher degree) may be given by a *family of inequations*, indexed by object types. Consider the simply-typed lambda calculus, which was defined with *typed* abstraction and application. Similarly, we have a *typed substitution* operation for TLC and, more generally, for any monad on Δ^T (cf. [Def. 36]). For $s, t \in T_{\text{TLC}}$ and $M \in \text{TLC}(V^{*s})_t$ and $N \in \text{TLC}(V)_s$, beta reduction is specified by

$$\lambda M(N) \rightsquigarrow M[* := N] ,$$

where our notation hides the fact that abstraction, application and substitution are typed operations. More formally, such a reduction rule might read as a family of inequations between morphisms of modules

$$\text{app}_{s,t} \circ (\text{abs}_{s,t} \times \text{id}) \leq _ [*^s :=_t _] ,$$

where $s, t \in T_{\text{TLC}}$ range over types of the simply-typed lambda calculus. Analogously to 1-signatures, we want to specify the beta rule without referring to the set T_{TLC} , but instead express it for an arbitrary representation R of the typed signature $(S_{\text{TLC}}, \Sigma_{\text{TLC}})$ (cf. [Ex. 30]), as in

$$\text{app}^R \circ (\text{abs}^R \times \text{id}) \leq _ [* := _] ,$$

where both the left and the right side of the inequation are given by suitable R -module morphisms of degree 2.

Definition 35. *Let (S, Σ) be a 1-signature, and let $U : \text{Rep}^\Delta(S, \Sigma) \rightarrow S\text{-RMnd}$ be the forgetful functor. Given two (classic) half-arities $\text{dom}(s)$ and $\text{cod}(s)$ of degree $n \in \mathbb{N}$, a half-equation $\alpha : \text{dom}(s) \rightarrow \text{cod}(s)$ of degree n over (S, Σ) is a natural transformation $\alpha : \text{dom}(s) \circ U \rightarrow \text{cod}(s) \circ U$. We call an inequation classic when its codomain is given by a classic half-arity.*

Definition 36 (Substitution of one Variable as a Half–Equation). Let T be a (nonempty) set and let P be a monad over Δ^T . For any $s, t \in T$ and $X \in \mathbf{Set}^T$ we define a binary substitution operation $(y, z) \mapsto y[* := z] := \sigma([\eta_X, x \mapsto z])(y)$. For any pair $(s, t) \in T^2$, we thus obtain a morphism of P –modules

$$\text{subst}_{s,t}^P : [P^s]_t \times [P]_s \rightarrow [P]_t .$$

By [Rem. 15](#) this family is equivalent to a module morphism of degree 2. We thus have a half–equation of degree 2 with classic domain and codomain over any typed signature,

$$\text{subst} : R \mapsto \text{subst}^R : [R_2^1]_2 \times [R_2]_1 \rightarrow [R_2]_2 .$$

Example 37 ([Ex. 30](#) cont.). The following map yields a half–equation over the signature TLC, as well as over the signature of PCF:

$$\text{app} \circ (\text{abs} \times \text{id}) : R \mapsto \text{app}^R \circ (\text{abs}^R \times \text{id}^R) : [R_2^1]_2 \times [R_2]_1 \rightarrow [R_2]_2 .$$

Definition 38 (Inequation). Given a signature (S, Σ) , an inequation over (S, Σ) , or (S, Σ) –inequation, of degree $n \in \mathbb{N}$ is a pair of parallel half–equations of degree n . We write $\alpha \leq \gamma$ for the inequation (α, γ) .

Example 39 (Beta Reduction). For any suitable 1–signature — i.e. for any 1–signature that has an arity for abstraction and an arity for application — we specify beta reduction using the parallel half–equations of [Def. 36](#) and [Ex. 37](#):

$$\text{app} \circ (\text{abs} \times \text{id}) \leq \text{subst} : [\Theta_2]_2^1 \times [\Theta_2]_1 \rightarrow [\Theta_2]_2 .$$

Example 40 (Fixpoints and Arithmetics of PCF). We specify some of the reduction rules of PCF via inequations over the 1–signature of PCF (cf. [Ex. 31](#)); for space reasons we refrain from specifying all of them. The reader may fill in the missing inequations, whose informal specification can be found, e.g., in [\[13\]](#).

$$\begin{aligned} \mathbf{Fix} &\leq \text{app} \circ (\text{id}, \mathbf{Fix}) : [\Theta]_{1 \Rightarrow 1} \rightarrow [\Theta]_1 \\ \text{app} \circ (\mathbf{Pred}, \mathbf{0}) &\leq \mathbf{0} : * \rightarrow [\Theta]_i \\ \text{app} \circ (\mathbf{Pred}, \text{app} \circ (\mathbf{Succ}, \mathbf{n})) &\leq \mathbf{n} : * \rightarrow [\Theta]_i \\ \text{app} \circ (\mathbf{Zero}?, \mathbf{0}) &\leq \mathbf{T} : * \rightarrow [\Theta]_o \\ \text{app} \circ (\mathbf{Zero}?, \text{app} \circ (\mathbf{Succ}, \mathbf{n})) &\leq \mathbf{F} : * \rightarrow [\Theta]_o \end{aligned}$$

Definition 41 (Representation of Inequations). A representation of an (S, Σ) –inequation $\alpha \leq \gamma : U \rightarrow V$ (of degree n) is any representation R over a set of types T of (S, Σ) such that $\alpha^R \leq \gamma^R$ pointwise, i.e. if for any pointed context $(X, \mathbf{t}) \in \mathbf{Set}^T \times T^n$, any $t \in T$ and any $y \in U_{(X, \mathbf{t})}^R(t)$, $\alpha^R(y) \leq \gamma^R(y)$, where we omit the sort argument t as well as the context (X, \mathbf{t}) from α and γ . We say that such a representation R satisfies the inequation $\alpha \leq \gamma$.

The category of representations of $((S, \Sigma), A)$ is defined as the full subcategory of $\text{Rep}^\Delta(S, \Sigma)$ of representations satisfying each inequation of A . According to [Rem. 15](#), the above inequation is equivalent to ask whether, for any $\mathbf{t} \in T^n$, any $t \in T$ and any $y \in U_{\mathbf{t}}^R(X)(t)$, $\alpha_{\mathbf{t}}^R(y) \leq \gamma_{\mathbf{t}}^R(y)$.

Definition 42 (2–Signature). A 2–signature is a pair given by a 1–signature (S, Σ) and a set A of classic inequations over (S, Σ) .

Example 43 (Representations of TLC with β). A representation of $(S_{\text{TLC}}, \Sigma_{\text{TLC}}, \beta)$ is given by a representation $(P, \text{Abs}, \text{App})$ of $(S_{\text{TLC}}, \Sigma_{\text{TLC}})$ over a set T “of types” such that, for any context $V \in \text{Set}^T$, any $s, t \in T$ and any $M \in P^s(V)(t)$ and $N \in P(V)(s)$,

$$\text{App}_{s,t}(\text{Abs}_{s,t}(M), N) \leq M[* := N] .$$

The initial such representation is given by the triple $(\text{TLC}\beta, \text{Abs}, \text{App})$, where

$$\begin{aligned} \text{Abs} &: [\text{TLC}\beta_2^1]_2 \rightarrow [\text{TLC}\beta_2]_{1 \rightsquigarrow 2} \\ \text{App} &: [\text{TLC}\beta_2]_{1 \rightsquigarrow 2} \times [\text{TLC}\beta_2]_1 \rightarrow [\text{TLC}\beta_2]_2 . \end{aligned}$$

The above example is an instance of the following general theorem for 2–signatures:

Theorem 44. For any set of classic (S, Σ) –inequations A , the category of representations of $((S, \Sigma), A)$ has an initial object.

A proof of the theorem can be found in the author’s PhD thesis [3].

The following remark gives a “manual” on how to use the universal property of initiality in order to specify a translation between two languages:

Remark 45 (Iteration Principle by Initiality). The universal property of the language generated by a 2–signature yields an *iteration principle* to define maps — translations — on this language, which are compatible by construction with substitution and reduction in the source and target languages. A translation from the language generated by (S, Σ, A) to the language generated by (S', Σ', A') can be obtained, via the universal property, as an initial morphism in $\text{Rep}^\Delta(S, \Sigma, A)$, obtained by equipping the relative monad $\hat{\Sigma}'_{A'}$ underlying the target language with a representation of the signature (S, Σ, A) . In more detail:

1. we give a representation of the type signature S in the set \hat{S}' . By initiality of \hat{S} , this yields a translation $\hat{S} \rightarrow \hat{S}'$ of sorts.
2. Afterwards, we specify a representation of the term signature Σ in the monad $\hat{\Sigma}'_{A'}$ by defining suitable (families) of morphisms of $\hat{\Sigma}'_{A'}$ –modules. This yields a representation R of (S, Σ) in the monad $\hat{\Sigma}'_{A'}$.
3. Finally, we verify that the representation R of (S, Σ) satisfies the inequations of A , that is, we check whether, for each $\alpha \leq \gamma : \text{dom}(\alpha) \rightarrow \text{cod}(\alpha) \in A$, and for each context V , each $t \in \hat{S}$ and $x \in \text{dom}(\alpha)_V^R(t)$, $\alpha^R(x) \leq \gamma^R(x)$.

Example 46 (Translation from PCF to ULC). We use the aforementioned iteration principle to specify a translation from PCF to ULC, which is semantically faithful with respect to the usual reduction relation of PCF — generated by the inequations of Ex. 40 (and some more, see [14]) — and beta reduction of ULC. For space reasons, we cannot present this example here; we refer to [3]. This example — initiality of the types and terms of PCF with its reductions, and a translation to ULC with beta reduction via associated category–theoretic iteration operator — has also been implemented in the proof assistant Coq. The source files and documentation are available on <http://math.unice.fr/laboratoire/logiciels>.

References

1. Ahrens, B.: Modules over relative monads for syntax and semantics (2011), <http://arxiv.org/abs/1107.5252> (to be published in Math. Struct. in Comp. Science)
2. Ahrens, B.: Extended Initiality for Typed Abstract Syntax. Logical Methods in Computer Science 8(2), 1–35 (2012)
3. Ahrens, B.: Initiality for Typed Syntax and Semantics. Ph.D. thesis, University of Nice Sophia Antipolis, France (2012), <http://benedikt-ahrens.org>
4. Altenkirch, T., Chapman, J., Uustalu, T.: Monads Need Not Be Endofunctors. In: Ong, L. (ed.) FOSSACS 2010. LNCS, vol. 6014, pp. 297–311. Springer, Heidelberg (2010)
5. Coq: The Coq Proof Assistant (2010), <http://coq.inria.fr>
6. Fernández, M., Gabbay, M.J.: Nominal rewriting. Information and Computation 205(6), 917–965 (2007)
7. Fiore, M.P., Hur, C.-K.: Equational Systems and Free Constructions (EXTENDED ABSTRACT). In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 607–618. Springer, Heidelberg (2007)
8. Ghani, N., Lüth, C.: Rewriting via coinserters. Nord. J. Comput. 10(4), 290–312 (2003)
9. Hirschowitz, A., Maggesi, M.: Modules over Monads and Linearity. In: Leivant, D., de Queiroz, R. (eds.) WoLLIC 2007. LNCS, vol. 4576, pp. 218–237. Springer, Heidelberg (2007)
10. Hirschowitz, A., Maggesi, M.: The algebraicity of the lambda-calculus. CoRR abs/0704.2900 (2007), <http://arxiv.org/abs/0704.2900>
11. Hirschowitz, A., Maggesi, M.: Modules over monads and initial semantics. Inf. Comput. 208(5), 545–564 (2010)
12. Hur, C.K.: Categorical equational systems: algebraic models and equational reasoning. Ph.D. thesis, University of Cambridge, UK (2010)
13. Hyland, J.M.E., Ong, C.H.: On full abstraction for PCF. Information and Computation 163, 285–408 (2000)
14. Plotkin, G.D.: LCF considered as a programming language. Theoretical Computer Science 5(3), 223–255 (1977)

Moving Arrows and Four Model Checking Results

Carlos Areces^{1,2}, Raul Fervari¹, and Guillaume Hoffmann¹

¹ FaMAF, Universidad Nacional de Córdoba, Argentina
{careces, fervari, hoffmann}@famaf.unc.edu.ar

² CONICET, Argentina

Abstract. We study dynamic modal operators that can change the model during the evaluation of a formula. In particular, we extend the basic modal language with modalities that are able to swap, delete or add pairs of related elements of the domain, while traversing an edge of the accessibility relation. We study these languages together with the sabotage modal logic, which can arbitrarily delete edges of the model. We define a suitable notion of bisimulation for the basic modal logic extended with each of the new dynamic operators and investigate their expressive power, showing that they are all incomparable. We also show that the complexity of their model checking problems is PSpace-complete.

1 Introduction

Modal logics [24] are particularly well suited to *describe* graphs, and this is fortunate as many situations can be modeled using graphs: an algebra, a database, the execution flow of a program or, simply, the arbitrary relations between a set of elements. This explains why modal logics have been used in many, diverse fields. They offer a well balanced trade-off between expressivity and computational complexity (model checking the basic modal language \mathcal{ML} is only polynomial, while its satisfiability problem is PSpace-complete). Moreover, the range of modal logics known today is extremely wide, so that it is usually possible to pick and choose the right modal logic for a particular application.

But if we want to describe and reason about *dynamic aspects* of a given situation, e.g., how the relations between a set of elements *evolve* through time or through the application of certain operations, the use of modal logics (or actually, any kind of logic with classical semantics) becomes less clear. We can always resort to modeling the whole space of possible evolutions of the system as a graph, but this soon becomes unwieldy. It would be more elegant to use truly dynamic modal logics with operators that can mimic the changes that structure will undergo. This is not a new idea, and a clear example of this kind of logics is the *sabotage logic* introduced by Johan van Benthem in [12].

Consider the following *sabotage game*. It is played on a graph with two players, Runner and Blocker. Runner can move on the graph from node to accessible node, starting from a designated point, and with the goal of reaching a given final point. He should move one edge at a time. Blocker, on the other hand, can

delete one edge from the graph, every time it is his turn. Of course, Runner wins if he manages to move from the origin to the final point in the graph, while Blocker wins otherwise. van Benthem discusses in [12] how to transform the sabotage game into a modal logic. van Benthem’s original idea has been studied in several other works [6,10] where the sabotage operator is defined as:

$$\mathcal{M}, w \models \langle gs \rangle \varphi \text{ iff there is a pair } (u, v) \text{ of } \mathcal{M} \text{ such that } \mathcal{M}_{\{(u,v)\}}^-, w \models \varphi,$$

where $\mathcal{M}_{\{(u,v)\}}^-$ is identical to \mathcal{M} except that the edge (u, v) has been removed from the accessibility relation.

It is clear that the $\langle gs \rangle$ operator *changes* the model in which a formula is evaluated. As van Benthem puts it, $\langle gs \rangle$ is an “external” modality that takes evaluation to another model, obtained from the current one by deleting some transition. It has been proved that solving the sabotage game is PSpace-hard, while the model checking problem of the associated modal logic is PSpace-complete and the satisfiability problem is undecidable. The logic fails to have both the finite model property and the tree model property [6,10].

In this article, we will investigate various model changing operators. The first one, $\langle sw \rangle$, has the ability to swap the direction of a traversed arrow. The $\langle sw \rangle$ operator is a \diamond operator — to be true at a state w it requires the existence of an accessible state v where evaluation will continue— but it changes the accessibility relation during evaluation — the pair (w, v) is deleted, and the pair (v, w) added to the accessibility relation. A picture will help understand the dynamics of $\langle sw \rangle$. The formula $\langle sw \rangle \diamond \top$ is true in a model with two related states:



As we can see in the picture, evaluation starts at state w with the arrow pointing from w to v , but after evaluating the $\langle sw \rangle$ operator, it continues at state v with the arrow now pointing from v to w . We will investigate two other dynamic operators in this article. $\langle ls \rangle$, for *local sabotage*, is a \diamond operator that destroys the traversed arrow, while $\langle br \rangle$, for *bridge*, models the opposite situation: it adds an arrow to an inaccessible point of the model and moves over there.

We have chosen these model changing operators with the intention of covering a sufficiently varied sample of alternatives. The goal is to investigate whether the differences among them lead to different properties of the logics they defined, and how they vary in expressive power. Clearly, other operators could have been included in this exploration, and actually some alternative choices have been already investigated in the literature, e.g., the adjacent sabotage operator discussed in [10].

Summing up then, we will study and compare the expressive powers of $\mathcal{ML}(\langle sw \rangle)$, $\mathcal{ML}(\langle gs \rangle)$, $\mathcal{ML}(\langle ls \rangle)$ and $\mathcal{ML}(\langle br \rangle)$, and we provide complexity results for their model checking problems.

2 Syntax and Semantics

The syntax of the dynamic modal logics we will study is a straightforward extension of the basic modal logic (see [2]):

Definition 1 (Syntax). Let PROP be a countable, infinite set of propositional symbols. Then the set FORM of formulas over PROP is defined as:

$$\text{FORM} ::= \perp \mid p \mid \neg\varphi \mid \varphi \wedge \psi \mid \blacklozenge\varphi,$$

where $p \in \text{PROP}$, $\blacklozenge \in \{\diamond, \langle sw \rangle, \langle gs \rangle, \langle ls \rangle, \langle br \rangle\}$ and $\varphi, \psi \in \text{FORM}$. Other operators are defined as usual. In particular, $\blacksquare\varphi$ is defined as $\neg\blacklozenge\neg\varphi$.

Formulas of the basic modal language \mathcal{ML} are those that contains only the \diamond operator beside the Boolean operators. We call $\mathcal{ML}(\blacklozenge)$ to the extension of \mathcal{ML} allowing also the \blacklozenge operator, for $\blacklozenge \in \{\langle sw \rangle, \langle gs \rangle, \langle ls \rangle, \langle br \rangle\}$.

Semantically, formulas of $\mathcal{ML}(\langle sw \rangle)$, $\mathcal{ML}(\langle gs \rangle)$, $\mathcal{ML}(\langle ls \rangle)$ and $\mathcal{ML}(\langle br \rangle)$ are evaluated in standard relational models, and the meaning of all the operators of the basic modal logic is unchanged. When we evaluate formulas containing dynamic operators, we will need to keep track of the edges that have been modified. To that end, let us define precisely the models that we will use. In the rest of this article we will use wv as a shorthand for $\{(w, v)\}$ or (w, v) . Context will always disambiguate the intended use.

Definition 2 (Models and Model Updates). A model \mathcal{M} is a triple $\mathcal{M} = \langle W, R, V \rangle$, where W is a non-empty set whose elements are called points or states; $R \subseteq W \times W$ is the accessibility relation; and $V : \text{PROP} \mapsto \mathcal{P}(W)$ is a valuation.

Given a model $\mathcal{M} = \langle W, R, V \rangle$, we define the following notations:

- (*swapping*) $\mathcal{M}_{vw}^* = \langle W, R_{vw}^*, V \rangle$, with $R_{vw}^* = (R \setminus vw) \cup vw$, $wv \in R$.
- (*sabotaging*) $\mathcal{M}_{vw}^- = \langle W, R_{vw}^-, V \rangle$, with $R_{vw}^- = R \setminus vw$, $wv \in R$.
- (*bridging*) $\mathcal{M}_{vw}^+ = \langle W, R_{vw}^+, V \rangle$, with $R_{vw}^+ = R \cup vw$, $wv \in (W \times W) \setminus R$.

Let w be a state in \mathcal{M} , the pair (\mathcal{M}, w) is called a pointed model; we will usually drop parenthesis and call \mathcal{M}, w a pointed model.

We are now ready to introduce the semantics.

Definition 3 (Semantics). Given a pointed model \mathcal{M}, w and a formula φ we say that \mathcal{M}, w satisfies φ , and write $\mathcal{M}, w \models \varphi$, when

$$\begin{aligned} \mathcal{M}, w \models p & \quad \text{iff } w \in V(p) \\ \mathcal{M}, w \models \neg\varphi & \quad \text{iff } \mathcal{M}, w \not\models \varphi \\ \mathcal{M}, w \models \varphi \wedge \psi & \quad \text{iff } \mathcal{M}, w \models \varphi \text{ and } \mathcal{M}, w \models \psi \\ \mathcal{M}, w \models \diamond\varphi & \quad \text{iff for some } v \in W \text{ s.t. } wRv, \mathcal{M}, v \models \varphi \\ \mathcal{M}, w \models \langle sw \rangle\varphi & \quad \text{iff for some } v \in W \text{ s.t. } wRv, \mathcal{M}_{vw}^*, v \models \varphi \\ \mathcal{M}, w \models \langle gs \rangle\varphi & \quad \text{iff for some } v, u \in W, \text{ s.t. } vRu, \mathcal{M}_{vu}^-, w \models \varphi \\ \mathcal{M}, w \models \langle ls \rangle\varphi & \quad \text{iff for some } v \in W \text{ s.t. } wRv, \mathcal{M}_{wv}^-, v \models \varphi \\ \mathcal{M}, w \models \langle br \rangle\varphi & \quad \text{iff for some } v \in W \text{ s.t. } \neg wRv, \mathcal{M}_{wv}^+, v \models \varphi. \end{aligned}$$

φ is satisfiable if for some pointed model \mathcal{M}, w we have $\mathcal{M}, w \models \varphi$.

We write $\mathcal{M}, w \equiv_{\mathfrak{L}} \mathcal{N}, v$ when both models satisfy the same \mathfrak{L} -formulas, i.e., for all $\varphi \in \mathfrak{L}$, $\mathcal{M}, w \models \varphi$ if and only if $\mathcal{N}, v \models \varphi$. We will drop the \mathfrak{L} subindex when no confusion arises.

Once syntax and semantics are in place, the following result that distinguishes the dynamic logics from \mathcal{ML} can be easily established. A basic result for \mathcal{ML} shows that it has the *tree model property*: every satisfiable formula of \mathcal{ML} can be satisfied at the root of a model where the accessibility relation defines a tree (i.e., there is a root, the relation is irreflexive, all elements different from the root can be reached from the root via the transitive closure of the accessibility relation, and no element has two different predecessors).

Theorem 4. $\mathcal{ML}(\diamond)$ does not have the tree model property, for $\diamond \in \{\langle sw \rangle, \langle gs \rangle, \langle ls \rangle, \langle br \rangle\}$

Proof. For details see the appendix. We present formulas that ensure that the accessibility relation does not define a tree. The $\langle gs \rangle$ case has already been proved in [6]. Suppose the following formulas hold at some point w :

1. $p \wedge (\bigwedge_{1 \leq i < 3} \square^i \neg p) \wedge \langle sw \rangle \diamond \diamond p$, then w has a reflexive successor;
2. $\diamond \diamond \top \wedge \neg [ls] \square \perp$, then w is reflexive;
3. $\diamond \diamond \top \wedge [gs] \square \perp$, then w is reflexive;
4. $\langle br \rangle \square \perp$, then w and some different point v are unconnected.

In each case, the formula cannot be satisfied in a tree. \square

As the four logics we introduced are conservative extensions of \mathcal{ML} , the formulas above show that each is strictly more expressive than \mathcal{ML} . A natural question is whether these dynamic logics are different from each other. We will use bisimulations to answer this question.

Because we need to keep track of the changes on the accessibility relation that the dynamic operators can introduce, we will define bisimulations as relations that link a point of evaluation together with the current accessibility relation.

Definition 5 (Bisimulations). Given models $\mathcal{M} = \langle W, R, V \rangle$ and $\mathcal{M}' = \langle W', R', V' \rangle$, together with points $w \in W$ and $w' \in W'$ we say that they are bisimilar and write $\mathcal{M}, w \stackrel{\simeq}{\sim} \mathcal{M}', w'$ if there is a relation $Z \subseteq (W \times \mathcal{P}(W^2)) \times (W' \times \mathcal{P}(W'^2))$ such that $(w, R)Z(w', R')$ satisfying conditions from Figure 7. Which conditions have to be satisfied depends on the operators present in the language.

If needed, we write $\stackrel{\simeq}{\sim}_{\mathfrak{L}}$ to indicate that the bisimulation corresponds to \mathfrak{L} .

Theorem 6 (Invariance for Dynamic Logics). For $\mathcal{ML}(\diamond)$, $\diamond \in \{\langle sw \rangle, \langle gs \rangle, \langle ls \rangle, \langle br \rangle\}$, $\mathcal{M}, w \stackrel{\simeq}{\sim}_{\mathcal{ML}(\diamond)} \mathcal{M}', w'$ implies $\mathcal{M}, w \equiv_{\mathcal{ML}(\diamond)} \mathcal{M}', w'$.

Proof. We will only prove the $\mathcal{ML}(\langle sw \rangle)$ case by structural induction.

The base case holds by (agree), and the \wedge and \neg cases are trivial.

[$\diamond \varphi$ case:] Let $\mathcal{M} = \langle W, R, V \rangle$ and $\mathcal{M}' = \langle W', R', V' \rangle$. Suppose $\mathcal{M}, w \models \diamond \varphi$. Then there is v in W s.t. wRv and $\mathcal{M}, v \models \varphi$. Since Z is a bisimulation, by

<i>always</i>	(nontriv)	Z is not empty
<i>always</i>	(agree)	If $(w, S)Z(w', S')$, w and w' make the same propositional variables true.
\diamond	(zig) (zag)	If wSv , there is $v' \in W'$ s.t. $w'S'v'$ and $(v, S)Z(v', S')$ If $w'S'v'$, there is $v \in W$ s.t. wSv and $(v, S)Z(v', S')$
$\langle sw \rangle$	(<i>sw</i> -zig) (<i>sw</i> -zag)	If wSv , there is $v' \in W'$ s.t. $w'S'v'$ and $(v, S_{vw}^*)Z(v', S_{v'w'}^*)$ If $w'S'v'$, there is $v \in W$ s.t. wSv and $(v, S_{vw}^*)Z(v', S_{v'w'}^*)$
$\langle gs \rangle$	(<i>gs</i> -zig) (<i>gs</i> -zag)	If vSu , there is $v', u' \in W'$ s.t. $v'S'u'$ and $(w, S_{vu}^-)Z(w', S_{v'u'}^-)$ If $v'S'u'$, there is $v, u \in W$ s.t. vSu and $(w, S_{vu}^-)Z(w', S_{v'u'}^-)$
$\langle ls \rangle$	(<i>ls</i> -zig) (<i>ls</i> -zag)	If wSv , there is $v' \in W'$ s.t. $w'S'v'$ and $(v, S_{wv}^-)Z(v', S_{w'v'}^-)$ If $w'S'v'$, there is $v \in W$ s.t. wSv and $(v, S_{wv}^-)Z(v', S_{w'v'}^-)$
$\langle br \rangle$	(<i>br</i> -zig) (<i>br</i> -zag)	If $\neg wSv$, there is $v' \in W'$ s.t. $\neg w'S'v'$ and $(v, S_{wv}^+)Z(v', S_{w'v'}^+)$ If $\neg w'S'v'$, there is $v \in W$ s.t. $\neg wSv$ and $(v, S_{wv}^+)Z(v', S_{w'v'}^+)$

Fig. 1. Conditions for $\mathcal{ML}(\diamond)$ -bisimulations

(zig) we have $v' \in W'$ s.t. $w'R'v'$ and $(v, R)Z(v', R')$. By inductive hypothesis, $\mathcal{M}', v' \models \varphi$ and by definition $\mathcal{M}', w' \models \diamond\varphi$. For the other direction use (zag).

[$\langle sw \rangle\varphi$ case:] For the left to the right direction suppose $\mathcal{M}, w \models \langle sw \rangle\varphi$. Then there is $v \in W$ s.t. wRv and $\mathcal{M}_{vw}^*, v \models \varphi$. Because Z is a bisimulation, by (*sw*-zig) we have $v' \in W'$ s.t. $w'R'v'$ and $(v, R_{vw}^*)Z(v', R_{v'w'}^*)$. By inductive hypothesis, $\mathcal{M}_{v'w'}^*, v' \models \varphi$ and by definition $\mathcal{M}', w' \models \langle sw \rangle\varphi$. For the other direction use (*sw*-zag). \square

3 Expressive Power

With the appropriate notions of bisimulation at hand we can now start the comparison of the expressive power of the different dynamic modal logics we introduced. We will use the following standard definition of when a logic is at least as expressive as another.

Definition 7 ($\mathcal{L} \leq \mathcal{L}'$). *We say that \mathcal{L}' is at least as expressive as \mathcal{L} (notation $\mathcal{L} \leq \mathcal{L}'$) if there is a function Tr between formulas of \mathcal{L} and \mathcal{L}' such that for every model \mathcal{M} and every formula φ of \mathcal{L} we have that*

$$\mathcal{M} \models_{\mathcal{L}} \varphi \text{ iff } \mathcal{M} \models_{\mathcal{L}'} \text{Tr}(\varphi).$$

\mathcal{M} is seen as a model of \mathcal{L} on the left and as a model of \mathcal{L}' on the right, and we use in each case the appropriate semantic relation $\models_{\mathcal{L}}$ or $\models_{\mathcal{L}'}$ as required.

We say that \mathcal{L} and \mathcal{L}' are *uncomparable* if $\mathcal{L} \not\leq \mathcal{L}'$ and $\mathcal{L}' \not\leq \mathcal{L}$.

By inspecting suitable models we can establish the following result.

Theorem 8. *For all $\diamond_1, \diamond_2 \in \{\langle sw \rangle, \langle gs \rangle, \langle ls \rangle, \langle br \rangle\}$ with $\diamond_1 \neq \diamond_2$, $\mathcal{ML}(\diamond_1)$ and $\mathcal{ML}(\diamond_2)$ are uncomparable.*




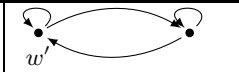
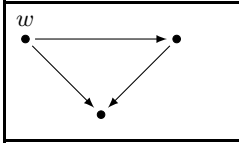
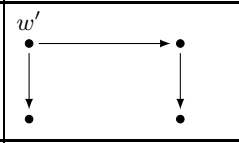
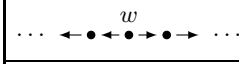
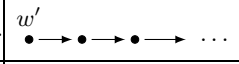
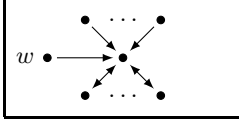
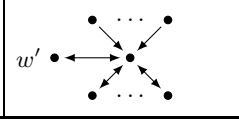
\mathcal{M}	\mathcal{M}'	Distinct by	Bisimilar for
		$\langle br \rangle \langle br \rangle \top$ $\langle gs \rangle \top$	$\mathcal{ML}(\langle ls \rangle)$ $\mathcal{ML}(\langle sw \rangle)$
		$\langle ls \rangle \diamond \top$ $\langle gs \rangle \diamond \top$	$\mathcal{ML}(\langle sw \rangle)$ $\mathcal{ML}(\langle br \rangle)$
		$\langle sw \rangle \langle sw \rangle \diamond \diamond \square \perp$ $[br][br] \perp$	$\mathcal{ML}(\langle gs \rangle)$ $\mathcal{ML}(\langle ls \rangle)$
		$\langle sw \rangle \diamond \square \perp$	$\mathcal{ML}(\langle br \rangle)$
		$\langle ls \rangle \diamond \square \perp$	$\mathcal{ML}(\langle gs \rangle)$

Fig. 2. Bisimilar models and distinguishing formulas

Proof. In Figure 2 we summarize our results by presenting pairs of models that are bisimilar for a given logic and distinguishable by another. More precisely, the formulas given in the third column are false at \mathcal{M}, w and true at \mathcal{M}', w' .

That the models are bisimilar for the given logics can be easily verified for the first two rows. In the third row, the given models are bisimilar for $\mathcal{ML}(\langle gs \rangle)$ and $\mathcal{ML}(\langle ls \rangle)$ because they are bisimilar for \mathcal{ML} , they are acyclic and (for $\mathcal{ML}(\langle gs \rangle)$) they contain the same number of edges. In the fourth row, both models are $\mathcal{ML}(\langle br \rangle)$ -bisimilar since they are infinite, hence one can add as many links as needed to points that are modally bisimilar.

Finally, the pointed models of the last row are the same graph with a different evaluation point. The graph is a star that has infinitely many ingoing branches, and infinitely many ingoing-outgoing branches. w is a point located at the end of an ingoing branch, and w' is at the end of an ingoing-outgoing branch. Let us present the $\mathcal{ML}(\langle gs \rangle)$ -bisimulation as a game between Spoiler and Duplicator. If Spoiler moves to the center of the star, Duplicator can do the same and both situations become undistinguishable. If Spoiler deletes one of the ingoing edges that has w or w' as origin, then Duplicator does the same on the other graph, and any further edge deletion can also be imitated. If Spoiler deletes the outgoing edge that goes from the center of the graph towards w' , then Duplicator can delete any outgoing edge without changing the graph, given that there are infinitely many edges of both kinds. \square

4 Model Checking Dynamic Logics

In this section we establish complexity results for the model checking task in the various dynamic modal logics we presented. All the results are established using a

similar argument: hardness proofs are done by encoding the satisfiability problem of Quantified Boolean Formulas (QBF) [8] as the model checking problem of each logic. While the idea behind the encoding is the same for all the logics involved, the encoding needs to be slightly modified in each case taking into consideration the semantics of the various dynamic operators.

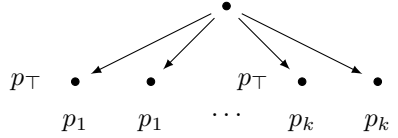
PSpace-hardness for global sabotage was already proved in [7,6], but we provide here a more direct proof.

Theorem 9. *For $\diamond \in \{\langle sw \rangle, \langle gs \rangle, \langle ls \rangle, \langle br \rangle\}$, model checking for any of the logics $\mathcal{ML}(\diamond)$ is PSpace-hard.*

Proof. We will reduce the PSpace-complete satisfiability problem of Quantified Boolean Formulas (QBF) to the model checking problem of each of these logics. For a complete proof of the case of $\mathcal{ML}(\langle sw \rangle)$, consult the appendix.

Consider $\mathcal{ML}(\langle sw \rangle)$. Let α be a QBF formula with variables $\{x_1, \dots, x_k\}$. Without loss of generality we can assume that α has no free variables and no variable is quantified twice. One can build in polynomial time the relational structure $\mathcal{M}_k = \langle W, R, V \rangle$ over a signature with one relational symbol and propositions $\{p_\top, p_1, \dots, p_k\}$, where:

$$\begin{aligned} W &= \{w\} \cup \{w_i^1, w_i^0 \mid 1 \leq i \leq k\} \\ V(p_i) &= \{w_i^1, w_i^0\} \\ V(p_\top) &= \{w_i^1 \mid 1 \leq i \leq k\} \\ R &= \{(w, w_i^1), (w, w_i^0) \mid 1 \leq i \leq k\} \end{aligned}$$



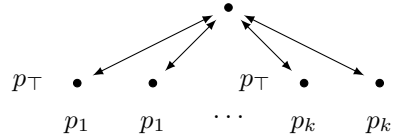
Let $(\)'$ be the following linear translation from QBF to $\mathcal{ML}(\langle sw \rangle)$

$$\begin{aligned} (\exists x_i. \alpha)' &= \langle sw \rangle(p_i \wedge \diamond(\alpha)') \\ (x_i)' &= \neg \diamond(p_i \wedge p_\top) \\ (\neg \alpha)' &= \neg(\alpha)' \\ (\alpha \wedge \beta)' &= (\alpha)' \wedge (\beta)'. \end{aligned}$$

It remains to see that α is satisfiable if, and only if, $\mathcal{M}_k, w \models (\alpha)'$ holds. This part of the proof is in the appendix. This shows that the model checking problem of $\mathcal{ML}(\langle sw \rangle)$ is PSpace-hard.

For $\mathcal{ML}(\langle gs \rangle)$ and $\mathcal{ML}(\langle ls \rangle)$, we use the following model:

$$\begin{aligned} W &= \{w\} \cup \{w_i^1, w_i^0 \mid 1 \leq i \leq k\} \\ V(p_i) &= \{w_i^1, w_i^0\} \\ V(p_\top) &= \{w_i^1 \mid 1 \leq i \leq k\} \\ R &= \{(w, w_i^1), (w, w_i^0), (w_i^1, w), (w_i^0, w) \\ &\quad \mid 1 \leq i \leq k\} \end{aligned}$$



Let $(\)'$ be the following linear translation from QBF to $\mathcal{ML}(\langle ls \rangle)$:

$$\begin{aligned} (\exists x_i. \alpha)' &= \langle ls \rangle(p_i \wedge \diamond(\alpha)') \\ (x_i)' &= \neg \diamond(p_i \wedge p_\top) \\ (\neg \alpha)' &= \neg(\alpha)' \\ (\alpha \wedge \beta)' &= (\alpha)' \wedge (\beta)'. \end{aligned}$$

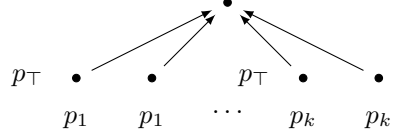
From QBF to $\mathcal{ML}(\langle gs \rangle)$, we provide the following translation:

$$\begin{aligned} (\exists x_i. \alpha)' &= \langle gs \rangle ((\neg \diamond(p_i \wedge p_\top) \vee \neg \diamond(p_i \wedge \neg p_\top)) \wedge \diamond(p_i \wedge \diamond(\alpha)')) \\ (x_i)' &= \neg \diamond(p_i \wedge p_\top) \\ (\neg \alpha)' &= \neg(\alpha)' \\ (\alpha \wedge \beta) &= (\alpha)' \wedge (\beta)'. \end{aligned}$$

In both cases, showing that a QBF formula α is satisfiable if, and only if, $\mathcal{M}_k, w \models (\alpha)'$ holds can be done similarly to the case of $\mathcal{ML}(\langle sw \rangle)$.

Finally, to prove PSpace-hardness for $\mathcal{ML}(\langle br \rangle)$, build the following model:

$$\begin{aligned} W &= \{w\} \cup \{w_i^1, w_i^0 \mid 1 \leq i \leq k\} \\ V(p_i) &= \{w_i^1, w_i^0\} \\ V(p_\top) &= \{w_i^1 \mid 1 \leq i \leq k\} \\ R &= \{(w_i^1, w), (w_i^0, w) \mid 1 \leq i \leq k\} \end{aligned}$$



And use the following linear translation $()'$:

$$\begin{aligned} (\exists x_i. \alpha)' &= \langle br \rangle (p_i \wedge \diamond(\alpha)') \\ (x_i)' &= \diamond(p_i \wedge p_\top) \\ (\neg \alpha)' &= \neg(\alpha)' \\ (\alpha \wedge \beta) &= (\alpha)' \wedge (\beta)'. \end{aligned}$$

□

Theorem 10. *Model checking for $\mathcal{ML}(\langle sw \rangle, \langle gs \rangle, \langle ls \rangle, \langle br \rangle)$ is in PSpace.*

Proof. The evaluation of the truth of a formula in a model can be done by a polynomial space algorithm that follows Definition 3.

The algorithm works on the same copy of the model, except when dealing with formulas whose main connector is $\langle sw \rangle$, $\langle gs \rangle$, $\langle ls \rangle$ or $\langle br \rangle$ (i.e., dynamic operators). In such cases, by proceeding depth-first among at most $|W|$ possible choices, the algorithm only allocates as much additional space as the size of the initial model to store the modified copy. This memory can be reclaimed once the result of the recursive call is known. The maximum number of copies of the input model in memory is bounded by the nesting of dynamic operators of the input formula. Hence the algorithm runs using only polynomial space. □

With the previous results we get:

Theorem 11. *For $\diamond \in \{\langle sw \rangle, \langle gs \rangle, \langle ls \rangle, \langle br \rangle\}$, model checking for any of the logics $\mathcal{ML}(\diamond)$ is PSpace-complete.*

5 Conclusions

In this article we investigate dynamic modal logics that can modify the model during the evaluation of a formula. Dynamic Epistemic Logics (DEL) as those investigated in [9, 5, 11, 13] are well known examples of languages which can also update the model during evaluation. The standard update operation used in DELs is to move evaluation to a submodel defined by a certain ‘announcement’,

e.g., to the model representing the fact that φ is now known, obtained as the restriction to all the nodes satisfying a formula φ . Instead, in this article we investigate logics that can explicitly modify the accessibility relation, as the sabotage logics first introduced by van Benthem in [12].

We introduce a number of operators with both local and global effects, and which can add, delete and modify edges in the accessibility relation. The goal was to investigate the different degrees of liberty that the operators offered, and how much overlap there was between the logics they defined, and the models they could describe.

We show in Sections 2 and 3 that the languages obtained by the extension of the basic modal logic with each of the dynamic operators can be characterized using bisimulations. Actually, even though each operator requires a particular pair of zig and zag conditions, the definition is modular and the set up homogeneous. All the bisimulations involved are of the same type, linking a pair of point of evaluation and accessibility relation in one model, with a similar pair in the other. Moreover, a suitable definition of bisimulation for the basic modal logic extended with any combination of the new dynamic operators can be obtained by using the adequate zig and zag conditions associated to the operators involved. Summing up then, even though the logics obtained are different in each case, they are all amenable to fairly classical modal analysis.

In Section 4 we turn to model checking, and show that the complexity of this reasoning task is PSpace-complete for all the logics considered. Once more, the proofs are fairly homogeneous in all cases. The general set up is the encoding of the PSpace-complete QBF satisfiability problem in each of the logics. In each case, a suitable representation for the assignment and the concrete translation used needs to be defined, but once this is done the proof is similar.

More precisely, we established the complexity of the *combined* model checking task, measured in function of the length of an input model and an input formula. It is also possible to consider the task of model checking against a fixed model, measuring its complexity in function of the size of an input formula (this is known as the formula complexity). One can also fix a formula and measure the complexity of model checking in function of the length of an input model (known as the program complexity or data complexity). Both notions were introduced in [14], and it has been shown in [6] that the formula complexity and the program complexity of $\mathcal{ML}(\langle gs \rangle)$ are respectively linear and polynomial. We believe that the proof generalizes to $\mathcal{ML}(\langle ls \rangle)$, $\mathcal{ML}(\langle sw \rangle)$ and $\mathcal{ML}(\langle br \rangle)$ with identical results.

Another natural direction for future research would be to investigate the complexity of the satisfiability problem of these logics. From [6], we already know that $\mathcal{ML}(\langle gs \rangle)$ is undecidable. We conjecture that using techniques from [31], it is possible to prove that the problem is undecidable in all remaining cases.

Acknowledgments. This work was partially supported by grants ANPCyT-PICT-2008-306, ANPCyT-PIC-2010-688, the FP7-PEOPLE-2011-IRSES Project “Mobility between Europe and Argentina applying Logics to Systems” (MEALS) and the Laboratoire Internationale Associé “INFINIS”.

References

1. Areces, C., Figueira, D., Figueira, S., Mera, S.: The expressive power of memory logics. *Review of Symbolic Logic* 4(2), 290–318 (2011)
2. Blackburn, P., de Rijke, M., Venema, Y.: *Modal Logic*. Cambridge University Press (2001)
3. Blackburn, P., Seligman, J.: Hybrid languages. *Journal of Logic, Language and Information* 4, 251–272 (1995)
4. Blackburn, P., Wolter, F., van Benthem, J. (eds.): *Handbook of Modal Logics*. Elsevier (2006)
5. Gerbrandy, J.: *Bisimulations on Planet Kripke*. Ph.D. thesis, University of Amsterdam. ILLC Dissertation series DS-1999-01 (1999)
6. Löding, C., Rohde, P.: Model Checking and Satisfiability for Sabotage Modal Logic. In: Pandya, P.K., Radhakrishnan, J. (eds.) *FSTTCS 2003*. LNCS, vol. 2914, pp. 302–313. Springer, Heidelberg (2003)
7. Löding, C., Rohde, P.: Solving the Sabotage Game Is PSPACE-Hard. In: Rovan, B., Vojtáš, P. (eds.) *MFCS 2003*. LNCS, vol. 2747, pp. 531–540. Springer, Heidelberg (2003)
8. Papadimitriou, C.: *Computational Complexity*. Addison-Wesley (1994)
9. Plaza, J.: Logics of public communications. *Synthese* 158(2), 165–179 (2007)
10. Rohde, P.: *On games and logics over dynamically changing structures*. Ph.D. thesis, RWTH Aachen (2006)
11. van Benthem, J.: Logics for information update. In: *TARK 2001: Proceedings of the 8th Conference on Theoretical Aspects of Rationality and Knowledge*, pp. 51–67. Morgan Kaufmann Publishers Inc. (2001)
12. van Benthem, J.: An Essay on Sabotage and Obstruction. In: Hutter, D., Stephan, W. (eds.) *Mechanizing Mathematical Reasoning*. LNCS (LNAI), vol. 2605, pp. 268–276. Springer, Heidelberg (2005)
13. van Ditmarsch, H., van der Hoek, W., Kooi, B.: *Dynamic Epistemic Logic*. Kluwer (2007)
14. Vardi, M.Y.: The complexity of relational query languages (extended abstract). In: Lewis, H.R., Simons, B.B., Burkhard, W.A., Landweber, L.H. (eds.) *STOC*, pp. 137–146. ACM (1982)

Appendix

Theorem 4. $\mathcal{ML}(\diamond)$ does not have the tree model property, for $\diamond \in \{\langle sw \rangle, \langle gs \rangle, \langle ls \rangle, \langle br \rangle\}$

Proof. We are going to list formulas that force no-treelike models:

1. The formula

$$\varphi = p \wedge \left(\bigwedge_{1 \leq i \leq 3} \Box^i \neg p \right) \wedge \langle sw \rangle \diamond \diamond p$$

is true at a state w in a model, only if w has a reflexive successor.

Suppose we evaluate φ at some state w of an arbitrary model. The ‘static’ part of the formula $p \wedge (\bigwedge_{1 \leq i \leq 3} \Box^i \neg p)$ makes sure that p is true in w and that no p state is reachable within three steps from w (in particular, w cannot be reflexive).

Because $\langle sw \rangle \diamond \diamond p$ is true at w , there should be an R -successor v where $\diamond \diamond p$ holds once the accessibility relation has been updated to R_{vw}^* . That is, v has to reach a p -state in exactly two R_{vw}^* -steps. But the only p -state sufficiently close is w which is reachable in one step. As w is not reflexive, v has to be reflexive so that we can linger at v for one loop and reach p in the correct number of states.

2. The formula

$$\varphi = \diamond \diamond \top \wedge [ls] \square \perp$$

is true at a state w in a model, only if w is reflexive.

Suppose we evaluate φ at some state w of an arbitrary model. On one hand, the ‘static’ part of the formula $\diamond \diamond \top$ ensures it is possible to take two accessibility relations. On the other hand, the ‘dynamic’ part of the formula $[ls] \square \perp$ tells us that after taking any accessibility relation and eliminating it, it is no longer possible to go anywhere else. This can only happen if the point w is reflexive and does not have any other outgoing links.

3. The formula

$$\varphi = \diamond \diamond \top \wedge [gs] \square \perp$$

(from [6]) is true at a state w in a model, only if w is reflexive.

4. The formula

$$\varphi = \langle br \rangle \square \perp$$

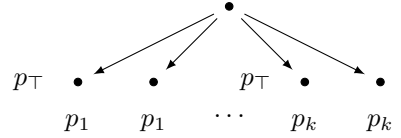
is only satisfiable in models that have at least two unconnected points. \square

Theorem 11. *Model checking for $\mathcal{ML}(\langle sw \rangle)$ is PSpace-hard.*

Proof. We will reduce the PSpace-complete satisfiability problem of Quantified Boolean Formulas (QBF) to the model checking problem of $\mathcal{ML}(\langle sw \rangle)$.

Let α be a QBF formula with variables $\{x_1, \dots, x_k\}$. Without loss of generality we can assume that α has no free variables and no variable is quantified twice. One can build in polynomial time the relational structure $\mathcal{M}_k = \langle W, R, V \rangle$ over a signature with one relational symbol and propositions $\{p_\top, p_1, \dots, p_k\}$, where:

$$\begin{aligned} W &= \{w\} \cup \{w_i^1, w_i^0 \mid 1 \leq i \leq k\} \\ V(p_i) &= \{w_i^1, w_i^0\} \\ V(p_\top) &= \{w_i^1 \mid 1 \leq i \leq k\} \\ R &= \{(w, w_i^1), (w, w_i^0) \mid 1 \leq i \leq k\} \end{aligned}$$



Let ()' be the following linear translation from QBF to $\mathcal{ML}(\langle sw \rangle)$

$$\begin{aligned} (\exists x_i. \alpha)' &= \langle sw \rangle (p_i \wedge \diamond(\alpha)') \\ (x_i)' &= \neg \diamond (p_i \wedge p_\top) \\ (\neg \alpha)' &= \neg(\alpha)' \\ (\alpha \wedge \beta)' &= (\alpha)' \wedge (\beta)'. \end{aligned}$$

It remains to see that α is satisfiable iff $\mathcal{M}_k, w \models (\alpha)'$ holds. Let us write $v \models_{\text{qbf}} \alpha$ if valuation $v : \{x_1, \dots, x_k\} \rightarrow \{0, 1\}$ satisfies α . For a model \mathcal{M} with relation R we define $v_R : \{x_1, \dots, x_k\}$ as “ $v_R(x_i) = 1$ iff $(w, w_i^1) \notin R$ ”, in the present case, iff the link between w and w_i^1 has been swapped.

Let β be any subformula of α . We will show by induction on β that $\mathcal{M}, w \models (\beta)'$ iff $v_R \models_{\text{qbf}} \beta$. The first observation is that R satisfies i) if x_i is free in β , then $(w, w_i^1) \notin R$ or $(w, w_i^0) \notin R$ but not both, and ii) if x_i is not free in β then $(w, w_i^1) \in R$ and $(w, w_i^0) \in R$. From here it will follow that $\mathcal{M}_k, w \models (\alpha)'$ iff $v \models_{\text{qbf}} \alpha$ for any v since α has no free variables, iff α is satisfiable.

For the base case, $v_R \models_{\text{qbf}} x_i$ iff $(w, w_i^1) \notin R$ which implies (from the definition of \mathcal{M}_k) $\mathcal{M}, w \models (x_i)'$. For the other direction, suppose $\mathcal{M}, w \not\models (x_i)'$. Hence $\mathcal{M}, w \models \diamond(p_i \wedge p_\top)$ which implies $(w, w_i^1) \in R$ and $u_R \not\models_{\text{qbf}} x_i$.

The boolean cases follow directly from the inductive hypothesis.

Consider the case $\beta = \exists x_i.\gamma$. Since no variable is bound twice in α we know $(w, w_{x_i}^1) \in R$ and $(w, w_i^0) \in R$. We have $v_R \models_{\text{qbf}} \beta$ iff $(v_R[x_i \mapsto 0]) \models_{\text{qbf}} \gamma$ or $v_R[x_i \mapsto 1] \models_{\text{qbf}} \gamma$ iff $(v_{R^{w_i^0 w}} \models_{\text{qbf}} \gamma$ or $v_{R^{w_i^1 w}} \models_{\text{qbf}} \gamma)$. By inductive hypothesis, this is the case if and only if $(\mathcal{M}^{w_i^0 w}, w_i^0 \models \diamond(\gamma)'$ or $\mathcal{M}^{w_i^1 w}, w_i^1 \models \diamond(\gamma)'$ iff $\mathcal{M}, w \models \langle sw \rangle(p_i \wedge \diamond(\gamma))'$ iff $\mathcal{M}, w \models (\exists x_i.\gamma)'$. \square

Standard Completeness for Extensions of MTL: An Automated Approach

Paolo Baldi, Agata Ciabattoni, and Lara Spendier

Technische Universität Wien, Austria

Abstract. We provide general conditions on hypersequent calculi that guarantee standard completeness for the formalized logics. These conditions are implemented in the PROLOG system *AxiomCalc* that takes as input any suitable axiomatic extension of Monoidal T-norm Logic **MTL** and outputs a hypersequent calculus for the logic and the result of the check. Our approach subsumes many existing results and allows for the computerized discovery of new fuzzy logics.

1 Introduction

Standard completeness, that is completeness of a logic with respect to algebras based on truth values in $[0, 1]$, has received increasing attention in the last few years. In a standard complete logic connectives are interpreted by suitable functions on $[0, 1]$, and this makes it a fuzzy logic in the sense of [11]. For example, conjunction and implication can be interpreted by a t-norm¹ and its residuum or by classes of t-norms. Gödel and Łukasiewicz being prominent examples of logics based on particular t-norms, while e.g. Monoidal t-norm logic **MTL** [8] is based on the class of left-continuous t-norms. **MTL** is a useful basic system as many interesting logics, including Gödel and Łukasiewicz logics, can be obtained by extending **MTL** with suitable properties expressed as Hilbert axioms.

Checking or discovering whether a logic is standard complete is sometimes a challenging task which deserves a paper for each specific logic, e.g., [5, 10, 12]. It is traditionally established by semantic techniques which are inherently logic-specific. Given a logic **L** described in a Hilbert-style system, semantic proofs usually consist of the following four steps (see, e.g., [5, 7, 8, 10, 11, 15]):

1. The algebraic semantics of the logic is identified (\mathcal{L} -algebras).
2. It is shown that if a formula is not valid in an \mathcal{L} -algebra, then it is not valid in a countable \mathcal{L} -chain (linearly ordered \mathcal{L} -algebra).
3. It is shown that any countable \mathcal{L} -chain can be embedded into a countable *dense* \mathcal{L} -chain by adding countably many new elements to the algebra and extending the operations appropriately. This establishes *rational completeness*: a formula is derivable in **L** iff it is valid in all dense \mathcal{L} -chains.
4. Finally, a countable dense \mathcal{L} -chain is embedded into a standard \mathcal{L} -algebra, that is an \mathcal{L} -algebra with lattice reduct $[0, 1]$, using a Dedekind-MacNeille-style completion.

¹ T-norms are the main tool in fuzzy set theory to combine vague information.

The crucial step 3. above (rational completeness) is the most difficult to establish as it relies on finding the right embedding, if any. A different approach to step 3. was proposed in [13] by using proof-theoretic techniques. The idea in [13] is that the admissibility of a particular syntactic rule (called *density*) in a logic \mathbf{L} can lead to a proof of rational completeness for \mathbf{L} . This is for instance the case when \mathbf{L} is any axiomatic extension of \mathbf{MTL} . Introduced by Takeuti and Titani in [17], the density rule formalized Hilbert-style has the following form

$$\frac{(A \rightarrow p) \vee (p \rightarrow B) \vee C}{(A \rightarrow B) \vee C}$$

where p is a propositional variable not occurring in A , B , or C . Ignoring C , this can be read contrapositively as saying (very roughly) “if $A > B$, then $A > p$ and $p > B$ for some p ”; hence the name “density” and the intuitive connection with rational completeness.

The new approach was used in [13] to establish standard completeness for various logics for which semantic techniques do not appear to work. Following this method, to establish standard completeness for a logic \mathbf{L} we need to

- (a) define a suitable proof system HL for \mathbf{L} extended with the density rule
- (b) check that this rule is eliminable (or admissible) in HL , i.e. that density does not enlarge the set of provable formulas
- (c) standard completeness may then be obtained in many cases (but not in general) by means of the Dedekind-MacNeille completion.

A convenient proof system $HMTL$ for \mathbf{MTL} uses hypersequents, which are a simple generalization of Gentzen sequents, see [1,14].

Step (b) above (density-elimination) was shown in [2,13] for various calculi, including $HMTL$. The proofs in [2,13] are calculi-specific and use heavy combinatorial arguments, in close analogy with Gentzen style cut-elimination proofs. A different method to eliminate applications of the density rule from derivations was introduced in [6]. It shows that each hypersequent calculus obtained by extending $HMTL$ by certain *sequent rules* admits density-elimination. Though more general than the proofs in [2,13], this result does not apply to many interesting extensions of \mathbf{MTL} whose additional axioms require *hypersequent rules*; e.g., it does not apply to weak nilpotent minimum logic \mathbf{WNM} [4,8].

The aim of this paper is to *automate* standard completeness proofs for large classes of axiomatic extensions of \mathbf{MTL} .

We introduce the program *AxiomCalc* that automates steps (a)-(c) above for propositional logics extending \mathbf{MTL} by any Hilbert axiom within the class \mathcal{P}_3 in the syntactic classification of [4]. The main theoretical contribution of this paper is the identification of sufficient conditions on hypersequent rules that ensure standard completeness for the formalized logics. As shown in Section 3, our conditions allow indeed density-elimination (and hence, by [13], they lead to rational completeness). Standard completeness follows, being the axioms we deal with preserved under suitable forms of Dedekind-MacNeille completions [3].

AxiomCalc implements the systematic procedure in [4] to define cut-free hypersequent calculi from Hilbert systems for large classes of logics (step (a)), and the check of our sufficient conditions which account for steps (b) and (c) *at once*.

Our approach subsumes many existing results on standard completeness for specific logics and allows for the computerized discovery of new fuzzy logics. For instance, it applies to **WNM** and to the logics obtained by extending **MTL** with the axioms $\neg(\alpha \cdot \beta)^n \vee ((\alpha \wedge \beta)^{n-1} \rightarrow (\alpha \cdot \beta)^n)$ for each $n \geq 2$. The latter family of logics is new and was discovered by playing with the PROLOG system *AxiomCalc*, available at <http://www.logic.at/people/lara/axiomcalc.html>.

2 Automated Proof Theory for Extensions of MTL

In this section we present the system *AxiomCalc* and describe the calculi that it generates. The program is a PROLOG-implementation of the algorithm in [4].

The basic system we will deal with is Monoidal T-norm Logic **MTL** [8] which was shown in [10] to be the logic of left continuous t-norms² and their residua. **MTL** is obtained by adding the prelinearity axiom $(\alpha \rightarrow \beta) \vee (\beta \rightarrow \alpha)$ to intuitionistic logic without contraction (also known as HBCK [16] or Full Lambek calculus with exchange and weakening **FLew**).

Formulas of **MTL** are built from propositional variables and the constants 0 and 1 by using \rightarrow (implication), \wedge (additive conjunction), \cdot (multiplicative conjunction), and \vee (disjunction). We use $\neg\alpha$ as an abbreviation for $\alpha \rightarrow 0$.

In the following α, β, \dots will stand for both formulas and for metavariables for formulas. To distinguish between rule applications and rule schemas we will denote finite (possibly empty) multisets of formulas with $\Gamma, \Delta, \Sigma, \Theta, \Lambda$ and metavariables for multisets of formulas with $\overline{\Gamma}, \overline{\Delta}, \overline{\Sigma}, \overline{\Theta}, \overline{\Lambda}$. Metavariables $\overline{\Pi}, \overline{\Psi}$ will stand for stoups, i.e., either a formula or the empty set.

Definition 1. A hypersequent is a finite multiset $S_1 \mid \dots \mid S_n$ where each $S_i, i = 1 \dots n$ is a sequent, called a component of the hypersequent.

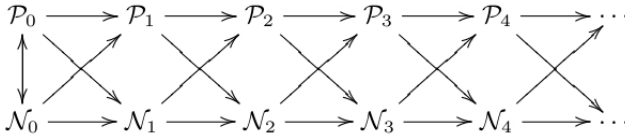
The symbol “ \mid ” is intended to denote disjunction at the meta-level. In this paper we will only consider hypersequents whose components contain at most one formula on their right-hand side.

Henceforth we will denote hypersequents by G, H and sequents (possibly built from metavariables) by S_i, C_i . The hypersequent calculus contains initial axioms, cut, and logical rules. These are as in sequent calculus, the only difference being the presence of a (possibly empty) side hypersequent G . Structural rules are divided into two groups: internal and external rules. The latter, which permit interaction between components, increase the expressive power of the hypersequent calculus with respect to sequent calculus.

² A t-norm is a commutative, associative, increasing function $* : [0, 1]^2 \rightarrow [0, 1]$ with identity element 1. $*$ is *left continuous* iff whenever $\{x_n\}, \{y_n\}$ ($n \in \mathbb{N}$) are increasing sequences in $[0, 1]$ s.t. their suprema are x and y , then $\sup\{x_n * y_n : n \in \mathbb{N}\} = x * y$. The residuum of $*$ is a function \rightarrow^* where $x \rightarrow^* y = \max\{z \mid x * z \leq y\}$.

Table 1. Hypersequent calculus *HMTL* for **MTL**

$\frac{G \overline{\Gamma} \Rightarrow \alpha \quad G \alpha, \overline{\Delta} \Rightarrow \overline{\Pi}}{G \overline{\Gamma}, \overline{\Delta} \Rightarrow \overline{\Pi}} \text{ (Cut)}$	$\frac{}{G \alpha \Rightarrow \alpha} \text{ (init)}$	$\frac{}{G 0 \Rightarrow} \text{ (0l)}$
$\frac{G \overline{\Gamma} \Rightarrow \alpha \quad G \overline{\Delta} \Rightarrow \beta}{G \overline{\Gamma}, \overline{\Delta} \Rightarrow \alpha \cdot \beta} \text{ (}\cdot\text{r)}$	$\frac{G \alpha, \beta, \overline{\Gamma} \Rightarrow \overline{\Pi}}{G \alpha \cdot \beta, \overline{\Gamma} \Rightarrow \overline{\Pi}} \text{ (}\cdot\text{l)}$	$\frac{G \overline{\Gamma} \Rightarrow \overline{\Pi}}{G 1, \overline{\Gamma} \Rightarrow \overline{\Pi}} \text{ (1l)}$
$\frac{G \overline{\Gamma} \Rightarrow \alpha \quad G \beta, \overline{\Delta} \Rightarrow \overline{\Pi}}{G \overline{\Gamma}, \alpha \rightarrow \beta, \overline{\Delta} \Rightarrow \overline{\Pi}} \text{ (}\rightarrow\text{l)}$	$\frac{G \alpha, \overline{\Gamma} \Rightarrow \beta}{G \overline{\Gamma} \Rightarrow \alpha \rightarrow \beta} \text{ (}\rightarrow\text{r)}$	$\frac{G \overline{\Gamma} \Rightarrow}{G \overline{\Gamma} \Rightarrow 0} \text{ (0r)}$
$\frac{G \overline{\Gamma} \Rightarrow \alpha \quad G \overline{\Gamma} \Rightarrow \beta}{G \overline{\Gamma} \Rightarrow \alpha \wedge \beta} \text{ (}\wedge\text{r)}$	$\frac{G \alpha_i, \overline{\Gamma} \Rightarrow \overline{\Pi}}{G \alpha_1 \wedge \alpha_2, \overline{\Gamma} \Rightarrow \overline{\Pi}} \text{ (}\wedge\text{l)}$	$\frac{}{G \Rightarrow 1} \text{ (1r)}$
$\frac{G \alpha, \overline{\Gamma} \Rightarrow \overline{\Pi} \quad G \beta, \overline{\Gamma} \Rightarrow \overline{\Pi}}{G \alpha \vee \beta, \overline{\Gamma} \Rightarrow \overline{\Pi}} \text{ (}\vee\text{l)}$	$\frac{G \overline{\Gamma} \Rightarrow \alpha_i}{G \overline{\Gamma} \Rightarrow \alpha_1 \vee \alpha_2} \text{ (}\vee\text{r)}$	$\frac{G \overline{\Gamma} \Rightarrow \overline{\Pi}}{G \overline{\Gamma}, \alpha \Rightarrow \overline{\Pi}} \text{ (wl)}$
$\frac{G \overline{\Gamma} \Rightarrow \overline{\Pi} \mid \overline{\Gamma} \Rightarrow \overline{\Pi}}{G \overline{\Gamma} \Rightarrow \overline{\Pi}} \text{ (EC)}$	$\frac{G}{G \overline{\Gamma} \Rightarrow \overline{\Pi}} \text{ (EW)}$	$\frac{G \overline{\Gamma} \Rightarrow}{G \overline{\Gamma} \Rightarrow \overline{\Pi}} \text{ (wr)}$
$\frac{G \overline{\Gamma}_1, \overline{\Delta}_1 \Rightarrow \overline{\Pi}_1 \quad G \overline{\Gamma}_2, \overline{\Delta}_2 \Rightarrow \overline{\Pi}_2}{G \overline{\Gamma}_1, \overline{\Gamma}_2 \Rightarrow \overline{\Pi}_1 \mid \overline{\Delta}_1, \overline{\Delta}_2 \Rightarrow \overline{\Pi}_2} \text{ (com)}$		


Fig. 1. Classification $(\mathcal{N}_n, \mathcal{P}_n)$ [4]

The (cut-free) hypersequent calculus *HMTL* for **MTL** is obtained by adding Avron's communication rule (*com*) to the hypersequent version of the sequent calculus for **FLew**, see Table 1. (Note that a sequent rule can be easily transformed into a hypersequent rule by adding the context G everywhere).

A classification of Hilbert axioms in the language of **FLew** was introduced in [4]. It is based on classes $(\mathcal{N}_n, \mathcal{P}_n)$ which intuitively account for the difficulty to deal with the axioms proof theoretically.

The general grammar for determining the class of each axiom is as follows (\mathcal{A} is the set of atomic formulas):

$$\begin{aligned} \mathcal{P}_0 &::= \mathcal{N}_0 ::= \mathcal{A} \\ \mathcal{P}_{n+1} &::= \mathcal{N}_n \mid \mathcal{P}_{n+1} \cdot \mathcal{P}_{n+1} \mid \mathcal{P}_{n+1} \vee \mathcal{P}_{n+1} \mid 1 \\ \mathcal{N}_{n+1} &::= \mathcal{P}_n \mid \mathcal{P}_{n+1} \rightarrow \mathcal{N}_{n+1} \mid \mathcal{N}_{n+1} \wedge \mathcal{N}_{n+1} \mid 0 \end{aligned}$$

Table 2. Some axioms and their corresponding logics

Class	Axiom	Rule (cf. Table 3)	Logic
\mathcal{N}_2	$\alpha \rightarrow \alpha \cdot \alpha$	(c)	Gödel logic G
\mathcal{P}_2	$\alpha \vee \neg \alpha$	(em)	Classical Logic CL
\mathcal{P}_3	$\neg \alpha \vee \neg \neg \alpha$ $\alpha \vee (\alpha \rightarrow \beta) \vee (\alpha \wedge \beta \rightarrow \gamma)$ $\neg(\alpha \cdot \beta) \vee (\alpha \wedge \beta \rightarrow \alpha \cdot \beta)$ $\neg(\alpha \cdot \beta)^n \vee ((\alpha \wedge \beta)^{n-1} \rightarrow (\alpha \cdot \beta)^n)$	(lq) (bc2) (wnm) (wnm) ⁿ	SMTL 3-valued G (with (c)) WNM new!

Table 3. Some analytic rules

$\frac{\overline{\Gamma}, \overline{\Gamma}, \overline{\Delta} \Rightarrow \overline{\Pi}}{\overline{\Gamma}, \overline{\Delta} \Rightarrow \overline{\Pi}} \text{ (c)} \quad \frac{G \overline{\Gamma}, \overline{\Delta} \Rightarrow \overline{\Pi}}{G \overline{\Gamma} \Rightarrow \overline{\Delta} \Rightarrow \overline{\Pi}} \text{ (em)} \quad \frac{G \overline{\Gamma}_1, \overline{\Gamma}_2 \Rightarrow}{G \overline{\Gamma}_1 \Rightarrow \overline{\Gamma}_2 \Rightarrow} \text{ (lq)}$
$\frac{G \overline{\Gamma}_1, \overline{\Delta}_2 \Rightarrow \overline{\Pi}_2 \quad G \overline{\Gamma}_1, \overline{\Delta}_3 \Rightarrow \overline{\Pi}_3 \quad G \overline{\Gamma}_2, \overline{\Delta}_3 \Rightarrow \overline{\Pi}_3}{G \overline{\Delta}_3 \Rightarrow \overline{\Pi}_3 \quad \overline{\Gamma}_2, \overline{\Delta}_2 \Rightarrow \overline{\Pi}_2 \quad \overline{\Gamma}_1, \overline{\Delta}_1 \Rightarrow \overline{\Pi}_1} \text{ (bc2)}$
$\frac{G \overline{\Gamma}_2, \overline{\Gamma}_1, \overline{\Delta}_1 \Rightarrow \overline{\Pi}_1 \quad G \overline{\Gamma}_1, \overline{\Gamma}_3, \overline{\Delta}_1 \Rightarrow \overline{\Pi}_1}{G \overline{\Gamma}_1, \overline{\Gamma}_1, \overline{\Delta}_1 \Rightarrow \overline{\Pi}_1 \quad G \overline{\Gamma}_2, \overline{\Gamma}_3, \overline{\Delta}_1 \Rightarrow \overline{\Pi}_1} \text{ (wnm)}$
$\frac{\{G (\overline{\Gamma}_i, \overline{\Gamma}_j)^n, \overline{\Sigma} \Rightarrow \overline{\Pi}\}_{1 \leq i, j \leq (n-1)} \quad \{G (\overline{\Gamma}_i, \overline{\Gamma}_{n+2j-1})^n, \overline{\Sigma} \Rightarrow \overline{\Pi}\}_{1 \leq i \leq (n-1); 1 \leq j \leq n} \quad \{G (\overline{\Gamma}_{n+2i-2}, \overline{\Gamma}_j)^n, \overline{\Sigma} \Rightarrow \overline{\Pi}\}_{1 \leq i \leq n; 1 \leq j \leq (n-1)} \quad \{G (\overline{\Gamma}_{n+2i-2}, \overline{\Gamma}_{n+2j-1})^n, \overline{\Sigma} \Rightarrow \overline{\Pi}\}_{1 \leq i, j \leq n}}{G \overline{\Gamma}_n, \dots, \overline{\Gamma}_{(3n-1)} \Rightarrow \overline{\Gamma}_1, \dots, \overline{\Gamma}_{n-1}, \overline{\Sigma} \Rightarrow \overline{\Pi}} \text{ (wnm)}^n$

A graphical representation of this classification is depicted in Figure 1. Note that the arrows \rightarrow stand for inclusions \subseteq of the classes.

Paper 4 also contains a procedure to transform axioms within the classes $\mathcal{N}_2, \mathcal{P}_2$ and \mathcal{P}_3 into equivalent rules which preserve cut-admissibility once added to the (hypersequent version of) the calculus for **FLew**.

Example 1. The axiom $\neg(\alpha \cdot \beta)^2 \vee ((\alpha \wedge \beta) \rightarrow (\alpha \cdot \beta)^2)$ (that is $\neg(\alpha \cdot \beta \cdot \alpha \cdot \beta) \vee ((\alpha \wedge \beta) \rightarrow (\alpha \cdot \beta \cdot \alpha \cdot \beta))$) is in the class \mathcal{P}_3 . The equivalent rule generated by the algorithm in 4 is

$$\frac{\{G | \overline{\Gamma}_1^2, \overline{\Gamma}_i^2, \overline{\Sigma} \Rightarrow \overline{\Pi}\}_{1 \leq i \leq 5} \quad \{G | \overline{\Gamma}_i^2, \overline{\Gamma}_{i+1}^2, \overline{\Sigma} \Rightarrow \overline{\Pi}\}_{2 \leq i \leq 4} \quad G | \overline{\Gamma}_2^2, \overline{\Gamma}_5^2, \overline{\Sigma} \Rightarrow \overline{\Pi}}{G | \overline{\Gamma}_2, \overline{\Gamma}_3, \overline{\Gamma}_4, \overline{\Gamma}_5 \Rightarrow | \overline{\Gamma}_1, \overline{\Sigma} \Rightarrow \overline{\Pi}} \text{ (wnm)}^2$$

³ An explicit description of the axioms in these classes is given in Appendix.

where the notation X^k within inference rules stands for X, \dots, X k times, with $k \geq 0$. For further examples see Tables 2 and 3.

2.1 Density and Convergent Rules

The density rule was introduced by Takeuti and Titani in their axiomatization of first-order Gödel logic [17]. In hypersequent calculi (an instance of) this rule has the form:

$$\frac{G' \mid \Sigma, p \Rightarrow \Pi \mid \Lambda \Rightarrow p}{G' \mid \Sigma, \Lambda \Rightarrow \Pi} (D)$$

where p is a propositional variable not occurring in Σ, Λ, Π or G' (p is an *eigen-variable*). Note that adding the density rule to a hypersequent calculus can have a dramatic effect. Consider, e.g., $HMTL + (em)$, i.e., $HMTL$ extended with the rule (em) (cf. Table 3): by adding (D) we are able to prove the empty sequent as follows:

$$\frac{\frac{\overline{p \Rightarrow p} \text{ (init)}}{p \Rightarrow \mid \Rightarrow p} \text{ (em)}}{\Rightarrow} \text{ (D)}$$

A similar situation arises for $HMTL + (bc2) + (c)$. This is no surprise since, as shown in [13], the addition and subsequent elimination of (D) from an extension of $HMTL$ leads to rational completeness for the formalized logic, and the two calculi above formalize logics that are not rational complete: classical and 3-valued Gödel logic (see Table 2).

However, for many extensions of $HMTL$, adding (D) has no effect on which hypersequents are derivable: applications of (D) can be *eliminated* from derivations. Below we identify properties that, when satisfied by hypersequent rules generated using the algorithm in [4], ensure density-elimination of the corresponding extensions of $HMTL$. Rules satisfying these properties will be called *convergent*.

Given a sequent S henceforth we will denote by $L(S)$ its left hand side and by $R(S)$ its right hand side. Let $S := \Gamma_1, \Gamma_2 \Rightarrow \Pi$, we indicate by $S[\Gamma_1/\Lambda]^l[\Pi/\Sigma \Rightarrow \Psi]^r$ the sequent $\Lambda, \Gamma_2, \Sigma \Rightarrow \Psi$. The notations apply also to metasequents, i.e., sequents built from metavariables.

In what follows we will refer to any hypersequent rule generated by the procedure in [4] as *completed*.

Definition 2. Let (r) be a completed hypersequent rule:

$$\frac{G \mid S_1 \quad \dots \quad G \mid S_m}{G \mid C_1 \mid \dots \mid C_q}$$

Let $G \mid S_i$ and $G \mid S_j$ be among its premises.

(0-pivot) $G \mid S_i$ is a 0-pivot if there is an $s \in \{1, \dots, q\}$ such that $R(S_i) = R(C_s)$ and the different metavariables in $L(S_i)$ are contained in those of $L(C_s)$.

(*n*-pivot) $G|S_j$ is an *n*-pivot for $G|S_i$, for $n > 0$, if the following conditions hold:

- $G|S_j$ is a 0-pivot
- $R(S_i) = R(S_j)$
- $L(S_j) = L(S_i[\overline{T}_1/\overline{\Delta}_1, \dots, \overline{T}_n/\overline{\Delta}_n]^l)$ for $\overline{T}_1, \dots, \overline{T}_n \in L(S_i)$ and $\overline{\Delta}_1, \dots, \overline{\Delta}_n \in L(S_j)$
- If $n > 1$, $G|S_j$ is a ($n-1$)-pivot for n premises $G|S_{j_1} \dots G|S_{j_n}$, and $L(S_j) = L(S_{j_i}[\overline{T}_1/\overline{\Delta}_1, \dots, \overline{T}_{i-1}/\overline{\Delta}_{i-1}, \overline{T}_{i+1}/\overline{\Delta}_{i+1}, \dots, \overline{T}_n/\overline{\Delta}_n]^l)$ for $\overline{T}_1, \dots, \overline{T}_{i-1}, \overline{T}_{i+1}, \dots, \overline{T}_n \in L(S_{j_i})$, $\overline{\Delta}_1, \dots, \overline{\Delta}_{i-1}, \overline{\Delta}_{i+1}, \dots, \overline{\Delta}_n \in L(S_{j_i})$ and $i = 1, \dots, n$.

Definition 3. A completed hypersequent rule (r) is convergent if for each premise $G|S_i$ one of the following conditions holds: (1) $R(S_i) = \emptyset$, (2) $G|S_i$ is a 0-pivot, or (3) there is a premise $G|S_j$ which is an *n*-pivot for $G|S_i$, with $n > 0$.

Intuitively, the conclusion of a convergent rule results from a “minimal interplay” among its premises. Indeed for a premise $G|S_i$, in which $R(S_i)$ is not empty, two cases can arise: either the metavariables contained in it are already present in a component of the rule’s conclusion, or there is a premise $G|S_j$ having this property and which allows us to obtain $G|S_i$ by suitable replacements of the metavariables.

Example 2. Convergent rules are all internal structural rules, (*wm*), (*lq*) and (*wm*)^{*n*}, see Table 3. For instance, in the particular case of (*wm*)² (cf. Ex. 1):

- All different metavariables in the premise $P_1 = G|\overline{T}_1^2, \overline{T}_1^2, \overline{\Sigma} \Rightarrow \overline{\Pi}$ are contained in the component $\overline{T}_1, \overline{\Sigma} \Rightarrow \overline{\Pi}$ of the conclusion. Therefore, P_1 is a 0-pivot.
- The premise P_1 is a 1-pivot for all premises $G|\overline{T}_1^2, \overline{T}_i^2, \overline{\Sigma} \Rightarrow \overline{\Pi}$, $2 \leq i \leq 5$ as they differ from P_1 only by one metavariable.
- P_1 is a 2-pivot for the remaining premises of (*wm*)².

Completed rules that are not convergent are (*em*) and (*bc2*).

AxiomCalc. The procedure in [4] to transform axioms into equivalent analytic (hyper)sequent rules is implemented in the PROLOG-system *AxiomCalc*. It takes as input any axiom provided by the user, indicates the class \mathcal{N}_n or \mathcal{P}_n to which the axiom belongs and, for axioms within $\mathcal{N}_2, \mathcal{P}_2$ and \mathcal{P}_3 , it generates (a paper that contains) the equivalent (hyper)sequent rules. Finally, the system checks whether the generated rules are convergent.

3 Sufficient Conditions for Density Elimination

In this section we prove that *HMTL* extended with any set of convergent rules admits density-elimination. Our proof uses and refines the method in [6] of density elimination by substitutions, which is outlined below: Let d be a sub-derivation ending in the following uppermost application of density

$$\frac{\vdots d' \quad G' \mid \Sigma, p \Rightarrow \Pi \mid \Lambda \Rightarrow p}{G' \mid \Sigma, \Lambda \Rightarrow \Pi} \text{ (D)}$$

(D) is removed by substituting the occurrences of p in d in an “asymmetric” way, according to whether p occurs in the left or in the right hand side of a sequent. More precisely, each component S of a hypersequent in d is replaced by $S[p/\Lambda]^l[p/\Sigma \Rightarrow \Pi]^r$. This way, the application of (D) above is simply replaced by (EC).

A problem: the resulting labeled tree, denoted by d^* , is in general not a correct derivation anymore. The reason being the presence in d of external structural rules different from (EW) and (EC), that, by mixing the content of various conclusion components, might lead to p -axioms in their premises, i.e. to hypersequents of the form $G \mid \Theta, p^k \Rightarrow p$ derivable from axioms simply using weakenings; the problem is that the asymmetric substitution on a p -axiom leads, e.g., to $G \mid \Theta, \Lambda^k, \Sigma \Rightarrow \Pi$, which is no longer derivable in the same way.

The proof in [6]: Density-elimination was proved for calculi containing only (EC), (EW) and (com) as external structural rules. The *only* problematic case was when in d one of the premises of (com) led to a p -axiom. This case was handled by *removing* in d^* this application of (com) and replacing it with a suitable (sub)derivation starting from the other premise.

The addition of convergent rules: We show below that, though a convergent rule (r) can manipulate more hypersequent components at once and hence might create p -axioms, (r) behaves well with respect to the asymmetric substitutions. Indeed, intuitively, if one or more premises of (r) led in d to a p -axiom, say $G \mid \Theta, p^k \Rightarrow p$, the special premises of (r) called pivot are used to derive their substituted version $G \mid \Theta, \Lambda^k, \Sigma \Rightarrow \Pi$ which allow us to correctly apply (r).

The *length* $|d|$ of a derivation d is, as usual, the (maximal number of inference rules) + 1, occurring on any branch. A (D)-free derivation is a derivation not containing the (D) rule. The following lemma, which allows us to suitably “move” multisets of formulas between components, is the key for our main proof.

Lemma 1. *Given HMTL + R, with R any set of convergent rules.*

1. Any derivation d of H can be transformed into a derivation of $H[p/\alpha]^l[p/\Rightarrow \alpha]^r$, for any formula α and propositional variable p .
2. Let d' and d_1 be derivations of $G' \mid \Sigma, p \Rightarrow \Pi \mid \Lambda \Rightarrow p$ ($p \notin G', \Sigma, \Pi, \Lambda$) and $G' \mid \Theta, \Delta \Rightarrow \Psi$. We can find a derivation of $G' \mid \Theta, \Lambda \Rightarrow \Psi \mid \Sigma, \Delta \Rightarrow \Pi$.

Proof. 1. Just replace p in d everywhere with α . The claim is proved by induction on the length of the resulting derivation, as convergent rules are completed (and hence substitutive, cf. the definition and the analogous lemma in [6]).

2. By 1. and d' we have a derivation d_2 of $G' \mid \Sigma, \odot \Delta \Rightarrow \Pi \mid \Lambda \Rightarrow \odot \Delta$ where $\odot \Delta$ stands for the multiplicative conjunction \cdot of the formulas in Δ (note that $p \notin G', \Sigma, \Pi, \Lambda$). The desired derivation follows by applying (Cut) between $G' \mid \Theta, \Lambda \Rightarrow \Psi \mid \Delta \Rightarrow \odot \Delta$ and the end hypersequent of

$$\frac{\frac{\frac{\vdots d_2}{G'|\Sigma, \odot\Delta \Rightarrow \Pi|\Lambda \Rightarrow \odot\Delta} \quad \frac{\frac{\vdots d_1}{G'|\Theta, \Delta \Rightarrow \Psi}}{G'|\Theta, \odot\Delta \Rightarrow \Psi|\Sigma, \odot\Delta \Rightarrow \Pi} \quad (\cdot l) + (EW)}{G'|\Theta, \Lambda \Rightarrow \Psi|\Sigma, \odot\Delta \Rightarrow \Pi} \quad (Cut)}$$

We are now ready for the main theorem. Henceforth we denote by S_i^* the sequent $S_i[p/\Lambda]^l[p/\Sigma \Rightarrow \Pi]^r$, and by G^*, H^* , the hypersequents G, H , where the same substitution is applied to each one of their components.

Theorem 1 (Density Elimination). *HMTL extended with any set R of convergent rules admits density elimination*

Proof. To perform density elimination, it is sufficient to remove topmost applications of (D) . Take the derivation d above in $HMTL + (D) + R$, ending in an application of (D) and let d' be the (D) -free derivation ending in

$$G'|\Sigma, p \Rightarrow \Pi|\Lambda \Rightarrow p$$

As convergent rules are particular completed rules, they preserve cut-elimination when added to $HMTL$. Hence we assume that d' is cut-free.

Claim: For each hypersequent H in d' that is not a p -axiom, one can find a (D) -free derivation of $G'|H^*$.

The result on density elimination follows from this claim. Just let H be $G'|\Lambda \Rightarrow p|\Sigma, p \Rightarrow \Pi$. We get that $G'|G'|\Lambda, \Sigma \Rightarrow \Pi|\Lambda, \Sigma \Rightarrow \Pi$ is derivable (note that $(G')^* = G'$ by the eigenvariable condition on p). The desired proof of $G'|\Lambda, \Sigma \Rightarrow \Pi$ follows by multiple applications of (EC) .

The proof of the claim proceeds by induction on the length of the cut-free subderivation d_H of H in $HMTL + R$. We distinguish cases according to the last rule (r) applied in d_H . The cases $|d_H| = 0$, or when (r) is (EC) or (EW) are easy. The proof for logical rules and for (com) proceeds as in [6].

Convergent rules: Assume that (r) is a convergent rule of the form

$$\frac{G|S_1 \quad \dots \quad G|S_m}{G|C_1 \quad \dots \quad |C_q} \quad (r)$$

and that the conclusion of (r) contains no p -axiom. We show how to find a derivation of

$$G'|G^*|C_1^*|\dots|C_q^*.$$

Take a premise $G|S_i$. If $G|S_i$ is not a p -axiom, the inductive hypothesis gives us a derivation of $G'|G^*|S_i^*$. Note that this is always the case when $R(S_i) = \emptyset$, and when $G|S_i$ is a 0-pivot as in the latter case the metavariables instantiated to obtain S_i are all included in one component of the conclusion. Thus, if $G|S_i$ was a p -axiom, the conclusion would be a p -axiom as well, thus contradicting the assumption.

Assume now that $G|S_i$ is a p-axiom. We show below that we can always obtain a (D) -free derivation of

$$G' | G^* | S_i^* | C_s^*$$

for some $s \in \{1, \dots, q\}$. Being (r) convergent, there is an n -pivot premise $G|S_j$ for $G|S_i$. We show how to use $G|S_j$ to obtain the required derivation.

- Case $n = 1$: There is a 1-pivot premise $G|S_j$ for $G|S_i$, i.e., (the metasequent leading to) S_i differs only in 1 metavariable from (that of) S_j . By Definition 2, $G|S_j$ is also a 0-pivot and hence it is not a p-axiom. Let $G|S_i$ and $G|S_j$ be obtained as instantiations of the following premises⁴ of (r) :

$$S_i \text{ is obtained from } \overline{\Theta}, \overline{\Gamma}^k, \overline{\Delta}^l \Rightarrow \overline{\Pi} \text{ and } S_j \text{ from } \overline{\Theta}, \overline{\Delta}^{k+l} \Rightarrow \overline{\Pi}$$

As $G|S_i$ is a p-axiom and $G|S_j$ is not, only $\overline{\Gamma}$ can be instantiated with a propositional variable p . The most general case is when $\overline{\Gamma}$ is instantiated by Γ, p^n , the metavariable $\overline{\Delta}$ by the multiset Δ , $\overline{\Theta}$ by Θ and $\overline{\Pi}$ by Π . Hence

$$G|S_i \text{ is } G|\Theta, \Gamma^k, \Delta^l, p^{mk} \Rightarrow p \text{ and } G|S_j \text{ is } G|\Theta, \Delta^{k+l} \Rightarrow p.$$

As $G|S_j$ is not a p-axiom, by the inductive hypothesis we have a derivation for $G'|G^*|\Theta, \Delta^{k+l}, \Sigma \Rightarrow \Pi$. Using the derivation d' of $G' | \Sigma, p \Rightarrow \Pi | \Lambda \Rightarrow p$, by k applications of Lemma 1.2 with (EW) and (EC) we get

$$G' | G^* | \Theta, \Delta^k, \Delta^l, \Sigma \Rightarrow \Pi | \Sigma, \Delta \Rightarrow \Pi$$

Now, by multiple applications of internal weakenings (wl) , we have

$$G' | G^* | \Theta, \Gamma^k, \Delta^{mk}, \Delta^l, \Sigma \Rightarrow \Pi | \Sigma, \Delta \Rightarrow \Pi$$

From further repeated applications of (wl) on the fourth component, we finally obtain $G'|G^*|S_i^*|C_s^*$, where C_s stands for the component of the conclusion to which all the metavariables in (the metasequent leading to) S_j belong.

- Assume that there is an n -pivot premise $G|S_j$ for $G|S_i$, with $n > 1$. By Def. 2 (the metasequent leading to) $G|S_i$ differs from (that of) $G|S_j$ by n metavariables and there exist n other premises for which $G|S_j$ is an $(n-1)$ -pivot. As in the previous case, let S_i and S_j be obtained respectively as instantiations of the following premises of (r)

$$\overline{\Theta}, \overline{\Gamma}_1^{k_1}, \dots, \overline{\Gamma}_n^{k_n}, \overline{\Delta}_1^{l_1}, \dots, \overline{\Delta}_n^{l_n} \Rightarrow \overline{\Psi} \text{ and } \overline{\Theta}, \overline{\Delta}_1^{k_1+l_1}, \dots, \overline{\Delta}_n^{k_n+l_n} \Rightarrow \overline{\Psi}$$

Assume w.l.o.g. that $G|S_j$ is:

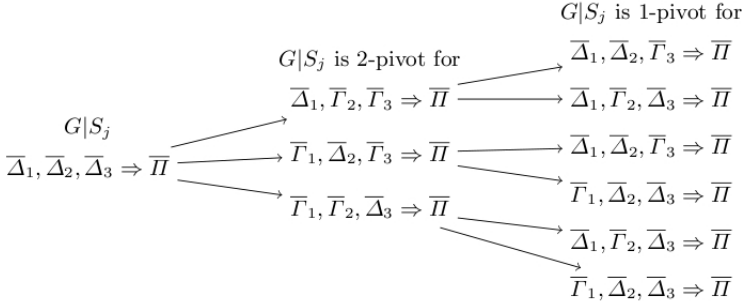
$$G|\Theta, \Delta_1^{k_1+l_1}, \dots, \Delta_n^{k_n+l_n} \Rightarrow p$$

Two cases have to be considered, according to the possible instantiations of the metavariables $\overline{\Gamma}_i$ with the propositional variable p in $G|S_i$:

⁴ To simplify the notation $\overline{\Theta}$ stands for all the metavariables in common, except $\overline{\Delta}$.

- (i) In $G|S_i$ all the metavariables \overline{T}_i are instantiated with a multiset Γ_i together with at least one occurrence of p . Then we repeatedly apply Lemma [2](#) to d' and $G|S_j$ together with (EW) and (EC) to replace $\Delta_1, \dots, \Delta_n$ with Λ in S_j , respectively k_1, \dots, k_n times. This way we get $G|\Theta, \Lambda^{k_1}, \dots, \Lambda^{k_n}, \Delta_1^{l_1}, \dots, \Delta_n^{l_n}, \Sigma \Rightarrow \Pi|\Delta_1, \Sigma \Rightarrow \Pi|\dots|\Delta_n, \Sigma \Rightarrow \Pi$. The desired hypersequent $G' | G^* | S_i^* | C_s^*$ follows by suitable applications of (wl) and (EC) (as in the 1-pivot case, C_s stands for the component of the conclusion to which all the metavariables in S_j belong).
- (ii) In $G|S_i$ all the metavariables \overline{T}_i are instantiated with Γ_i, p^{m_i} and $m_i > 0$ only for r of them ($1 \leq r < n$). (Note that in this case Lemma [2](#) would replace each metavariable Δ_i with Λ , leading to at least n occurrences of Λ , and $n > r$). The idea here is to find another premise of (r) which is not a p-axiom, for which suitable applications of Lemma [2](#) do the job. The existence of (at least one) such a premise is guaranteed by the notion of n -pivot.

We first illustrate the way we proceed with an example for $n = 3$. Assume that S_i arises as an instantiation of $\overline{T}_1, \overline{T}_2, \overline{T}_3 \Rightarrow \overline{\Pi}$ and S_j as an instantiation of $\overline{\Delta}_1, \overline{\Delta}_2, \overline{\Delta}_3 \Rightarrow \overline{\Pi}$ ($G|S_j$ is a 3-pivot for $G|S_i$). By definition of 3-pivot, there exist 3 premises in (r) for which $G|S_j$ is a 2-pivot. For each of these premises, there exist 2 premises in (r) for which $G|S_j$ is a 1-pivot. In the figure below we show how all these premises are related w.r.t the metavariables they instantiate.



(Case $r = 1$) If only 1 metavariable, say \overline{T}_1 , is instantiated in S_i with Γ_1, p we need to find a corresponding premise which will not contain a p-axiom, i.e., that does not contain \overline{T}_1 . The first occurrence of such a premise is among the premises that have $G|S_j$ as a 2-pivot, that is $\overline{\Delta}_1, \overline{T}_2, \overline{T}_3 \Rightarrow \overline{\Pi}$.

(Case $r = 2$) Assume now that 2 metavariables, say $\overline{T}_1, \overline{T}_2$, are instantiated with Γ_1, p and Γ_2, p , respectively. Again, we need to find a corresponding premise that is not a p-axiom. In this case, the set of premises that have $G|S_j$ as a 2-pivot does not suffice because each of them contains either \overline{T}_1 or \overline{T}_2 . The first occurrence of a premise that is not a p-axiom is among the premises that have $G|S_j$ as a 1-pivot, i.e., $\overline{\Delta}_1, \overline{\Delta}_2, \overline{T}_3 \Rightarrow \overline{\Pi}$.

In general, we can eventually find a premise that is not a p-axiom among those that have $G|S_j$ as $(n-r)$ -pivot. Assume for the general case that the occurrences of p in $G|S_i$ are related to the instantiation of r different metavariables, w.l.o.g $\overline{\Gamma}_1, \dots, \overline{\Gamma}_r$, i.e., S_i is

$$\Theta, \Gamma_1^{k_1}, \dots, \Gamma_n^{k_n}, \Delta_1^{l_1}, \dots, \Delta_n^{l_n}, p^{m_1 k_1 + \dots + m_r k_r} \Rightarrow p \quad \text{with } m_1, \dots, m_r > 0$$

Then we can find premises for which $G|S_j$ is an $(n-r)$ -pivot; (the metasequent leading to) those premises differ from that of $G|S_i$ in r metavariables. (At least) one of these premises will not be a p-axiom, and hence it will have the form:

$$G | \Theta, \Delta_1^{k_1+l_1}, \dots, \Delta_r^{k_r+l_r}, \Gamma_{r+1}^{k_{r+1}}, \Delta_{r+1}^{l_{r+1}}, \dots, \Gamma_n^{k_n}, \Delta_n^{l_n} \Rightarrow p$$

By inductive hypothesis we have a derivation of:

$$G' | G^* | \Theta, \Delta_1^{k_1+l_1}, \dots, \Delta_r^{k_r+l_r}, \Gamma_{r+1}^{k_{r+1}}, \Delta_{r+1}^{l_{r+1}}, \dots, \Gamma_n^{k_n}, \Delta_n^{l_n}, \Sigma \Rightarrow \Pi$$

Then we repeatedly apply Lemma [11.2](#) together with *(EW)* and *(EC)* to replace $\Delta_1, \dots, \Delta_r$ with Λ , respectively k_1, \dots, k_r times. After suitable applications of internal weakening and further external contractions, we finally get

$$G' | G^* | S_i^* | C_s^*.$$

In summary, when the last rule in d_H is convergent, for each premise $G|S_i$ we have:

- If $G|S_i$ does not contain any p-axiom, $G'|G^*|S_i^*$ is (D)-free derivable.
- If $G|S_i$ contains a p-axiom, then $G'|G^*|S_i^*|C_s^*$ is (D)-free derivable.

The required derivation of $G'|G^*|C_1^*|\dots|C_q^*$ follows by *(r)* and subsequent applications of *(EC)*, if needed. This completes the proof of the main claim.

4 From Density Elimination to Standard Completeness

Theorem [11](#) together with the results in [\[3, 13\]](#) lead to standard completeness for any logic **L** extending **MTL** with any set \mathcal{A} of axioms having equivalent convergent rules. (For all concepts of universal algebra below we refer to [\[3, 9, 13\]](#)).

As shown in [\[13\]](#), density elimination is indeed a uniform method to establish rational completeness for any extension of **MTL**. From Theorem [11](#) we can therefore state the following: let \mathcal{L} be an *MTL*-algebra (see [\[8\]](#) and steps 1-4 in the introduction) satisfying the equations \mathcal{A}^* corresponding to the axioms in \mathcal{A}

A formula α is satisfied in each dense \mathcal{L} -chain $\Leftrightarrow \alpha$ is derivable in **MTL** + \mathcal{A} .

Standard completeness is then achieved through so called *Dedekind Mac-Neille completion*, which generalizes to various ordered algebraic structures Dedekind's embedding of the rational numbers into the reals.

It is shown, e.g. in [13], that the Dedekind-McNeille completion of a dense *MTL*-chain is still a dense *MTL*-chain (in other words, it is preserved by Dedekind-MacNeille-completion). It is easy to see that the results in [3] on the preservation of equations by this completion hold for the equations \mathcal{A}^* when restricting to *MTL*-chains. Hence the Dedekind-MacNeille completion of a dense \mathcal{L} -chain is still a dense \mathcal{L} -chain, and in addition it is order-isomorphic to $[0, 1]$. This leads to *standard completeness* for the logic **L**.

Remark. Our sufficient condition for density-elimination can be easily extended to first-order calculi, hence leading, by the results in [6], to standard completeness proofs for axiomatic extensions of first-order **MTL**. In contrast, convergency of rules might be too weak to ensure density-elimination for hypersequent calculi not containing (wl) and (wr) . For these calculi, formalizing logics which extend Uninorm Logic **UL** [13], only calculi-tailored proofs of density-elimination are available; the proof in [6] applies indeed to very few of them, namely, those extending the calculus for **UL** only with additional internal structural rules having a very simple structure (e.g. contraction (c) is not one of them).

4.1 A Case Study

As a corollary of our results follows that the family of logics obtained by extending **MTL** with the axioms $(wnm)^n: \neg(\alpha \cdot \beta)^n \vee ((\alpha \wedge \beta)^{n-1} \rightarrow (\alpha \cdot \beta)^n)$, for $n \geq 2$ (here x^n stands for $x \cdots x$, n times) are standard complete and hence they are fuzzy logics in the sense of [11]. This new family of logics, discovered by playing with *AxiomCalc*, contains infinitely many different logics. This can be easily seen by noticing that $(wnm)^n$ is valid in the m -valued logic of Łukasiewicz if and only if $m \leq n + 1$.

Acknowledgments. Work supported by the FWF START Y544-N23 and the WWTF project MA07-016.

References

1. Avron, A.: The method of hypersequents in the proof theory of propositional non-classical logics. In: Hodges, W., et al. (eds.) *Logic: From Foundations to Applications*. Proc. Logic Colloquium, Keele, UK, pp. 1–32 (1993, 1996)
2. Baaz, M., Zach, R.: Hypersequents and the Proof Theory of Intuitionistic Fuzzy Logic. In: Clote, P.G., Schwichtenberg, H. (eds.) *CSL 2000*. LNCS, vol. 1862, pp. 187–201. Springer, Heidelberg (2000)
3. Ciabattoni, A., Galatos, N., Terui, K.: MacNeille Completions of FL-algebras. *Algebra Universalis* 66(4), 405–420 (2011)
4. Ciabattoni, A., Galatos, N., Terui, K.: From axioms to analytic rules in nonclassical logics. In: *Proceedings of LICS 2008*, pp. 229–240 (2008)
5. Ciabattoni, A., Esteva, F., Godo, L.: T-norm based logics with n -contraction. *Neural Network World* 12(5), 441–453 (2002)

6. Ciabattoni, A., Metcalfe, G.: Density elimination. *Theor. Comput. Sci.* 403(2-3), 328–346 (2008)
7. Esteva, F., Gispert, J., Godo, L., Montagna, F.: On the Standard and Rational Completeness of some Axiomatic Extensions of the Monoidal T-norm Logic. *Studia Logica* 71(2), 199–226 (2002)
8. Esteva, F., Godo, L.: Monoidal t-norm based logic: towards a logic for left-continuous t-norms. *Fuzzy Sets and Systems* 124, 271–288 (2001)
9. Galatos, N., Jipsen, P., Kowalski, T., Ono, H.: Residuated Lattices: an algebraic glimpse at substructural logics. *Studies in Logics and the Foundations of Mathematics*. Elsevier (2007)
10. Jenei, S., Montagna, F.: A proof of standard completeness for Esteva and Godo’s MTL logic. *Studia Logica* 70(2), 183–192 (2002)
11. Hájek, P.: *Metamathematics of Fuzzy Logic*. Kluwer (1998)
12. Horčík, R.: Alternative Proof of Standard Completeness Theorem for MTL. *Soft Computing* 11(2) (2006)
13. Metcalfe, G., Montagna, F.: Substructural fuzzy logics. *Journal of Symbolic Logic* 7(3), 834–864 (2007)
14. Metcalfe, G., Olivetti, N., Gabbay, D.: *Proof Theory for Fuzzy Logics*. Springer Series in Applied Logic, vol. 36 (2008)
15. Montagna, F., Ono, H.: Kripke semantics, undecidability and standard completeness for Esteva and Godo’s logic $MTL\forall$. *Studia Logica* 71(2), 227–245 (2002)
16. Ono, H., Komori, K.: Logics without the contraction rule. *Journal of Symbolic Logic* 50, 169–201 (1985)
17. Takeuti, G., Titani, T.: Intuitionistic fuzzy logic and intuitionistic fuzzy set theory. *Journal of Symbolic Logic* 49(3), 851–866 (1984)

A Appendix

The normal form of axioms within the classes \mathcal{N}_2 , \mathcal{P}_2 and \mathcal{P}_3 is the following.

\mathcal{N}_2 : Axioms have the form $\bigwedge_{1 \leq i \leq n} \delta_i$, in which every δ_i is a $\alpha_1 \cdots \alpha_m \rightarrow \beta$ where:

- $\beta = 0$ or $\beta_1 \vee \cdots \vee \beta_k$ and each β_l is a multiplicative conjunction of propositional variables and
- each α_i is of the form $\bigwedge_{1 \leq j \leq p} \gamma_i^j \rightarrow \beta_i^j$ where
 - $\beta_i^j = 0$ or a propositional variable, and
 - γ_i^j is a multiplicative conjunction or a disjunction of propositional variables (or 1).

\mathcal{P}_2 : Axioms have the form $\bigvee_{1 \leq i \leq n} \delta_i$, where each δ_i is of the form

$\bigwedge_{1 \leq j \leq m} \alpha_j \rightarrow \beta_j$ or $\bigwedge_{1 \leq j \leq m} \alpha_j$ where:

- each α_j is a multiplicative conjunction or disjunction of propositional variables and 1, and
- $\beta_j = 0$ or a propositional variable.

\mathcal{P}_3 : Axioms have the form $\delta_1 \vee \cdots \vee \delta_n$, where each δ_i is in \mathcal{N}_2 .

The Logic of Justified Belief Change, Soft Evidence and Defeasible Knowledge

Alexandru Baltag, Bryan Renne*, and Sonja Smets**

University of Amsterdam
Institute for Logic, Language and Computation

Abstract. We present a logic for reasoning about the evidence-based knowledge and beliefs and the evidential dynamics of non-logically-omniscient agents. We do this by adapting key tools and techniques from Dynamic Epistemic Logic, Justification Logic, and Belief Revision so as to provide a lightweight, yet fine-grained approach that characterizes well-known epistemic and doxastic attitudes in terms of the evidential reasoning that justifies these attitudes. We then add the dynamic operations of evidence introduction, evidence-based inference, strong acceptance of new evidence (evidential “upgrade”), and irrevocable acceptance of additional evidence (evidential “update”). We exemplify our theory by providing a formal dynamic account of Lehrer’s well-known Gettier-type scenario involving the famous Ferrari and the infamous Messrs. Nogot and Havit.

Keywords: Dynamic Epistemic Logic, Justification Logic, Belief Revision.

1 Introduction

As shown by the famous Gettier counterexamples [8], “knowledge” cannot simply be equated with “justified true belief.” But what is the missing ingredient in this old Platonic equation? While epistemologists have proposed different answers to fill the gap, all would agree that not just any justification will do in order to turn an item of true belief into knowledge. It is essential that “knowledge” comes equipped with a *correct*, or “good,” justification. Taking this insight as our starting point, we offer in this paper a new formalization for a plethora of notions ranging from justified belief to defeasible knowledge, each of which comes with its own justification based on how well an agent’s evidence supports her epistemic attitude.

* Funded by an Innovational Research Incentives Scheme Veni grant from the Netherlands Organisation for Scientific Research (NWO).

** Funded in part by an Innovational Research Incentives Scheme Vidi grant from the Netherlands Organisation for Scientific Research (NWO) and by the European Research Council under the European Community’s Seventh Framework Programme (FP7/2007-2013) / ERC Grant agreement no. 283963.

The so-called Defeasibility Theory defines “knowledge” as *true justified belief that is stable under belief revision with any new evidence*: “if a person has knowledge, then that person’s justification must be sufficiently strong that it is not capable of being defeated by evidence that he does not possess” (Pappas and Swain [13]). One of the problems is interpreting what “evidence” means in this context. One possible interpretation, considered by at least one author [15], takes “evidence” to mean “any proposition,” meaning we include possible *misinformation*: “real knowledge” should be robust even in the face of false evidence. This interpretation corresponds to our “infallible knowledge” modality K , which could be called “absolutely unrevisable belief.” This is a fully introspective type of knowledge, satisfying all the laws of the modal system S5.

However, the most common interpretation of Defeasibility Theory is to take it as requiring persistence of belief only in the face of “any true information.” The resulting notion of “knowledge” was formalized by Stalnaker in [17], and defined there as follows: “an agent knows that φ if and only if φ is true, she believes that φ , and she continues to believe φ if any true information is received”. This interpretation corresponds to our “defeasible knowledge” modality \square , which is positively (but not necessarily negatively) introspective, satisfying all the axioms of the modal system S4.

In [6], two of the authors of this paper studied these two notions in detail, using a dynamic logic of belief change to make precise the sense in which these modal operators match the above-mentioned characterizations in terms of their potential (in)defeasibility. However, both the above notions of “knowledge” suffer from the problem of logical omniscience. So at best they can be taken to capture some kind of *implicit, or potential, knowledge*. Moreover, Lehrer’s conception [10, 11] of defeasible knowledge is more sophisticated: he requires, not only that the belief itself be stable in the face of any true evidence, but also that the *justification* supporting this belief be similarly stable.

In this paper, we formalize the *explicit defeasible knowledge* that can be actually possessed by a (non-logically omniscient) agent. For this, we develop a version of Justification Logic (JL), in the tradition of [2], with the new feature that it borrows concepts from Belief Revision theory to deal with “soft” (fallible) evidence. Furthermore, we combine this approach with ideas and techniques from Dynamic Epistemic Logic (DEL) [4–6, 20], including important ideas from the temporal DEL literature [14, 16, 22], obtaining a Dynamic Justification Logic that can deal with justified belief change and soft evidential dynamics.

Thus, in essence we bring together the work of two traditions in Logic (DEL and JL), while using models coming from a third tradition (Belief Revision theory). The added value comes from the interplay of these settings, which in particular allows us to capture several of the subtle distinctions made in [10, 12], pointing to scenarios in which an agent has a justified true belief but no good evidence to turn his belief into knowledge. We formalize various types of epistemic-evidential actions, and we use them to give dynamic characterizations of explicit “knowledge” (in both its defeasible and its infallible versions).

We provide complete, decidable proof systems for these logics, and apply them to the analysis of one of the well-known Gettier-type counterexamples in the literature.

The approaches in the available literature that are closest to our work are Artemov’s paper [1] on the Gettier problem, and the work of van Benthem and Velázquez-Quesada [19, 21] on the dynamics of evidence. In our last section we make a more detailed comparison between these papers and ours. For now, it suffices to say that our solution is closely related to these approaches and was in fact inspired by them, but it is nevertheless original and avoids some of the problems encountered in these works.

2 Belief, Justification, Awareness, and Knowledge

2.1 Syntax

Definition 1 (Language). *Given a set Φ of atomic sentences, the language $\mathcal{L} := (\mathcal{T}, \mathcal{F})$ consists of the set \mathcal{T} of evidence terms t and the set \mathcal{F} of propositional formulas (sentences) φ defined by the following double recursion:*

$$\begin{aligned} \varphi &::= \perp \mid p \mid \neg\varphi \mid \varphi \wedge \varphi \mid Et \mid t \gg \varphi \mid \Box\varphi \mid K\varphi \mid Y\varphi \quad \text{with } p \in \Phi \\ t &::= c_\varphi \mid t \cdot t \mid t + t \end{aligned}$$

Notation: let \diamond denote $\neg\Box\neg$, let \hat{K} denote $\neg K\neg$, and let \hat{Y} denote $\neg Y\neg$. The set $\text{sub}(t)$ of subterms of a term t is defined by induction on the construction of t as follows: $\text{sub}(c_\varphi) = \{c_\varphi\}$, $\text{sub}(s \cdot u) = \{s \cdot u\} \cup \text{sub}(s) \cup \text{sub}(u)$, $\text{sub}(s + u) = \{s + u\} \cup \text{sub}(s) \cup \text{sub}(u)$. The set $\text{sub}(\varphi)$ of subformulas of a formula φ is defined by induction on the construction of φ as follows: $\text{sub}(\perp) := \{\perp\}$, $\text{sub}(p) := \{p\}$, $\text{sub}(\neg\theta) := \{\neg\theta\} \cup \text{sub}(\theta)$, $\text{sub}(\theta \wedge \theta') := \{\theta \wedge \theta'\} \cup \text{sub}(\theta) \cup \text{sub}(\theta')$, $\text{sub}(Et) := \{Et\}$, $\text{sub}(t \gg \theta) := \{t \gg \theta\}$, $\text{sub}(\Box\theta) := \{\Box\theta\} \cup \text{sub}(\theta)$, $\text{sub}(K\theta) := \{K\theta\} \cup \text{sub}(\theta)$, and $\text{sub}(Y\theta) := \{Y\theta\} \cup \text{sub}(\theta)$. We define an operation $(\cdot)^Y : \mathcal{T} \cup \mathcal{F} \rightarrow \mathcal{T} \cup \mathcal{F}$ by setting: $(c_\varphi)^Y := c_{(\varphi^Y)}$, $(t \cdot s)^Y := t^Y \cdot s^Y$, and $(t + s)^Y := t^Y + s^Y$ for terms; and $\perp^Y := \perp$, $p^Y := p$, $(\neg\varphi)^Y := \neg\varphi^Y$, $(\varphi \wedge \psi)^Y := \varphi^Y \wedge \psi^Y$, $(Et)^Y := Et^Y$, $(t \gg \varphi)^Y := t^Y \gg \varphi^Y$, $(K\varphi)^Y := YK\varphi$, $(\Box\varphi)^Y := Y\Box\varphi$, $(Y\varphi)^Y := YY\varphi$.

Et says that *evidence t is available* to the agent (though not necessarily accepted by her). $t \gg \varphi$ says that t is *admissible evidence* for φ : if accepted, this evidence supports φ . $\Box\varphi$ says that *the agent (implicitly) defeasibly knows φ* . $K\varphi$ says that *the agent (implicitly) infallibly knows φ* . $Y\varphi$ says that “*yesterday*” (i.e., before the last epistemic action) φ was true. c_φ is an *evidential certificate*: a “canonical” piece of evidence in support of sentence φ . $t \cdot s$ is a compound evidence, obtained by combining (using Modus Ponens) the two pieces of evidence t and s . Finally, $t + s$ is a body of evidence that aggregates (without performing logical inference) all the evidence provided by t and s ; $t \cdot s$ therefore supports both the statements supported by t and those supported by s .

By “defeasible knowledge” \Box we mean here knowledge in the sense of the Defeasibility Theory: justified true belief that cannot be defeated by any new *true* information that the agent might receive. By “infallible” knowledge K we

mean “absolutely certain,” absolutely unrevisable, fully introspective knowledge: belief that cannot fail to be true, and so it cannot be defeated by any new information (including false testimony). We will later show that our formal operators match these characterizations. Note that both these notions are forms of *implicit* knowledge. We will later introduce the corresponding types of *explicit* knowledge.

Definition 2 (Admissibility). Admissibility is the smallest binary relation $\gg \subseteq \mathcal{T} \times \mathcal{F}$ satisfying the following conditions: (1) $c_\varphi \gg \varphi$; (2) if $t \gg (\psi \Rightarrow \varphi)$ and $s \gg \psi$, then $(t \cdot s) \gg \varphi$; (3) if $t \gg \varphi$ or $s \gg \varphi$, then $(t + s) \gg \varphi$. Note that admissibility is both a syntactic meta-relation and a symbol in the language. It will be clear from context which is which.

Lemma 1 (Temporal Admissibility). $t \gg \varphi$ implies $t^Y \gg \varphi^Y$.

Proof. By induction on the construction of t . □

Lemma 2 (Computability of Admissibility). The map $t \mapsto \{\varphi \mid t \gg \varphi\}$ of type $\mathcal{T} \rightarrow \wp(\mathcal{F})$ is computable, and for every t the set $\{\varphi \mid t \gg \varphi\}$ is finite.

Proof. The map is recursive, and the finiteness of $\{\varphi \mid t \gg \varphi\}$ can be proved by induction on the complexity of terms. □

Definition 3. $\mathcal{T}^e := \{t \in \mathcal{T} \mid \exists \varphi : t \gg \varphi\}$ is the set of admissible terms.

Definition 4 (Propositional Content). For every term $t \in \mathcal{T}$, we define the propositional content con_t of t as the conjunction of all the formulas for which t is admissible evidence: $\text{con}_t := \bigwedge \{\theta \mid t \gg \theta\}$. For $t \notin \mathcal{T}^e$, this is the conjunction of an empty set of formulas, so in this case (if we interpret \bigwedge as infimum in the complete Boolean algebra of propositions) we get tautologically true content: $\text{con}_t = \top := \neg \perp$.

Definition 5 (Implicit Belief, Implicit Acceptance, Implicit Evidence). We introduce the following abbreviations for the language \mathcal{L} :

$B\varphi := \diamond \square \varphi$ says that the agent (implicitly) believes φ ,
 $A(t) := \bigwedge_{c_\varphi \in \text{sub}(t)} B\varphi$ says that the agent (implicitly) accepts evidence t ,
 $G(t) := \bigwedge_{c_\varphi \in \text{sub}(t)} \square \varphi$ says that t is good (implicit) evidence,
 $I(t) := \bigwedge_{c_\varphi \in \text{sub}(t)} K\varphi$ says that t is infallible (implicit) evidence, and
 $t : \varphi := A(t) \wedge t \gg \varphi$ says that t is (implicit) evidence for belief of φ .

Like the implicit knowledge notions K and \square , implicit belief suffers from logical omniscience. We now introduce the corresponding explicit notions, which reflect the beliefs, knowledge and justifications that are actually possessed by a (non-logically-omniscient) agent.

Definition 6 (Explicit Belief and Knowledge). We introduce the following additional abbreviations for the language \mathcal{L} :

$B^e \varphi := B\varphi \wedge Ec_\varphi$ says that the agent explicitly believes φ ,
 $\square^e \varphi := \square \varphi \wedge Ec_\varphi$ says that the agent explicitly defeasibly knows φ ,
 $K^e \varphi := K\varphi \wedge Ec_\varphi$ says that the agent explicitly infallibly knows φ , and
 $t :^e \varphi := t : \varphi \wedge Et$ says that t is explicit evidence for belief of φ .

2.2 Semantics

Definition 7 (Model). A model $M = (W, \llbracket \cdot \rrbracket, \sim, \geq, \rightsquigarrow, E)$ is a structure consisting of a nonempty set W of possible worlds; a valuation map $\llbracket \cdot \rrbracket : \Phi \rightarrow \wp(W)$; binary relations \sim , \geq , and \rightsquigarrow on W , with \sim (“epistemically indistinguishable from”) representing epistemic possibility/indistinguishability, \geq (“no more plausible than”) representing relative plausibility, and \rightsquigarrow (“is the temporal predecessor of”) representing immediate temporal precedence (going forward in time from a moment to the next moment); as well as an evidence map $E : W \rightarrow \wp(\mathcal{T})$; all satisfying the following conditions:

- \sim is an equivalence relation and \geq is a preorder \square
- Indefeasibility: $w \geq v \Rightarrow w \sim v$.
- Local Connectedness: $w \sim v \Rightarrow (w \geq v \vee v \geq w)$.
- Propositional Perfect Recall: $(w \rightsquigarrow v \sim v') \Rightarrow \exists w'(w \sim w' \rightsquigarrow v')$.
- Evidential Perfect Recall: $w \rightsquigarrow w' \Rightarrow \{t^Y \mid t \in E(w)\} \subseteq E(w')$.
- Uniqueness of Past: $(w' \rightsquigarrow w \wedge w'' \rightsquigarrow w) \Rightarrow w' = w''$.
- Persistence of Facts: $w \rightsquigarrow w' \Rightarrow (w \in \llbracket p \rrbracket \Leftrightarrow w' \in \llbracket p \rrbracket)$.
- (Implicit) Evidential Introspection: $w \sim v \Rightarrow E(w) = E(v)$.
- Subterm Closure: If $t \cdot t' \in E(w)$ or $t + t' \in E(w)$, then $t \in E(w)$ and $t' \in E(w)$.
This says that a compound evidence is actually available to the agent only if its component pieces of evidence are available.
- Certification of Evidence: If $t \in E(w)$ and $t \gg \varphi$, then $c_\varphi \in E(w)$.
This says that every actual evidence in support of a sentence φ can be converted into a certificate of correctness: a canonical piece of evidence c_φ that certifies it. All explicit knowledge can be certified.

A pointed model is a pair (M, w) consisting of a model M and a designated world w in M called the “actual world.”

Many authors in the Belief Revision literature require their models to satisfy some version of the following requirement:

Definition 8. The Best Worlds Assumption applies to a model iff for every non-empty set $P \subseteq W$ of indistinguishable worlds (i.e., such that $w \sim w'$ for all $w, w' \in P$), the set

$$\min P := \{w \in P \mid w' \geq w \text{ for all } w' \in P\}$$

(consisting of the most plausible worlds in P) is also non-empty.

The Best Worlds Assumption is useful, since it allows for a very natural and intuitive definition of (conditional) belief $B(\varphi|P)$. Some authors (e.g., Grove [\[9\]](#)) weaken this condition to cover only the sets P that are *definable* by some sentence ψ in their language: this is indeed enough to define syntactical conditional belief operators $B(\varphi|\psi)$. However, in this paper, we will consider an even stronger condition, called *standardness*:

¹ A *preorder* is a reflexive and transitive binary relation. For a preorder \geq , we denote by $>$ the strict version given by $t > s := (s \geq t) \wedge (t \not\geq s)$. We denote by \leq and $<$ the converse relations.

Definition 9 (Standard Model). A model $M = (W, [\cdot], \sim, \geq, \rightsquigarrow, E)$ is said to be standard if both the strict converse-plausibility relation $<$ and the immediate temporal predecessor relation \rightsquigarrow are well-founded. This means that there are no infinite chains $w_0 > w_1 > w_2 > \dots$ of more and more plausible worlds, and there are no infinite chains $w_0 \rightsquigarrow w_1 \rightsquigarrow w_2 \rightsquigarrow \dots$ going back in time. Observe that well-foundedness implies acyclicity, so in a standard model there are no temporal loops. Note also that the well-foundedness of \rightsquigarrow together with the Propositional Perfect Recall condition imply “temporal perfect recall”: $w \rightsquigarrow v$ implies $w \not\sim v$. Similarly, note that every standard model satisfies the Best Worlds Assumption.²

Definition 10 (Truth). We now define a satisfaction relation $(M, w) \models \varphi$ between pointed models (M, w) and formulas $\varphi \in \mathcal{F}$. We also denote $(M, w) \models \varphi$ in the more familiar way by $w \models_M \varphi$, omitting the subscript M when M is fixed.

$$\begin{aligned}
 w &\not\models \perp \\
 w &\models p \quad \text{iff } w \in [p] \\
 w &\models \neg\varphi \quad \text{iff } w \not\models \varphi \\
 w &\models \varphi \wedge \psi \quad \text{iff } w \models \varphi \text{ and } w \models \psi \\
 w &\models Et \quad \text{iff } t \in E(w) \\
 w &\models t \gg \varphi \quad \text{iff } t \gg \varphi \\
 w &\models \Box\varphi \quad \text{iff } v \models \varphi \text{ for every } v \leq w \\
 w &\models K\varphi \quad \text{iff } v \models \varphi \text{ for every } v \sim w \\
 w &\models Y\varphi \quad \text{iff } v \models \varphi \text{ for every } v \rightsquigarrow w
 \end{aligned}$$

Given a model $M = (W, [\cdot], \sim, \geq, \rightsquigarrow, E)$, we can extend the valuation map $[\cdot]$ to all sentences, by putting $[\varphi] = \{w \in W \mid w \models \varphi\}$. Validity $\models \varphi$ means that $(M, w) \models \varphi$ for every standard pointed model (M, w) .

The following result shows that belief, as defined above, fits with its most widely accepted definition in standard models:

Lemma 3. In a standard model $M = (W, [\cdot], \sim, \geq, \rightsquigarrow, E)$, “belief” is the same as “truth in the most plausible worlds”:

$$w \models_M B\varphi \quad \text{iff} \quad w' \models_M \varphi \text{ for all } w' \in \min\{w' \in W \mid w \sim w'\} .$$

2.3 Example of the Gettier Problem

The following example of a Gettier problem is adapted from [12]. Our (unnamed) agent (Lehrer’s “Claimant,” who we assume to be a woman) is the professor of a class consisting of two students, Mr. Nogot and Mr. Havit. Let us denote by p the sentence “Mr. Nogot owns a Ferrari” and by q the sentence “Mr. Havit owns a Ferrari.”

Mr. Nogot tells our agent that he owns a Ferrari and shows her the title papers and a picture of him driving a Ferrari. This testimonial evidence supports sentence p , and so it is admissible for p ; hence, we will denote this evidence by c_p .

² Indeed, it is easy to see that this condition follows from the well-foundedness of $<$ together with the above Local Connectedness assumption.

The evidence term c_p is thus *available* to our agent (since it was made available to her by Mr. Nogot). Let us further assume that this evidence is *accepted* by her: on the basis of c_p , she believes p (i.e., that Mr. Nogot owns a Ferrari). Moreover, let us assume that this belief is actually false: in fact, Mr. Nogot does not own a Ferrari, he just lied to our agent, forged the car title and faked the picture (using Photoshop to edit himself into the driver’s seat). Furthermore, let us assume that, unknown to our agent, Mr. Havit actually does own a Ferrari.

Based on her available accepted evidence c_p and using propositional inference, our agent concludes that *some student in her class owns a Ferrari* ($p \vee q$). This belief is *true* (since in fact Mr. Havit owns one), it is *justified* (given Mr. Nogot’s testimony and the rules of logic), but it is *not “knowledge”* in Lehrer’s sense. Of course, $p \vee q$ is not infallible knowledge (in the absolutely certain sense captured by the operator K above), since the agent possesses no “hard” evidence for $p \vee q$. Indeed, testimonial evidence is “soft”: the fact that Mr. Nogot claims that he owns a Ferrari is still consistent with the possibility that nobody in the class owns any car. Moreover, our agent’s justified belief in $p \vee q$ is easily *defeasible, even by true evidence*: if in the future she would (correctly) learn that Mr. Nogot does not in fact own a Ferrari ($\neg p$), then she would be forced to drop her (correct) belief in $p \vee q$. Hence, this true justified belief is not “knowledge,” even in the fallible, defeasible sense, captured in our formalism by the operator. \square

Here is a simple model of the epistemic situation described in this story:

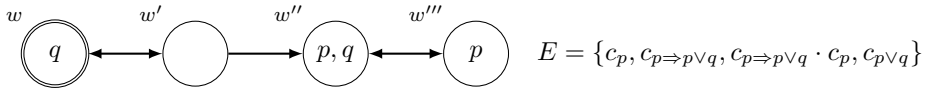


Fig. 1. The Nogot-Havit scenario

The set of possible worlds is $W = \{w, w', w'', w'''\}$ and the valuation is $\llbracket p \rrbracket = \{w'', w'''\}$, $\llbracket q \rrbracket = \{w, w''\}$. In Figure 1, we represent each possible world by a circle (labeled with the name of the world and encompassing the atomic sentences true at that world). The double-circled world indicates the real world or current state of affairs (in which q is true and p is false; i.e., Mr. Havit has a Ferrari and Mr. Nogot does not). We represent the plausibility relations \geq by horizontal arrows (pointing from a world w to all the worlds $v \leq w$ that are at least as plausible as w), but we omit the arrows that can be obtained by reflexivity (looping) and transitivity (arrow composition). The one-way arrow from w' to w'' (and the one-way arrows, obtained by transitivity, from w to both w'' and w''' and from w' to w''') show the p -worlds are more plausible than the $\neg p$ -worlds. As a consequence, the agent implicitly believes p (since p is true in all the most plausible worlds w'' and w'''). The epistemic indistinguishability relation is not directly represented but can be recovered by closing the horizontal arrows under transitivity, reflexivity and *symmetry*. So here all the four worlds are epistemically indistinguishable, which expresses the fact that the agent has no

hard evidence concerning p and q , and thus she has no infallible knowledge (K) concerning their truth values: all four Boolean combinations are epistemically possible (in the sense of K). The available evidence is the same at all four worlds (in agreement with the condition of Implicit Evidential Introspection) and consists of Nogot’s testimonial evidence c_p , logical evidence $c_{p \Rightarrow p \vee q}$ supporting the axiom $p \Rightarrow p \vee q$, inferential evidence $c_{p \Rightarrow p \vee q} \cdot c_p$ (obtained by combining the previous two in accordance with Modus Ponens) supporting $p \vee q$, and a certificate $c_{p \vee q}$ confirming that she derived $p \vee q$. According to our definitions, the agent has (both implicit and) explicit true justified belief in $p \vee q$: the sentence $(p \vee q) \wedge B^e(p \vee q) \wedge (c_{p \Rightarrow p \vee q} \cdot c_p) : (p \vee q)$ holds at the real world w . However, she does not have (either implicit or explicit) knowledge of $p \vee q$ (either in the infallible sense of K or in the defeasible sense of \square).

In this model, all evidence is accepted: $c_{p \Rightarrow p \vee q}$ is infallibly so (since the agent has implicit infallible knowledge of axioms) and c_p is incorrectly accepted (since the agent implicitly believes p but p is in fact false). But this is *not* a general requirement in our setting: availability does not imply acceptance. Indeed, mutually inconsistent evidence terms might be available (in the sense that the agent is aware of them, can compute them or is considering them), while in our models, belief is always consistent. For instance, our agent may be aware of some very weak evidence *against* p , say the fact (denoted by $c_{\neg p}$) that she never actually saw Mr. Nogot in a Ferrari, but she might choose to reject such evidence. In this case, she still keeps the same (implicit and explicit) beliefs as in the above story, as illustrated by the following model:

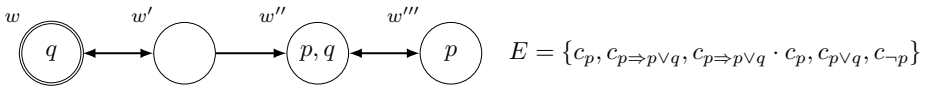


Fig. 2. The Nogot-Havit scenario with additional evidence $c_{\neg p}$

Note that in both the above models, the agent has *no explicit introspection* about her beliefs or about her justifications! She simply does not consider such issues. If we want to model a situation in which the agent uses introspection to become aware of her explicit belief in p , then we obtain the model in Figure 3.

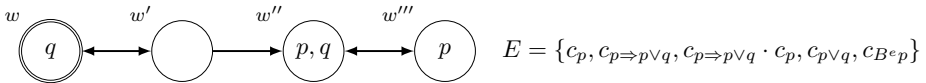


Fig. 3. The Nogot-Havit scenario with introspection of belief

The fact that there are no \rightsquigarrow -arrows in any of these diagrams simply expresses the fact that we chose the current moment as the starting point (moment 0) in our story. Of course, a more accurate representation would include the *history*

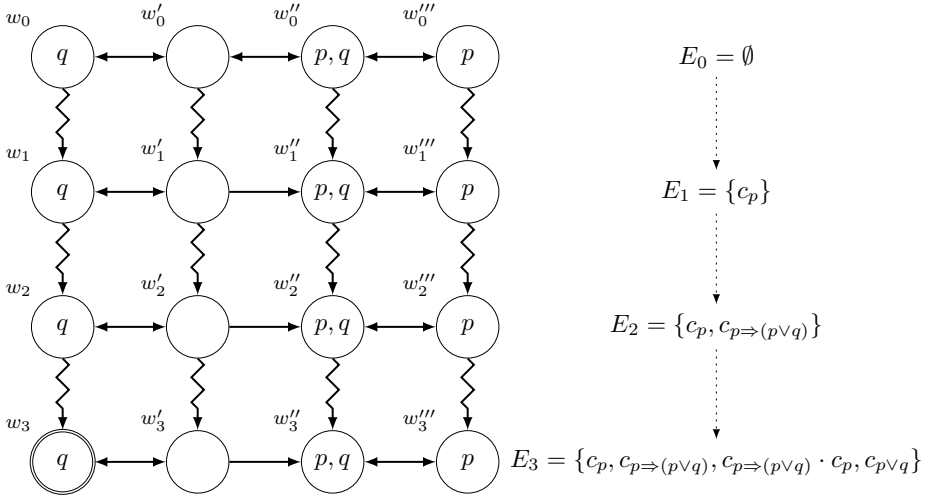


Fig. 4. Temporal development leading to the Nogot-Havit scenario

of how our agent came to believe $p \vee q$. A possible such history could be given by the model in Figure 4.

The real world at the current moment (previously denoted by w) is now denoted by w_3 . The vertical arrows represent the immediate temporal precedence relation, going from one moment to the next moment. So the time flows downward in this diagram. According to this history, originally (at the “true” initial moment w_0) the agent had no evidence ($E = \emptyset$), and no non-trivial beliefs about p and q , so she considered all four Boolean combinations to be equally plausible. After this, she received and accepted Mr. Nogot’s testimonial evidence c_p , leading her to explicitly believe p (at moment w_1), and thus implicitly (but not yet explicitly!) believe $p \vee q$. At the next moment w_2 , she thought about the logical axiom $p \Rightarrow p \vee q$, became aware of its applicability to this particular instance, and so the infallible evidence $c_{p \Rightarrow p \vee q}$ became available to her. She then used Modus Ponens, thereby computing the evidence term $c_{p \Rightarrow p \vee q} \cdot c_p$ that supports the conclusion $p \vee q$, certified this derivation by adding $c_{p \vee q}$ to her evidence set, and therefore acquired an explicit belief in $p \vee q$ (at the current moment w_3).

2.4 Proof System

Definition 11 (Theory). JB, the theory of justified belief, is defined in Table 7.

Lemma 4 (Derivable Principles). We have the following.

1. *Application for Admissibility:* $\vdash (s \gg \varphi) \Rightarrow (t \gg (\varphi \Rightarrow \psi) \Leftrightarrow (t \cdot s) \gg \psi)$.
2. *Application:* $\vdash (s \gg \varphi) \Rightarrow (t : (\varphi \Rightarrow \psi) \wedge s : \varphi \Leftrightarrow (t \cdot s) : \psi)$.
3. *Weakening of Justified Belief:* $\vdash t : \varphi \Rightarrow B\varphi$.
4. *Certification of Implicit Belief:* $\vdash B\varphi \Rightarrow c_\varphi : \varphi$.

Table 1. The theory JB

AXIOM SCHEMES	
Classical Logic: Axioms of Classical Propositional Logic	
Knowledge of Available Evidence:	$\vdash Et \Rightarrow KEt$
Subterm Closure:	$\vdash E(t \cdot s) \Rightarrow Et \wedge Es$ $\vdash E(t + s) \Rightarrow Et \wedge Es$
Certification of Evidence:	$\vdash t \gg \varphi \wedge Et \Rightarrow Ec_\varphi$
Admissibility:	$\vdash t \gg \varphi$ whenever $t \gg \varphi$ $\vdash \neg(t \gg \varphi)$ whenever $t \not\gg \varphi$
Infallible Knowledge:	S5 axioms for K
Defeasible Knowledge:	S4 axioms for \square
Indefeasibility:	$\vdash K\varphi \Rightarrow \square\varphi$
Local Connectedness:	$\vdash K(\varphi \vee \square\psi) \wedge K(\psi \vee \square\varphi) \Rightarrow (K\varphi \vee K\psi)$
Normality of Y :	$\vdash Y(\varphi \Rightarrow \psi) \Rightarrow (Y\varphi \Rightarrow Y\psi)$
Propositional Perfect Recall:	$\vdash YK\varphi \Rightarrow KY\varphi$
Evidential Perfect Recall:	$\vdash YEt \wedge \neg Y\perp \Rightarrow Et^Y$
Uniqueness of Past:	$\vdash \neg Y\varphi \Rightarrow Y\neg\varphi$
Persistence of Facts:	$\vdash Yp \Leftrightarrow (\neg Y\perp \Rightarrow p)$

RULES

$$\frac{\varphi \Rightarrow \psi \quad \varphi}{\psi} \text{ (MP)} \quad \frac{\varphi}{\square\varphi} \text{ (\square N)} \quad \frac{\varphi}{K\varphi} \text{ (KN)} \quad \frac{\varphi}{Y\varphi} \text{ (YN)}$$

5. *Weakening of Justified Defeasible Knowledge:* $\vdash (t \gg \varphi) \wedge G(t) \Rightarrow \square\varphi$.
6. *Certification of Defeasible Knowledge:* $\vdash \square\varphi \Rightarrow (c_\varphi \gg \varphi) \wedge G(c_\varphi)$.
7. *Weakening of Justified Infallible Knowledge:* $\vdash (t \gg \varphi) \wedge I(t) \Rightarrow K\varphi$.
8. *Certification of Infallible Knowledge:* $\vdash K\varphi \Rightarrow (c_\varphi \gg \varphi) \wedge I(c_\varphi)$.

Proof. (1) follows by the definition of admissibility. (2) follows by the definition of $u:\theta$, (1), and classical reasoning. (4) follows by the definition of $A(c_\varphi)$. (6) follows by the definition of $G(c_\varphi)$. (8) follows by the definition of $I(c_\varphi)$.

Recalling that $u:\theta = A(u) \wedge u \gg \theta$ and $A(u) = \bigwedge_{c_\theta \in \text{sub}(u)} B\theta$, the proof of (3) is by induction on the construction of t . Base case: $\vdash c_\varphi:\varphi \Rightarrow B\varphi$ follows by the definition of $A(c_\varphi)$ and classical reasoning. Induction step: assuming $\vdash s_i:\theta \Rightarrow B\theta$ and for each $i \in \{1, 2\}$ and $\theta \in \mathcal{F}$, we wish to show that $\vdash (s_1 \cdot s_2):\varphi \Rightarrow B\varphi$. Let $S := \{\psi \mid s_1 \gg (\psi \Rightarrow \varphi) \wedge s_2 \gg \psi\}$. If $S = \emptyset$, then $\vdash \neg(s_1 \cdot s_2) \gg \varphi$ and so we have $\vdash (s_1 \cdot s_2):\varphi \Leftrightarrow \perp$ and hence $\vdash (s_1 \cdot s_2):\varphi \Rightarrow B\varphi$. So let us assume that $S \neq \emptyset$. We then have by classical reasoning that $\vdash (s_1 \cdot s_2):\varphi \Leftrightarrow s_1:(\psi \Rightarrow \varphi) \wedge s_2:\psi$ for an arbitrarily selected $\psi \in S \neq \emptyset$. By the induction hypothesis and classical reasoning, we then have $\vdash (s_1 \cdot s_2):\varphi \Rightarrow B(\psi \Rightarrow \varphi) \wedge B\psi$. Applying modal reasoning, it follows that $\vdash (s_1 \cdot s_2):\varphi \Rightarrow B\varphi$.

Recalling that $G(u) = \bigwedge_{c_\theta \in \text{sub}(u)} G\theta$, the proof of (5) is by induction on the construction of t . Base case: $t = c_\varphi$ and the result follows by the definition of $G(c_\varphi)$ and classical reasoning. Induction step: assuming $\vdash (s_i \gg \theta) \wedge G(s_i) \Rightarrow \square\varphi$ for each $i \in \{1, 2\}$ and $\theta \in \mathcal{F}$, we wish to show that $\vdash (s_1 \cdot s_2) \gg \varphi \wedge G(s_1 \cdot s_2) \Rightarrow \square\varphi$.

We define $S := \{\psi \mid s_1 \gg (\psi \Rightarrow \varphi) \wedge s_2 \gg \psi\}$. If $S = \emptyset$, then $\vdash \neg(s_1 \cdot s_2) \gg \varphi$ and hence $\vdash (s_1 \cdot s_2) \gg \varphi \wedge G(s_1 \cdot s_2) \Leftrightarrow \perp$, from which the result follows by classical propositional reasoning. So let us assume that $S \neq \emptyset$. Choosing an arbitrary $\psi \in S$, we have $\vdash (s_1 \cdot s_2) \gg \varphi \wedge G(s_1 \cdot s_2) \Leftrightarrow s_1 \gg (\psi \Rightarrow \varphi) \wedge s_2 \gg \psi \wedge G(s_1) \wedge G(s_2)$ by classical propositional reasoning, the definition of admissibility, and the definition of $G(u)$. But we then have $\vdash (s_1 \cdot s_2) \gg \varphi \wedge G(s_1 \cdot s_2) \Rightarrow \Box(\psi \Rightarrow \varphi) \wedge \Box\psi$ by the induction hypothesis and classical reasoning and therefore that $\vdash (s_1 \cdot s_2) \gg \varphi \wedge G(s_1 \cdot s_2) \Rightarrow \Box\varphi$ by modal reasoning.

The argument for (7) is similar to that for (5). \square

A common criticism of epistemic modal logic is that it suffers from *logical omniscience*: the agent believes all logical consequences of her beliefs, including in particular all valid formulas. But in JB, *only implicit belief* $B\varphi$ and *implicit knowledge*—either *infallible* $K\varphi$ or *defeasible* $\Box\varphi$ —*satisfies logical omniscience*. That is, $K\varphi$ (or $\Box\varphi$) says only that the agent can come to infallibly (or defeasibly) know φ *only in principle*. Implicit knowledge may be thought of as “potential knowledge” of φ that the agent might in principle obtain, though perhaps she will never have this knowledge in actuality.

Explicit knowledge $K^e\varphi$ (or $\Box^e\varphi$) is very different. This represents the agent’s *actual knowledge*, in that $K^e\varphi = K\varphi \wedge Ec_\varphi$ and $\Box^e\varphi = \Box\varphi \wedge Ec_\varphi$ say that the agent not only has the potential to realize her implicit knowledge of φ but also that *she has in fact gone through the trouble of obtaining and correctly validating the certificate of correctness c_φ for φ* (i.e., Ec_φ). Therefore, *explicit knowledge does not satisfy logical omniscience*.

Definition 12 (Iterated Axioms and Logical Terms). *An iterated axiom is a formula of the form*

$$\underbrace{X_1 X_2 X_3 \cdots X_n}_{\text{zero or more } X_i\text{'s}} \varphi,$$

where each $X_i \in \{\Box, K, Y\}$ and φ is an axiom. The set of logical terms is the smallest set that contains certificates c_φ for each iterated axiom φ and is closed under the evidence-combining operator $t \cdot s$ (for Modus Ponens).

The logical terms are those that are built by applying the inference operator \cdot to certificates of knowledge c_φ for iterated axioms φ . We may think of logical terms as the logical arguments we use to justify iterated axioms and their logical consequences. The forthcoming Theorem 11 shows that the agent can in principle always find purely logical justification to support infallible knowledge of logical truths.

Lemma 5 (Necessitation Elimination). *For each $\varphi \in \mathcal{F}$, we have $\vdash \varphi$ iff φ is provable from iterated axioms without the use of necessitation rules (i.e., KN , $\Box N$, or YN).*

Proof. By induction on the number of necessitations. \square

Theorem 1 (Theorem Internalization). *For each $\varphi \in \mathcal{F}$, we have $\vdash \varphi$ iff there exists a logical term t such that $\vdash I(t) \wedge t \gg \varphi$.*

Proof. The right-to-left direction follows by Lemma 4(7), so we focus on the left-to-right direction. We write $\vdash^* \varphi$ to mean that φ is provable from iterated axioms without the use of necessitation rules. By Lemma 5, it suffices for us to prove by induction on the proof length that $\vdash^* \varphi$ implies there is a term t such that $\vdash^* (t \gg \varphi) \wedge I(t)$. Proceeding, we recall that $I(s) = \bigwedge_{c_\psi \in \text{sub}(s)} K\psi$.

- Case: φ is an iterated axiom.
 Since φ is an iterated axiom, c_φ is a logical term and $\vdash^* K\varphi$. Hence $\vdash^* I(c_\varphi)$. Further, we have $c_\varphi \gg \varphi$ by the definition of admissibility and hence $\vdash c_\varphi \gg \varphi$. Conclusion: $\vdash^* (c_\varphi \gg \varphi) \wedge I(c_\varphi)$.
- Case: φ follows by MP from $\psi \Rightarrow \varphi$ and ψ .
 By the inner induction hypothesis, there exist logical terms t and s such that $\vdash^* t \gg (\psi \Rightarrow \varphi) \wedge I(t)$ and $\vdash^* (s \gg \psi) \wedge I(s)$. Hence $\vdash^* (t \cdot s) \gg \psi$ and $\vdash^* I(t \cdot s)$. Conclusion: $\vdash^* ((t \cdot s) \gg \varphi) \wedge I(t \cdot s)$. \square

Theorem 2 (Soundness and Completeness for Non-Standard Models). *JB is sound and strongly complete with respect to the class of all models.*

Proof. Soundness is by induction on the length of derivation. We omit the details. Completeness is by way of a canonical model construction. We define the *canonical model* $\Omega := (W^\Omega, \llbracket \cdot \rrbracket, \sim, \geq, \rightsquigarrow, E)$ by setting

$$\begin{aligned}
 W^\Omega &:= \{ \Gamma \subseteq \mathcal{F} \mid \Gamma \text{ is maximal consistent} \} , \\
 \llbracket p \rrbracket &:= \{ \Gamma \in W \mid p \in \Gamma \} , \\
 \Gamma \sim \Delta &\text{ iff } \{ \theta \mid K\theta \in \Gamma \} \subseteq \Delta , \\
 \Gamma \geq \Delta &\text{ iff } \{ \theta \mid \Box\theta \in \Gamma \} \subseteq \Delta , \\
 \Gamma \rightsquigarrow \Delta &\text{ iff } \{ \theta \mid Y\theta \in \Delta \} \subseteq \Gamma , \text{ and} \\
 E(\Gamma) &:= \{ t \in \mathcal{T} \mid Et \in \Gamma \} .
 \end{aligned}$$

It is easy to see that Ω is a model: most properties follow by standard correspondence theory [7], while the properties of Evidential Perfect Recall, Knowledge of Available Evidence, Subterm Closure, and Certification of Evidence follow by modal reasoning using axioms of the same name.

What remains is for us to prove the *Truth Lemma*: for each $\Gamma \in W^\Omega$ and each $\theta \in \mathcal{F}$, we have $\theta \in \Gamma$ iff $\Gamma \models_\Omega \theta$. The proof is by induction on the construction of θ . All steps of this induction are standard [7], except the ones referring to formulas of the form Et or $t \gg \varphi$. For formulas Et , to have $Et \in \Gamma$ is what it means to have $t \in E(\Gamma)$, which is itself equivalent to $\Gamma \models Et$ by the definition of truth. For formulas $t \gg \varphi$, it follows immediately from the Admissibility axioms (Table 1), maximal consistency, and the definition of truth that we have $(t \gg \varphi) \in \Gamma$ iff $\Gamma \models t \gg \varphi$. This completes the proof of the Truth Lemma. Strong completeness follows immediately in the usual way [7]. \square

Theorem 3 (Completeness for Standard Models, Finite Model Property). *JB is sound and weakly complete with respect to the class of standard models. Moreover, it is also weakly complete with respect to the class of finite standard models.*

Proof (Proof Sketch). First, we unravel the canonical model to Ω obtain another model $\Omega \times \mathbb{Z}$ in which \rightsquigarrow is acyclic. For this, we take copies (w, k) of each world w in Ω and each integer $k \in \mathbb{Z}$. Accordingly, the set of worlds of our new model $\Omega \times \mathbb{Z}$ will be $W^\Omega \times \mathbb{Z}$ with $(w, k) \sim (w', k')$ iff $k = k'$ and $w \sim w'$, $(w, k) \geq (w', k')$ iff $k = k'$ and $w \geq w'$, $(w, k) \rightsquigarrow (w', k')$ iff $k' = k - 1$ and $w \rightsquigarrow w'$, $E(w, k) = E(w)$, and $\llbracket p \rrbracket = \{(w, k) \mid w \in \llbracket p \rrbracket\}$. We obtain a (non-standard) model $\Omega \times \mathbb{Z}$, in which the relation \rightsquigarrow is acyclic. One can easily check (by induction on formulas) that $(w, k) \models_{\Omega \times \mathbb{Z}} \varphi$ iff $w \models_\Omega \varphi$ for each $k \in \mathbb{Z}$ and $\varphi \in \mathcal{F}$.

Fix a consistent formula ψ and a world v in Ω satisfying $v \models_\Omega \psi$ and hence $(v, 0) \models_{\Omega \times \mathbb{Z}} \psi$. Take the submodel $M := (\Omega \times \mathbb{Z})_{(v, 0)}$ generated by $(v, 0)$ via the relations \sim , \geq , and \rightsquigarrow ; i.e., M is the restriction of (the relations, functions, and valuation of) the model $\Omega \times \mathbb{Z}$ to the set

$$W^M := \{(w, k) \mid (w, k) \rightsquigarrow^* (v', 0) \text{ for some } v' \sim v\} .$$

Here we use the iterated temporal arrows \rightsquigarrow^n and \rightsquigarrow^* , which are defined inductively by setting $w \rightsquigarrow^0 w'$ iff $w = w'$, $w \rightsquigarrow^{n+1} w'$ iff there exists w'' such that $w \rightsquigarrow^n w'' \rightsquigarrow w'$, and $w \rightsquigarrow^* w'$ iff there exists some $n \in \mathbb{N}$ such that $w \rightsquigarrow^n w'$. It is easy to see that the set W^M is closed (as a submodel of $\Omega \times \mathbb{Z}$) under the relations \sim , \geq , and \rightsquigarrow . (The proof for \sim uses Propositional Perfect Recall, and the proof for \geq uses Indefeasibility and the result for \rightsquigarrow ; see Definition [9](#).) So M is indeed a generated submodel, and hence by standard results in modal logic about generated submodels [\[7\]](#), it follows that for every $(w, n) \in W^M$ and every formula φ , we have $(w, n) \models_M \varphi$ iff $w \models_\Omega \varphi$. Hence $(v, 0) \models_M \psi$. It is also easy to see that each temporal layer of this model is connected: $(w, n) \sim (w', n')$ holds in M iff $n = n'$.

Let now m be the modal Y -depth of formula ψ ; that is, m is the maximum number of nested Y -modalities occurring in ψ . For each $0 \leq n \leq m$, let $\Psi_n := \text{sub}_n(\psi)$ be the set of all subformulas of ψ of modal Y -depth less than or equal to n . We construct a new model M' by “cutting” M to depth m (i.e., deleting all worlds (w, n) having $n > m$) and applying to the n^{th} temporal layer of the resulting submodel (for each nonnegative $n \leq m$) the transitive filtration with respect to the set Ψ_{m-n} . More precisely, we define an equivalence relation \equiv on W^M by

$$(w, n) \equiv (w', n') \text{ iff } (n = n') \wedge \forall \varphi \in \Psi_{m-n} ((w, n) \models_M \varphi \Leftrightarrow (w', n') \models_M \varphi) .$$

The set W' of possible worlds in our new model M' will consist of all the \equiv -equivalence classes of worlds of depth at most m :

$$W' = \{\overline{(w, n)} \mid (w, n) \in W^M \text{ and } 0 \leq n \leq m\} ,$$

where $\overline{(w, n)} = \{(w', n') \in W^M \mid (w, n) \equiv (w', n')\}$. For $R \in \{\sim, \rightsquigarrow\}$, we take the induced relations on classes (the “smallest filtration”):

$$\overline{(w, n)}R\overline{(w', n')} \text{ iff } \exists(v, k) \in \overline{(w, n)}, \exists(v', k') \in \overline{(w', n')} : (v, k)R(v', k') .$$

In the case of \sim , this amounts to $\overline{(w, n)} \sim \overline{(w', n')}$ iff $n = n'$, and in the case of \rightsquigarrow , this boils down (with the aid of Propositional Perfect Recall) to

$$\overline{(w, n)} \rightsquigarrow \overline{(w', n')} \text{ iff } (n' = n - 1) \wedge \exists w'' (w'' \rightsquigarrow w' \wedge \forall \varphi \in \Psi_{m-n} (w \models_{\Omega} \varphi \Leftrightarrow w'' \models_{\Omega} \varphi)) .$$

For \geq , we take the transitive filtration:

$$\overline{(w, n)} \geq \overline{(w', n')} \text{ iff } (n' = n) \wedge \forall \varphi \in \Psi_{m-n} (w \models_{\Omega} \Box \varphi \Rightarrow w' \models_{\Omega} \Box \varphi) .$$

The valuation is defined as in any filtration. Formulas of the form Et and $t \gg \varphi$ are treated in the same way that filtration treats atomic formulas. (For formulas Et , this works because all worlds in M belonging to the same temporal layer n are \sim -indistinguishable and so agree on the truth values of formulas Et by Evidential Perfect Recall. For formulas $t \gg \varphi$, it works because all worlds in Ω agree on the truth values of formulas $t \gg \varphi$.)

The resulting model M' is finite and inherits all the properties of the previous models, in particular \rightsquigarrow is acyclic and $>$ is transitive and irreflexive (and hence also acyclic). Since every acyclic relation on a finite set is well-founded, M' is a standard model. Finally, it is trivial to check (by induction on n) that, for every $0 \leq n \leq m$, every world $\overline{(w, m - n)} \in W'$ and every formula $\varphi \in \Psi_n$, we have $(w, n) \models_{M'} \varphi \Leftrightarrow (w, n) \models_M \varphi$. In particular, we obtain $(v, 0) \models_{M'} \psi$. \square

Corollary 1 (Decidability). *The logic JB is decidable.*

Proof. The size of the finite model M' constructed in the above proof is bounded by $N = m \cdot 2^{|\text{sub}(\psi)|}$, where m is the modal Y -depth of ψ . Hence we can simply investigate one by one all models (up to isomorphism) of size at most N , checking whether ψ is satisfied in any of them. \square

It is common in Justification Logic to have “evidence internalization terms” $!t$ and $?t$ that allow the agent to introspectively verify his evidence or lack thereof according to the following schemes:

$$\begin{array}{l} \text{PC. } t : \varphi \Rightarrow !t : (t : \varphi) \\ \text{NC. } \neg t : \varphi \Rightarrow ?t : (\neg t : \varphi) \end{array}$$

PC (“Positive Checker”) says that if the agent has potential evidence t for φ , then she can in principle use $!t$ (pronounced “bang t ”) to check that t is indeed potential evidence for φ . NC (“Negative Checker”) says that if t is not potential evidence for φ , then the agent can in principle check this as well using $?t$. PC is typically required in order for the Theorem Internalization result to hold. However, as we saw above, positive checker is not needed to prove this result for JB. The reason is that our certificates c_{φ} allow us to recover a form of PC. In fact, certificates allow us to recover a form of NC as well. Indeed, the following schemes are derivable in our system

$$\begin{aligned} \text{PC}' . \quad & t : \varphi \Rightarrow (c_t : \varphi) : (t : \varphi) \\ \text{NC}' . \quad & \neg t : \varphi \Rightarrow (c_{\neg t} : \varphi) : (\neg t : \varphi) \end{aligned}$$

The next result is a kind of “doxastic internalization,” showing that all the implicit beliefs of a rational agent are in principle justifiable, all her explicit beliefs are explicitly justified, and all her (infallible, or at least, defeasible) knowledge can be given a correct (i.e., infallible, or at least “good”) justification.

Theorem 4 (Knowledge and Belief Internalization). *For each $\varphi \in \mathcal{F}$:*

$$\begin{aligned} w \models_M B\varphi & \text{ iff there is a term } t \text{ such that } w \models_M t : \varphi ; \\ w \models_M B^e\varphi & \text{ iff there is a term } t \text{ such that } w \models_M t :^e\varphi ; \\ w \models_M \Box\varphi & \text{ iff there is a term } t \text{ such that } w \models_M t : \varphi \wedge G(t) ; \\ w \models_M \Box^e\varphi & \text{ iff there is a term } t \text{ such that } w \models_M t :^e\varphi \wedge G(t) ; \\ w \models_M K\varphi & \text{ iff there is a term } t \text{ such that } w \models_M t : \varphi \wedge I(t) ; \\ w \models_M K^e\varphi & \text{ iff there is a term } t \text{ such that } w \models_M t :^e\varphi \wedge I(t) . \end{aligned}$$

In words: something is (implicit or explicit) belief iff it is (implicitly or explicitly) justifiable by some (implicitly or explicitly accepted) evidence; something is (implicit or explicit) defensible knowledge iff it is (implicitly or explicitly) justifiable by some good evidence; something is (implicit or explicit) infallible knowledge iff it is (implicitly or explicitly) justifiable by some infallible evidence.

Proof. The right-to-left directions of the statements about implicit knowledge and belief follow by soundness, the validity $\models t : \varphi \Rightarrow t \gg \varphi$, and Lemma 4 parts (3), (5), and (7). For the left-to-right directions of the statements about implicit knowledge and belief, take $t := c_\varphi$. We have $\models c_\varphi \gg \varphi$ by the definitions of admissibility and truth. Hence $w \models c_\varphi : \varphi$ by the assumption on the left, which implies $B\varphi = A(c_\varphi)$. In the case of the implicit knowledge statements, the additional conjunction on the right side of the “iff” follows by the assumption on the left and the fact that $G(c_\varphi) = \Box\varphi$ and $I(c_\varphi) = K\varphi$. Results for the explicit knowledge and belief statements follow by taking $t = c_\varphi$ and recalling that $\models c_\varphi \gg \varphi$. \square

3 Evidence Dynamics

In this paper we will consider only four types of epistemic actions. (1) The first type is the action $t+$ by which an evidence term t becomes available: the agent can form this term, or becomes aware of the possibility of this evidence, without necessarily accepting it. A special example of this is $c_\varphi+$, for some axiom instance φ : this represents the “logical” action of becoming aware of an axiom instance. Other examples include the actions $c_{B\varphi}+$ and $c_{\neg B\varphi}+$: these represent acts of introspection by which the agent becomes aware of some of her implicit beliefs and non-beliefs. (2) The second type is the action $t \otimes s$ by which, given previously available evidence terms t and s , the agent forms a new term $t \cdot s$ representing the logical action of performing a Modus Ponens step. (3) The third type is the action $t!$ of updating with some “hard” evidence t (coming from an absolutely

infallible source). This corresponds to the standard DEL update (all the worlds that do not fit evidence t are deleted), except that its input is a new piece of evidence t rather than a proposition. Moreover, this is an “explicit” update: the new evidence becomes *available to* (and accepted with absolute certainty by) the agent, although only in its “past” form (t^Y), since it is evidence about the world as it was *before* the update. Similarly, all the previously available evidence is still available but only its “past” form as evidence about the situation before the update.³ (4) The fourth type is the action $t\uparrow$ of upgrading with some “soft” evidence t coming from a strongly trusted (though not infallible) source. The new evidence is (strongly) accepted (although not infallibly known). Modulo the same differences as in the case of update, this is essentially an explicit version of the action called “radical upgrade” in the DEL literature and “lexicographic revision” in the Belief Revision literature. All worlds that fit the new evidence become more plausible than the worlds that do not fit it.

There are of course many other possible epistemic actions. In particular, one can define an explicit version of the action $t\uparrow$ known as “conservative upgrade” in the DEL literature and as “minimal revision” in the Belief Revision literature: only the most plausible worlds fitting the new evidence become the most plausible overall. But for simplicity, in this paper we restrict ourselves to the four types of actions mentioned above.

Definition 13 (Language with Updates). $\mathcal{L}^{act} := (\mathcal{F}^{act}, \mathcal{F}^{act})$ is the extension of the basic language $\mathcal{L} = (\mathcal{T}, \mathcal{F})$ obtained by adding modal operators $[\alpha]$ for epistemic actions $\alpha \in \{t+, t \otimes s, t!, t\uparrow\}$, for every $t, s \in \mathcal{T}$. (Note that this not only extends the set of formulas: due to our terms c_φ , it also extends the set of terms.) The notions of subterm, subformula, admissibility, and model are lifted to \mathcal{L}^{act} in the obvious way. We assign the following informal readings to the new modal formulas:

- $[t+]\varphi$ says that after making evidence t available, φ is true,
- $[t \otimes s]\varphi$ says that after combining evidence t and s (by Modus Ponens), φ is true,
- $[t!]\varphi$ says that after updating with hard evidence t , formula φ is true, and
- $[t\uparrow]\varphi$ says that after (radically) upgrading with soft evidence t , formula φ is true.

For every action α we define a sentence pre_α , called the precondition of α , and a set of terms $\mathcal{T}(\alpha)$ called the evidence set of α :

³ This is needed in order to deal with Moore sentences. However, we want to endow the agent with some basic insight of the principle that epistemic actions do not change ontic facts: she should be instantly aware of this, without having to perform additional inference steps to derive this. This explains our definition of φ^Y , which leaves unchanged all the purely propositional formulas (i.e., Boolean combinations of atoms), so that for such formulas we have $(c_\varphi)^Y = c_\varphi$. In effect, epistemic actions with factual evidence will actually produce evidence about the current world as it is after the update.

$$\begin{aligned}
\text{pre}_{t+} &:= \text{pre}_{t\uparrow} = \top \\
\text{pre}_{t!} &:= \text{con}_t = \bigwedge \{ \theta \mid t \gg \theta \} \\
\text{pre}_{t \otimes s} &:= Et \wedge Es
\end{aligned}$$

$$\begin{aligned}
\mathcal{T}(t+) = \mathcal{T}(t!) = \mathcal{T}(t\uparrow) &:= \text{sub}(t) \cup \{c_\theta \mid s \gg \theta \text{ for some } s \in \text{sub}(t)\} \\
\mathcal{T}(t \otimes s) &:= \{t \cdot s\} \cup \{c_\theta \mid t \cdot s \gg \theta\}
\end{aligned}$$

The precondition pre_α captures the condition of possibility of action α : actions $t+$ and $t\uparrow$ can always happen, $t!$ can only happen if t really is hard information (i.e., if the proposition supported by t is actually true), and $t \otimes s$ can only happen if t and s are already available. The evidence set $\mathcal{T}(\alpha)$ consists of all the evidence terms that become available due to α .

To say that a formula is reduced means that it does not contain a subformula of the form $[\alpha]\varphi$. Note that the notion of subformula is lifted to \mathcal{L}^{act} from that given in Definition 7 in the obvious way. So, for example, φ is not a subformula of $c_{[c_\varphi!]\varphi} \gg [c_\varphi!]\varphi$.

Definition 14 (Truth for \mathcal{L}^{act}). Given a model $M = (W, \llbracket \cdot \rrbracket, \sim, \geq, \rightsquigarrow, E)$, a world $w \in W$ and an epistemic action $\alpha \in \{t+, t \otimes s, t!, t\uparrow\}$, we will use the notation w^α to denote the ordered pair (w, α) , and we will use this to formally represent the “updated” world resulting from performing action α in world w . The satisfaction relation $(M, w) \models \varphi$ is defined as an extension of our previous definition of truth (Definition 10) obtained by adding the following clauses for dynamic modalities $[\alpha]\varphi$, with $\alpha \in \{t+, t \otimes s, t!, t\uparrow\}$:

$$x \models_M [\alpha]\varphi \text{ iff } x^\alpha \models_{M[\alpha]} \varphi \text{ with } M[\alpha] := (W^\alpha, \llbracket \cdot \rrbracket^\alpha, \sim^\alpha, \geq^\alpha, \rightsquigarrow^\alpha, E^\alpha)$$

and

$$\begin{aligned}
W^\alpha &:= W \cup \{w^\alpha \mid w \in \llbracket \text{pre}_\alpha \rrbracket\} \\
E^\alpha(w) &:= E(w) \text{ for } w \in W \\
E^\alpha(w^\alpha) &:= \{u^Y \mid u \in \mathcal{T}(\alpha) \cup E(w)\} \\
\llbracket p \rrbracket^\alpha &:= \llbracket p \rrbracket \cup \{w^\alpha \in W^\alpha \mid w \in \llbracket p \rrbracket\} \\
\sim^\alpha &:= \sim \cup \{(w^\alpha, v^\alpha) \mid w \sim v\} \\
\rightsquigarrow^\alpha &:= \rightsquigarrow \cup \{(w, w^\alpha) \mid w \in \llbracket \text{pre}_\alpha \rrbracket\} \\
\geq^\alpha &:= \geq \cup \{(w^\alpha, v^\alpha) \mid w \geq v\} \text{ for } \alpha \in \{t+, t \otimes s, t!\} \\
\geq^{t\uparrow} &:= \geq \cup \{(w^{t\uparrow}, v^{t\uparrow}) \mid (w \notin \llbracket \text{con}_t \rrbracket \wedge v \in \llbracket \text{con}_t \rrbracket) \vee \\
&\quad (w \notin \llbracket \text{con}_t \rrbracket \wedge w \geq v) \vee (v \in \llbracket \text{con}_t \rrbracket \wedge w \geq v)\} \\
\geq^{t!} &:= \geq \cup \{(w^{t!}, v^{t!}) \mid w \geq v\} \text{ for } t \notin \mathcal{T}^e
\end{aligned}$$

3.1 Example Continued

We now generate the temporal progression of Lehrer’s Nogot-Havit scenario using our epistemic actions. First, we begin from a situation of complete ignorance represented by the model in Figure 5.

We successively apply the following actions: first the upgrade $c_p\uparrow$, by which our agent upgrades with (i.e., accepts as soft evidence) Mr. Nogot’s false testimony; then the logical action $c_{p \rightarrow p \vee q}+$, by which she becomes aware of evidence for the axiom $p \rightarrow p \vee q$; then the action $c_{p \rightarrow p \vee q} \otimes c_p$, by which she combines

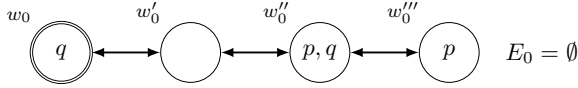


Fig. 5. Initial situation with complete ignorance

her pieces of evidence using Modus Ponens, acquiring explicit justified belief in $p \vee q$. The result of these transformations is exactly the model in Figure 4.

But now we can go one step further in the future. Suppose that the agent receives hard evidence (from an infallible source such as Lehrer’s Critic, who always tells the truth) that Mr. Nogot does not own a Ferrari. We can interpret this as an update $c_{\neg p}$, which can be applied to the model in Figure 4 to yield the model in Figure 6 below. Our agent has been “Gettierized”: some new true evidence defeated her true justified belief!

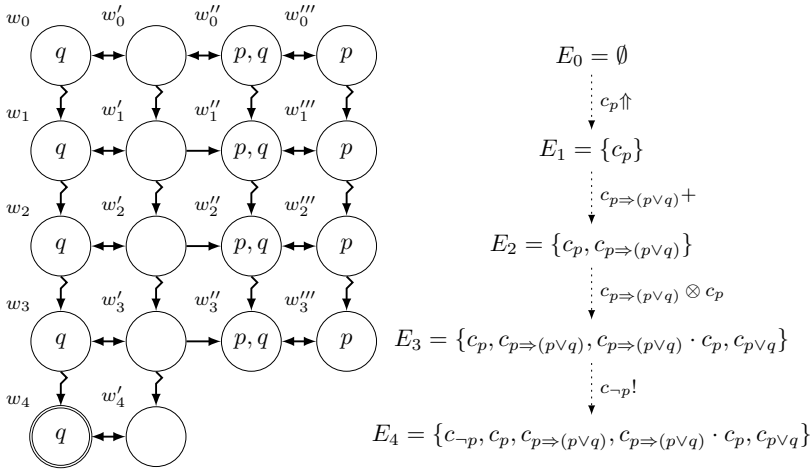


Fig. 6. “Gettierization” of the agent from the Nogot-Havit scenario

3.2 An Explicit Version of Moore Sentences

Finally, to explain the need for the Y operator and justify the way we defined evidential dynamics, let us consider the situation after the first step above, as

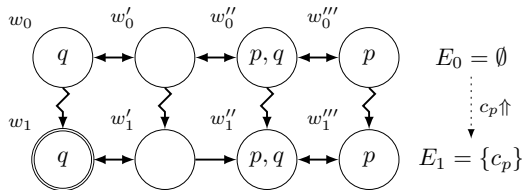


Fig. 7. After action $c_p \uparrow$

shown in Figure 7. Our agent has just received Mr. Nogot’s (lying) testimony c_p and performed an upgrade $c_p \uparrow$. Notice that now she explicitly believes p , but she is not yet aware of it: she has not yet performed any introspection acts, so she does not explicitly know that she explicitly believes p . She simply has not reflected upon her own beliefs. Now suppose the infallible Critic intervenes, giving her hard evidence $c_{\neg p}$ that $\neg p$, and, at the same time, she reflects upon her beliefs, becoming infallibly aware of her explicit belief in p , justified by her “hard” introspective evidence $c_{B^e p}$. Note that, unlike the classical examples of Moore sentences learnt by a fully introspective agent, in this case *both these pieces of hard evidence are non-redundant*: the agent explicitly learns something new from each of them. We can interpret this action as an update $(c_{\neg p} + c_{B^e p})!$, which applied to the model in Figure 7 yields the model in Figure 8. Note the new evidence set: the agent did not simply add the new term (and its subterms $c_{B^e p}$ etc.) to her set. Adding $c_{B^e p}$ would in any case be useless, since by now (in the new model) she already believes $\neg p$, so the evidence $c_{B^e p}$ would simply be rejected. But this action does give her a new, important (and correct!) introspective piece of evidence, namely $c_{YB^e p}$: she explicitly learns that she used to believe p (before the update). This is indeed new: at the time when she was holding the explicit belief in p , she was not introspective about it. Now that she is aware of this (past) belief, she does not hold it anymore!

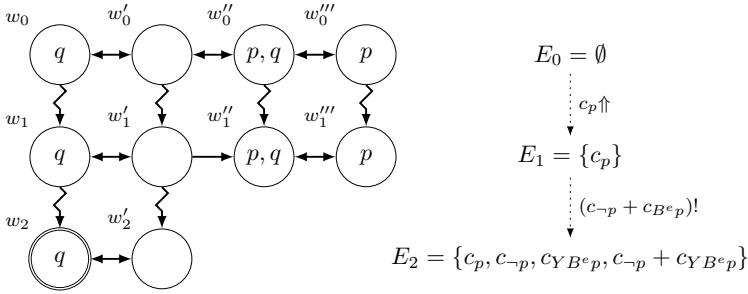


Fig. 8. After action $(c_{\neg p} + c_{B^e p})!$

3.3 Robustness of Knowledge

Using dynamics, we can now show in what sense our notions of knowledge capture their intended interpretations:

Theorem 5. Something is (explicit) defeasible knowledge iff it is (explicitly) believed to have been true no matter what new *hard* (and hence true) evidence is received:

$$\begin{aligned}
 w \models \Box \varphi & \text{ iff } w \models [t!]BY\varphi \text{ for every evidence term } t; \\
 w \models \Box^e \varphi & \text{ iff } w \models [t!]B^e Y\varphi \text{ for every evidence term } t.
 \end{aligned}$$

Something is (explicit) infallible knowledge iff it is (explicitly) believed to have been true no matter what new *soft* (possibly false) evidence is received:

$$\begin{aligned} w \models K\varphi \text{ iff } w \models [t\uparrow]BY\varphi \text{ for every evidence term } t; \\ w \models K^e\varphi \text{ iff } w \models [t\uparrow]B^eY\varphi \text{ for every evidence term } t. \end{aligned}$$

Definition 15 (Theory). DJB, the theory of dynamic justified belief, is defined in Table 2.

Lemma 6 (Reduction). For each $\varphi \in \mathcal{F}^{act}$, there is a reduced $\varphi^\dagger \in \mathcal{F}^{act}$ such that $\vdash \varphi \Leftrightarrow \varphi^\dagger$.

Theorem 6 (Soundness and Completeness). $\vdash \varphi$ iff $\models \varphi$ for each $\varphi \in \mathcal{F}^{act}$.

Table 2. The theory DJB

AXIOM SCHEMES	
JB: Axioms and rules of JB (Table 1)	
Persistence of Facts:	$[\alpha]p \Leftrightarrow (\text{pre}_\alpha \Rightarrow p)$
	Functionality: $[\alpha]\neg\varphi \Leftrightarrow (\text{pre}_\alpha \Rightarrow \neg[\alpha]\varphi)$
Conjunction Distributivity:	$[\alpha](\varphi \wedge \psi) \Leftrightarrow [\alpha]\varphi \wedge [\alpha]\psi$
Evidence Dynamics:	$[\alpha]Et^Y \quad \text{for } t \in \mathcal{T}(\alpha)$
	$[\alpha]Et^Y \Leftrightarrow (\text{pre}_\alpha \Rightarrow Et) \quad \text{for } t \notin \mathcal{T}(\alpha)$
	$[\alpha]Es \Leftrightarrow \neg\text{pre}_\alpha \quad \text{for } s \notin \{t^Y \mid t \in \mathcal{T}\}$
Admissibility Dynamics:	$[\alpha](t \gg \varphi) \Leftrightarrow (\text{pre}_\alpha \Rightarrow t \gg \varphi)$
Knowledge Dynamics:	$[\alpha]K\varphi \Leftrightarrow (\text{pre}_\alpha \Rightarrow K[\alpha]\varphi)$
	$[\alpha]\Box\varphi \Leftrightarrow (\text{pre}_\alpha \Rightarrow \Box[\alpha]\varphi) \quad \text{for } \alpha \in \{t+, t \otimes s, t!\}$
	$[t\uparrow]\Box\varphi \Leftrightarrow \Box(\neg\text{con}_t \Rightarrow [t\uparrow]\varphi) \wedge$ $(\text{con}_t \Rightarrow \Box[t\uparrow]\varphi \wedge K(\neg\text{con}_t \Rightarrow [t\uparrow]\varphi))$
Temporal Dynamics:	$[\alpha]Y\varphi \Leftrightarrow (\text{pre}_\alpha \Rightarrow \varphi)$

4 Conclusion and Comparison with Related Work

Our framework is a variant of the traditional Justification Logic semantics for justified belief and knowledge. But a key difference lies in our semantics for justified belief $t:\varphi$. According to the traditional Justification Logic semantics (the ‘‘Fitting semantics’’) [1–3], the agent has justified belief $t:\varphi$ if and only if: (1) t is admissible for φ and (2) the agent believes φ (in the sense that $B\varphi$ holds for an appropriate modal operator B). Here the notion of admissibility is weaker than ours: the traditional semantics does not require that each atomic piece of evidence is admissible for a finite number of formulas (let alone for a unique formula); further, the traditional account does not require that a compound piece of evidence $t \cdot s$ or $t + s$ be admissible for a formula if and only if the constituents t and s are admissible for certain related formulas (only the ‘‘if’’ direction is required). For example, while both the traditional account and ours agree that $(s \gg \varphi) \Rightarrow (t \gg (\varphi \Rightarrow \psi)) \Rightarrow (t \cdot s) \gg \psi$, only ours also ensures that $(s \gg \varphi) \Rightarrow ((t \cdot s) \gg \psi \Rightarrow t \gg (\varphi \Rightarrow \psi))$. It is therefore possible in the traditional semantics that a compound piece of evidence $t \cdot s$ is admissible for

a formula without its constituent pieces of evidence t and s having any relation to this formula on their own. We forbid this in our setup: admissibility always provides an “evidential chain” that links the formulas for which a compound piece of evidence is admissible to the finitely many formulas for which its constituent pieces of evidence are admissible. This evidential chain plays an important role in our semantics of (implicit) justified knowledge and belief: the agent has (implicit) justified knowledge or belief if and only if she has a piece of evidence t whose structure describes a step-by-step evidential reasoning process along a well-formed evidential chain from basic assumptions—all of which are supported by acceptable, good, or infallible certificates—to a final conclusion that is justified by the reasoning process encoded by the evidential chain. (Here the emphasis is on the “only if” direction: she cannot have justified knowledge or belief without having a proper justification!) We therefore have not only that the resulting compound piece of evidence is admissible for the conclusion φ , but that this evidence necessarily encodes a meaningful and evidentially sound argument for t (that begins with premises c_ψ whose certified ψ ’s are validated according to belief, defeasible knowledge, or infallible knowledge, and that proceeds step-wise via Modus Ponens $s_1 \cdot s_2$ and monotonic combination of evidence $s_1 + s_2$ to the eventual conclusion φ). In contrast, the traditional semantics admits the possibility that a combined piece of evidence $t \cdot s$ is admissible for an assertion φ without any evidential chain linking admissible formulas of t and of s to φ , and, further, that one or more of t or s is individually faulty (in the sense that none of the formulas for which it is admissible is believed or known). In such a circumstance, the agent has a “justified belief” in φ based on a compound piece of evidence $t \cdot s$ whose components t and s are individually unrelated to φ and are not all reliable evidence for those things for which they are admissible. We expressly forbid such a possibility in our semantics: compound pieces of evidence always provide a proper evidential chain from reliable certificates to a justified conclusion.

In our setting we consider different ways in which an agent can change his evidence and update his beliefs and knowledge: the agent can be confronted with new evidence coming from an external source, he can reach new conclusions by bringing pieces of available evidence together or he can become aware of how to perform a specific inference on evidence. Related work in the Dynamic Epistemic Logic literature can be found in [19, 21], where van Benthem and Velázquez-Quesada provide a logical system that can handle information changes that include an agent’s acts of inference. These authors start from a particular type of awareness logic and enhance it with the dynamic features that are common in Dynamic Epistemic Logic. Their focus lies on how implicit and explicit knowledge can be related, specifically addressing the question “what do agents have to do to make their implicit knowledge explicit?” [19]. In [21], their work was extended to consider implicit and explicit belief and belief revision. If we compare the “evidence sets” in our models with van Benthem and Velázquez-Quesada’s “awareness sets,” we see that their setting can be thought of as a special case of ours. Indeed as noted in [21], their setting is the special

case in which each formula can have one unique justification, namely the formula itself. Another important difference between the two approaches points to how one can deal with the synchronization between explicit and implicit knowledge under epistemic actions that change the model at hand. Indeed as noted also in [19], an update with epistemic higher-order information (i.e., information that refers to the knowledge of beliefs of the agent) can easily push the explicit and implicit knowledge of the agent “out of sync.” An example is our explicit version of a Moore-sentence–type announcement in Figure 8. The evidential updates defined by van Benthem and Velázquez-Quesada are simply adding the new evidence to the awareness set (or, in our setting, to the evidence set). But in the case of a Moore announcement, this would add evidence that is already obsolete! Van Benthem and Velázquez-Quesada attempt to solve this problem by proposing in [19, 21] a different definition of “explicit knowledge,” which in our setting would correspond to $K^e\varphi = K(\varphi \wedge Ec_\varphi)$. In our view, this variant definition, although interesting, is not enough to solve the synchronization problem because it loses track of the past. According to us, this problem simply cannot be solved if one does not keep track of the past: our solution is to enhance our models with a temporal operator that allows us to refer back to the previous state before the epistemic action happened. Hence all new evidence that becomes available to the agent comes in its “past” form: it always is evidence about the world as it was before the learning or reasoning action took place. In the case of purely propositional information, this will still be valid evidence about the current state of the world. But in the case of doxastic information, it will simply be (new!) evidence about the agent’s past beliefs: a previously non-introspective agent becomes aware of some of her past beliefs, even if in the meantime she already changed them!

Overall, our explicit treatment of justification, awareness, dynamics and epistemics tackles several of the issues that have been left open in the work of [19]. In future work we aim to combine our approach with the semantic treatment of “evidence” in [18], where van Benthem and Pacuit use neighborhood models for evidence and develop a theory of “evidence management.”

References

1. Artemov, S.: The logic of justification. *Review of Symbolic Logic* 1(4), 477–513 (2008)
2. Artemov, S., Fitting, M.: Justification logic. In: Zalta, E.N. (ed.) *The Stanford Encyclopedia of Philosophy*. Fall 2011 edn. (2011)
3. Artemov, S., Nogina, E.: Introducing justification into epistemic logic. *Journal of Logic and Computation* 15(6), 1059–1073 (2005)
4. Baltag, A., Moss, L.S.: Logics for epistemic programs. *Synthese* 139(2), 165–224 (2004)
5. Baltag, A., Moss, L.S., Solecki, S.: The logic of public announcements, common knowledge, and private suspicions. In: Gilboa, I. (ed.) *Proceedings of the 7th Conference on Theoretical Aspects of Rationality and Knowledge (TARK VII)*, Evanston, Illinois, USA, pp. 43–56 (1998)

6. Baltag, A., Smets, S.: A qualitative theory of dynamic interactive belief revision. In: Bonanno, G., van der Hoek, W., Wooldridge, M. (eds.) *Logic and the Foundations of Game and Decision Theory (LOFT 7)*. Texts in Logic and Games, pp. 9–58. Amsterdam University Press (2008)
7. Blackburn, P., de Rijke, M., Venema, Y.: *Modal Logic*. Cambridge University Press (2001)
8. Gettier, E.: Is justified true belief knowledge? *Analysis* 23, 121–123 (1963)
9. Grove, A.: Two modellings for theory change. *Journal of Philosophical Logic* 17, 157–170 (1988)
10. Lehrer, K.: *Theory of Knowledge*. Routledge, London (1990)
11. Lehrer, K., Paxson, T.J.: Knowledge: Undeclared justified true belief. *Journal of Philosophy* 66, 225–237 (1969)
12. Lehrer, K.: *Theory of Knowledge*. Westview Press, United States (2000)
13. Pappas, G., Swain, M. (eds.): *Essays on Knowledge and Justification*. Cornell Univ. Press, Ithaca (1978)
14. Renne, B., Sack, J., Yap, A.: Dynamic Epistemic Temporal Logic. In: He, X., Horty, J., Pacuit, E. (eds.) *LORI 2009*. LNCS, vol. 5834, pp. 263–277. Springer, Heidelberg (2009)
15. Rott, H.: Stability, strength and sensitivity: Converting belief into knowledge. *Erkenntnis* 61, 469–493 (2004)
16. Sack, J.: Temporal languages for epistemic programs. *Journal of Logic, Language, and Information* 17(2), 183–216 (2008)
17. Stalnaker, R.: On logics of knowledge and belief. *Philosophical Studies* 128, 169–199 (2006)
18. van Benthem, J., Pacuit, E.: Dynamic logics of evidence-based beliefs. *Studia Logica* 99(1), 61–92 (2011)
19. van Benthem, J., Velazquez Quesada, F.: The dynamics of awareness. *Synthese* 177, 5–27 (2010)
20. van Benthem, J.: *Logical Dynamics of Information and Interaction*. Cambridge University Press (2011)
21. Velazquez Quesada, F.R.: Small steps in dynamics of information. ILLC dissertation series DS-2011-02, University of Amsterdam ILLC (2011)
22. Yap, A.: Dynamic epistemic logic and temporal modality. In: Girard, P., Roy, O., Marion, M. (eds.) *Dynamic Formal Epistemology*. Synthese Library, vol. 351, pp. 33–50. Springer (2011)

Minimization via Duality

Nick Bezhanishvili¹, Clemens Kupke², and Prakash Panangaden³

¹ Department of Computing, Imperial College London

² Department of Computer Science, University of Oxford

³ School of Computer Science, McGill University

Abstract. We show how to use duality theory to construct minimized versions of a wide class of automata. We work out three cases in detail: (a variant of) ordinary automata, weighted automata and probabilistic automata. The basic idea is that instead of constructing a maximal quotient we go to the dual and look for a minimal subalgebra and then return to the original category. Duality ensures that the minimal subobject becomes the maximally quotiented object.

1 Introduction

The main goal of this paper is to exploit a simple observation of category theory that yields some striking results when applied to particular instances. The simple observation is this: a quotient construction in a category is the same as a subobject construction in the opposite category. How can such a simple observation be useful or interesting? First, there is the mathematically nontrivial task of giving a useful alternative description of the opposite category. Second it is useful because finding the relevant subobjects may be easier than an explicit quotient construction in the original category of interest. Furthermore one gets completely new algorithms for the minimization problem. Third, it is interesting because one ties together seemingly unrelated ideas into a coherent whole. These ideas have appeared not only in minimization but in questions related to machine learning, specifically the problem of modelling systems with hidden state [9]

We are going to exploit duality or, more precisely, a dual equivalence for the minimization of automata, ie. we use the fact that a given category of transition systems, say C , is *equivalent* to the opposite of another category \mathcal{D}^{op} . Our basic strategy for minimization is as follows. We start with a transition system S of a certain type. We go to the dual category and look for a special subobject: a so-called *zero-generated subobject* which has the property that it *must embed* into any other subobject of the dual object. Then we come back to the original category and find that the zero-generated subobject is the *minimal realization* of the original object. The general abstract statement is of course very easy, practically an observation. Examples of this phenomenon are, however, very interesting.

The first example that we treat in this way is based on reasoning about systems with hidden state [9]. It is related in spirit, but not in detail, to an old algorithm due to Brzozowski [7] which is a minimization algorithm that seems to work by black magic. A precise categorical treatment of Brzozowski's algorithm has recently been given by Bonchi et al. [6]. Our second example deals with weighted automata. We derive an algorithm due to Stefan Kiefer [12] which he discovered after listening to a presentation

of the third author. The third example is that of probabilistic transition systems. We use Gelfand duality to show how to minimize what are called belief automata in the AI literature [11].

2 The Abstract Setting

In this section we describe a categorical setting in which our minimization results can be placed. This framework is by no means exhaustive, we hope it will lend itself to further generalizations and clarifications. In this paper, it merely has the role to present our minimization procedures under a common umbrella. Readers who are interested in concrete examples, rather than in the unifying categorical viewpoint, can skip this section.

The starting point for our presentation of automata minimization is a basic duality: Let \mathcal{C} and \mathcal{D} be dually equivalent categories and let $F : \mathcal{C}^{\text{op}} \rightarrow \mathcal{D}$ and $G : \mathcal{D} \rightarrow \mathcal{C}^{\text{op}}$ be the contravariant functors establishing this duality. The following three examples will be discussed in our paper.

- Example 1.*
1. Let $\mathcal{C} = \mathbf{FinSet}$ be the category of finite sets and $\mathcal{D} = \mathbf{FinBA}$ the category of finite Boolean algebras. The epimorphisms in \mathcal{C} are the surjective maps.
 2. Let $\mathcal{C} = \mathcal{D} = \mathbf{FDVec}$ the category of finite-dimensional real vector spaces and linear maps. The epimorphisms in \mathcal{C} with domain $C \in \mathcal{C}$ correspond to congruence relations on C .
 3. Let $\mathcal{C} = \mathbf{KHaus}$ the category of compact Hausdorff spaces and $\mathcal{D} = \mathbf{C^*Alg}$ be the category of real commutative C^* -algebras. The epimorphisms in \mathcal{C} with domain $C \in \mathcal{C}$ correspond to equivalence relations on C that are closed in the topology.

In our examples we will be dealing with categories \mathcal{C} that have an (Epi, Mono)-factorization system (cf. [2]), i.e., we can factor any morphism $f \in \mathcal{C}$ into $f = m \circ e$ where m is a mono, e is an epi and this factorization is unique up to isomorphism. The automata we are studying in this paper are coalgebras for functors $T : \mathcal{C} \rightarrow \mathcal{C}$ that preserve monos. It is not difficult to see that the factorization structure of \mathcal{C} can be lifted to $\mathbf{Coalg}(T)$ if $T : \mathcal{C} \rightarrow \mathcal{C}$ preserves monos and that the factorization of a morphism in $\mathbf{Coalg}(T)$ can be computed in the base category \mathcal{C} (cf. e.g. [1]). This factorization structure can be used in order to define the minimization of a coalgebra.

Definition 1. Let $T : \mathcal{C} \rightarrow \mathcal{C}$ be a functor and let $S = (S, \gamma)$ be a T -coalgebra. An epimorphism $e : S \rightarrow S'$ of S is called minimization of S iff for all other quotients $e' : S \rightarrow S''$ there exists a unique map $g : S'' \rightarrow S'$ such that $g \circ e' = e$.

In order to ensure that the minimization of a T -coalgebra exists we further assume that the base category is co-wellpowered, i.e., that for every object C in \mathcal{C} the collection of epimorphisms $e : C \rightarrow C'$ forms a set. In this case, given our assumptions on the category \mathcal{C} and the functor $T : \mathcal{C} \rightarrow \mathcal{C}$, the minimization of a coalgebra $(X, \gamma) \in \mathbf{Coalg}(T)$ exists (cf. [1] Thm. 3.8).

In order to compute the minimization of a T -coalgebra we proceed as follows: We first lift the basic duality $\mathcal{C} \cong \mathcal{D}^{\text{op}}$ to a duality between $\mathbf{Coalg}(T)$ and some category $\mathbf{Alg}(L)$ for some functor $L : \mathcal{D} \rightarrow \mathcal{D}$ and we denote by $\hat{F} : \mathbf{Coalg}(T) \rightarrow \mathbf{Alg}(L)^{\text{op}}$ and $\hat{G} : \mathbf{Alg}(L) \rightarrow \mathbf{Coalg}(T)^{\text{op}}$ the functors that witness this dual equivalence.

Remark 1. Note that a functor $L : \mathcal{D} \rightarrow \mathcal{D}$ such that $\mathbf{Coalg}(T) \cong \mathbf{Alg}(L)^{\text{op}}$ can always be defined as follows: Let $G : \mathcal{D} \rightarrow \mathcal{C}^{\text{op}}$ and $F : \mathcal{C}^{\text{op}} \rightarrow \mathcal{D}$ be the functors that constitute the dual equivalence between \mathcal{C} and \mathcal{D} . If we put $L = F \circ T^{\text{op}} \circ G$, it is not difficult to see that $\mathbf{Coalg}(T) \cong \mathbf{Alg}(L)^{\text{op}}$. In our examples we will use, however, more concrete representations of L and of the algebras in $\mathbf{Alg}(L)$.

By the duality $\mathbf{Coalg}(T) \cong \mathbf{Alg}(L)^{\text{op}}$ and the fact that minimizations in $\mathbf{Coalg}(T)$ exist it is clear that each object $A \in \mathbf{Alg}(L)$ has a minimal subobject $i : A' \rightarrow A$ such that for all other subobjects $j : A'' \rightarrow A$ there exists a map $k : A' \rightarrow A''$ such that $j \circ k = i$. For any T -coalgebra (ie, automaton) $S = (S, \gamma : S \rightarrow TS)$ the minimization can therefore be obtained by taking the dual of the minimal subobject of the dual algebra $\hat{F}S$ of S . We summarize the observations of this section in the following meta-theorem.

Theorem 1. *Let \mathcal{C} be a co-wellpowered category with (Epi,Mono)-factorization, \mathcal{D} be a category dually equivalent to \mathcal{C} , and $T : \mathcal{C} \rightarrow \mathcal{C}$ be a functor that preserves monos. Assume also that $L : \mathcal{D} \rightarrow \mathcal{D}$ is a functor such that $\mathbf{Coalg}(T)$ and $\mathbf{Alg}(L)$ are dually equivalent. Let*

$$\hat{F} : \mathbf{Coalg}(T) \rightarrow \mathbf{Alg}(L)^{\text{op}} \quad \text{and} \quad \hat{G} : \mathbf{Alg}(L) \rightarrow \mathbf{Coalg}(T)^{\text{op}}$$

be the functors establishing this duality. Let S be any T -coalgebra with C_m being the minimal subobject of $\hat{F}(S)$. Then $\hat{G}(C_m)$ is the minimization of S .

3 Partially Observable Deterministic Finite Automata

In this section we are working in the basic setting of Example [□□□](#)

Definition 2. *We define partially observable deterministic finite automata (PODFA) to be quintuples $S = (S, \mathcal{A}, \mathcal{O}, \delta : S \rightarrow S^{\mathcal{A}}, \gamma : S \rightarrow 2^{\mathcal{O}})$ where S is a finite set of states, \mathcal{A} is a finite set of actions, \mathcal{O} is a finite set of observations, δ is a transition function and γ is an observation function.*

The only difference from the usual automata is the presence of observations. We do not see what state the automaton is in currently, instead we see some observations that partly reveal the state, we may think of γ as a relation between states and observations.

We fix the set of actions and observations henceforth; thus automata are just triples (S, δ, γ) . These automata are coalgebras for the functor $T : \mathbf{FinSet} \rightarrow \mathbf{FinSet}$ given by

$$T(S) = S^{\mathcal{A}} \times 2^{\mathcal{O}}, \quad T(f : S \rightarrow S') = \lambda(\alpha : \mathcal{A} \rightarrow S, O \subseteq \mathcal{O}). \langle f \circ \alpha, O \rangle.$$

A homomorphism for these coalgebras is a function $f : S \rightarrow S'$ such that the following diagram commutes:

$$\begin{array}{ccc} S & \xrightarrow{f} & S' \\ \langle \delta, \gamma \rangle \downarrow & & \downarrow \langle \delta', \gamma' \rangle \\ S^{\mathcal{A}} \times 2^{\mathcal{O}} & \xrightarrow{f^{\mathcal{A}} \times \text{id}} & S'^{\mathcal{A}} \times 2^{\mathcal{O}} \end{array}$$

where $f^{\mathcal{A}}(\alpha) = f \circ \alpha$. This translates to the following conditions:

$$\forall s \in S, \omega \in \mathcal{O}, \omega \in \gamma(s) \iff \omega \in \gamma'(f(s)), \text{ and } \forall s \in S, a \in \mathcal{A}, f(\delta(s, a)) = \delta'(f(s), a). \quad (1)$$

Definition 3. *The category of coalgebras for the functor \mathbf{T} is called **PODFA**, the category of partially observable deterministic finite automata.*

In order to define a category dual to **PODFA** we use a well-known finite variant of basic Stone duality; see, for example [8].

Fact 2. *The categories **FinSet** and **FinBA** are dually equivalent. This dual equivalence is established via the contravariant functors $2 : \mathbf{FinSet} \rightarrow \mathbf{FinBA}$ and $\text{At} : \mathbf{FinBA} \rightarrow \mathbf{FinSet}$ where 2 denotes the contravariant power set functor and At the functor that maps a given finite Boolean algebra to its set of atoms and a homomorphism $h : B_1 \rightarrow B_2$ to the function $\text{At}(h) : \text{At}(B_2) \rightarrow \text{At}(B_1)$ with $\text{At}(h)(b') := \bigwedge \{b \in B_1 \mid b' \leq h(b)\}$ for all $b' \in \text{At}(B_2)$.*

Based on this duality, we establish a duality between the category **PODFA** and the category of finite Boolean algebras with operators.

Definition 4. *The category **FBAO** of finite Boolean algebras with operators (FBAOs) has as objects finite Boolean algebras B with the usual operators \wedge and \neg with a greatest element \top and least element \perp together with unary operators $(a) : B \rightarrow B$, for each action a , such that (a) is a Boolean homomorphism. For each observation $\omega \in \mathcal{O}$, we also have constants $\underline{\omega}$. We denote an object of **FBAO** by*

$$\mathcal{B} = (B, \{(a) \mid a \in \mathcal{A}\}, \{\underline{\omega} \mid \omega \in \mathcal{O}\}, \top, \wedge, \neg).$$

*The morphisms of **FBAO** are the usual Boolean homomorphisms preserving, in addition, the constants and the unary operators.*

Remark 2. FBAOs fit easily into the general framework that we outlined in Section 2. We define a functor $\mathbf{L} : \mathbf{FinBA} \rightarrow \mathbf{FinBA}$ by putting $\mathbf{L}B = \coprod_{a \in \mathcal{A}} B + F_{\mathbf{FinBA}}(\mathcal{O})$ where $F_{\mathbf{FinBA}}(\mathcal{O})$ denotes the free Boolean algebra generated by the finite set of observations \mathcal{O} and \coprod and $+$ denote coproducts in **FinBA**. As the category of Boolean algebras is locally finite, finitely generated Boolean algebras are finite and a coproduct of two finite Boolean algebras is also finite. It is easy to see that the categories $\mathbf{Alg}(\mathbf{L})$ and **FBAO** are isomorphic.

Finite Boolean algebra with operators provide the categorical dual of finite automata.

Proposition 1. $\mathbf{PODFA} \cong \mathbf{FBAO}^{\text{op}}$ via two contravariant functors $\hat{F} : \mathbf{PODFA}^{\text{op}} \rightarrow \mathbf{FBAO}$ and $\hat{G} : \mathbf{FBAO} \rightarrow \mathbf{PODFA}^{\text{op}}$.

Proof. We define the two functors explicitly and check the requisite conditions. First we define a functor $\hat{F} : \mathbf{PODFA}^{\text{op}} \rightarrow \mathbf{FBAO}$ as follows.

$$(S, \delta, \gamma) \mapsto (2^S, \mathbb{A}, \mathcal{Q})$$

where $\underline{\omega} = \{s \mid \omega \in \gamma(s)\}$ and $(a)b = \{s \mid \delta(s, a) \in b\}$, b an element of the Boolean algebra 2^S is just a subset of the states of \mathcal{S} . The arrow part of the functor is just inverse image which clearly preserves the usual Boolean operators. The fact that it preserves the new operators and the constants is immediate from equations (II).

Now we define a functor $\hat{G} : \mathbf{FBAO} \rightarrow \mathbf{PODFA}^{\text{op}}$:

$$(B, \mathbb{A}, \Omega) \mapsto (\text{At}(B), \delta, \gamma).$$

Here $\text{At}(B)$ means the set of atoms of B , since B is a finite Boolean algebra it is certainly atomic. We define

$$\delta(b, a) = \bigwedge \{b' \in B \mid b \leq (a)b'\} \quad (2)$$

and

$$\gamma(b) = \{\omega \in \mathcal{O} \mid b \leq \underline{\omega} \in \Omega\} \quad (3)$$

where b is an atom of B . The arrow part of the functor is defined as follows:

$$f : B_1 \rightarrow B_2, \quad \hat{G}(f)(b \in \text{At}(B_2)) = \bigwedge \{c \in B_1 \mid b \leq f(c)\}. \quad (4)$$

In order to ensure that \hat{G} is well defined we need to verify that $\delta(b, a)$ is an atom and also that $\hat{G}(f)(b)$ is an atom in the above definitions when b is an atom.

Assume that b is an atom. We claim that $\beta \stackrel{\text{def}}{=} \bigwedge \{b' \mid b \leq (a)b'\} \neq \perp$. Suppose that it were equal to \perp . The set $\{b' \mid b \leq (a)b'\}$ consists of finitely many elements $\{b_1, \dots, b_n\}$, so we have $b_1 \wedge b_2 \wedge \dots \wedge b_n = \perp$. Now for each b_i we have $b \leq (a)b_i$ by the definition of the set, so $b \leq (a)b_1 \wedge \dots \wedge (a)b_n$. Using the fact that the operators (a) are Boolean homomorphisms it is easy to see that $b \leq (a)b_1 \wedge \dots \wedge (a)b_n = (a)(b_1 \wedge \dots \wedge b_n) = (a)\perp = \perp$. Thus if $b \neq \perp$ then $\beta \neq \perp$.

In fact there is a unique atom c in the set $\{b' \mid b \leq (a)b'\}$. To see that there is an atom in this set we calculate as follows

$$b \leq \top \equiv (a)\top = (a)\left(\bigvee_{x \in \text{At}(B)} x\right) = \bigvee_{x \in \text{At}(B)} (a)x$$

and, since, b is an atom, for some $c \in \text{At}(B)$ we have $b \leq (a)c$. Suppose that there were two such atoms c_1, c_2 . Then $b \leq (a)c_1, b \leq (a)c_2$, hence $b \leq (a)(c_1 \wedge c_2) = (a)\perp = \perp$, which contradicts the assumption that b is an atom. Since $\{b' \mid b \leq (a)b'\}$ contains an atom and its meet is not \perp the meet must be that atom. Thus, we could have defined $\delta(b, a)$ to be the unique atom c such that $b \leq (a)c$. The proof that $\hat{G}(f)(b)$ is an atom is similar.

Now suppose that we have a PODFA $\mathcal{S} = (S, \delta, \gamma)$, we construct $\hat{G}(\hat{F}(\mathcal{S}))$ to obtain another PODFA $\mathcal{S}' = (S', \delta', \gamma')$. Clearly the atoms of 2^S just gives back S ; in other words $\{\cdot\} : S \rightarrow \text{At}(2^S)$ is a (natural) isomorphism. Then

$$\gamma'(\{s\}) = \{\omega \in \mathcal{O} \mid \{s\} \subseteq \underline{\omega}\} = \{\omega \mid \omega \in \gamma(s)\} = \gamma(s).$$

Now consider any state s and action a

$$\begin{aligned} \delta'(\{s\}, a) &= \{s'\} && \text{with } \{s\} \subseteq (a)\{s'\} \\ &= \{s'\} && \text{with } \{s\} \subseteq \{s'' \mid \delta(s'', a) = s'\} \\ &= \{\delta(s, a)\}. \end{aligned}$$

Thus the iso in the category of sets is in fact an iso in the category **PODFA**. To show that we get an isomorphism going the other way is straightforward as well. We embed B into $2^{\text{At}(B)}$ by mapping $b \in B$ to $\{u \in \text{At}(B) \mid u \leq b\}$. ■

This shows that we get a precise duality between automata with observations and FBAOs. The interesting point, however, is that one can get a minimal realization via this theory.

Definition 5. Consider the language \mathcal{L} :

$$t ::= \top \mid \hat{\omega} \mid \langle a \rangle t \mid t_1 \wedge t_2 \mid \neg t.$$

Now, for a given automaton $\mathcal{S} = (S, \delta, \gamma)$ we can define a satisfaction relation for the formulas as follows:

$$\begin{array}{lll} s \models \top & \text{always} & s \models \hat{\omega} \text{ iff } \omega \in \gamma(s) \\ s \models t_1 \wedge t_2 & \text{iff } s \models t_1 \text{ and } s \models t_2 & s \models \neg t \text{ iff } s \not\models t \\ s \models \langle a \rangle t & \text{iff } \delta(s, a) \models t. & \end{array}$$

We say that a subset U of S is definable by \mathcal{L} if $U = \llbracket t \rrbracket := \{s \in S \mid s \models t\}$ for some $t \in \mathcal{L}$.

From the subsets definable by \mathcal{L} , call this D , we can obtain another Boolean algebra with operators.

Proposition 2. The set of subsets definable by \mathcal{L} of a PODFA \mathcal{S} , with the evident restrictions of the operators, gives a minimal subalgebra C_m of $\hat{F}(\mathcal{S})$, i.e., a subalgebra C_m that is contained in any other subalgebra of $\hat{F}(\mathcal{S})$.

The following theorem is an immediate consequence of Theorem [1](#).

Theorem 3. The automaton $\hat{G}(C_m)$ is the minimal realization of \mathcal{S} .

In the previous construction we used a logic that had the Boolean connectives as well as the modal operators and the observations as primitive propositions. In fact, one just needs the last two in the logic provided one is working with deterministic systems. As we will show below in such cases the Boolean connectives are superfluous.

Given the set of actions and observations we define a simple sublanguage \mathcal{L}_0 of \mathcal{L} as follows:

$$\mathcal{L}_0 \ni t ::= \hat{\omega} \mid \langle a \rangle t.$$

The following lemma is easy to prove, if we are working with deterministic automata.

Lemma 1. If a formula of \mathcal{L} distinguishes two states then so does a formula of \mathcal{L}_0 .

We say that a set of states is definable by \mathcal{L}_0 if it is of the form $\llbracket t \rrbracket$ for some t in \mathcal{L}_0 . The subsets definable by \mathcal{L}_0 do not form a Boolean algebra. Let us call the collection of these subsets D' . Then the Boolean algebra generated by D' is exactly the same as the Boolean algebra of the subsets definable by \mathcal{L} . However, this is only true because the operators are required to be boolean algebra homomorphisms. The reason we require

that the operators be boolean algebra homomorphisms is because we are dealing with *deterministic* automata. In general, one could have sets defined by formulas like $(a)(p \wedge (b)q)$ in D which are not in the Boolean algebra generated by D' . Thus, we can work with D' and use the Boolean algebra generated by it to get to D and then proceed as we did in the previous subsection.

Proposition 3. *Given a PODFA the boolean algebra with operators generated by the subsets definable by \mathcal{L}_0 gives the zero-generated subobject of the dual FBAO object. Thus one can construct the minimal PODFA by using formulas of \mathcal{L}_0 .*

Our algorithm resembles aspects of Brzozowski's algorithm in the following sense. We construct a dual object which resembles Brzozowski's reversed machine in the sense that an FBAO object is essentially running the original machine backward. However, there are many differences. First of all we are not dealing with reachability aspects and secondly we have multiple observations; essentially we have Moore machines rather than acceptors. The precise categorical version of Brzozowski's algorithm is given by Bonchi et al. [6] who exploit the duality of observability and reachability as originally discussed by Arbib and Manes [3].

4 Linear Weighted Automata

In this section we show that the minimization of weighted automata fit into our framework. Weighted automata are also called multiplicity automata and are important in learning theory [4]. Our minimization construction is inspired by a construction by Stephan Kiefer [12] who came up with a minimization algorithm after hearing a talk on duality for ordinary automata given by the third author of the present paper. As in [5], we model linear weighted automata as coalgebras for a functor on the category \mathbf{FVect} of finite dimensional real vector spaces. In order to prepare for the categorical picture we recall the well-known self-duality property of the category of finite-dimensional vector spaces.

For a vector space V we denote by $\text{End}(V)$ the set of linear maps $T : V \rightarrow V$ and by V^* the dual space of V consisting of linear mappings $\phi : V \rightarrow \mathbb{R}$. We can easily extend $(-)^*$ to a contravariant functor $(-)^* : \mathbf{FVect} \rightarrow \mathbf{FVect}^{\text{op}}$ that maps a morphism $T : V \rightarrow W$ to the morphism $T^* : W^* \rightarrow V^*$ given by $T^*(\sigma)(v) := \sigma(T(v))$ for $\sigma \in W^*, v \in V$.

Fact 4. *There is a natural isomorphism $\tau : \text{Id}_{\mathbf{FVect}} \rightarrow (-)^{**}$, given by $\tau_V(v)(\phi) = \phi(v)$ for all $v \in V, \phi \in V^*$ and $V \in \mathbf{FVect}$. Therefore the pair $((-)^*, ((-)^*)^{\text{op}})$ is a dual equivalence between \mathbf{FVect} and itself.*

Definition 6. *The category \mathbf{WAut}_o of output linear weighted automata over alphabet \mathcal{A} can be defined as the category of coalgebras for the functor $\mathbf{T} = (-)^{\mathcal{A}} \times \mathbb{R}$. Equivalently, an object in \mathbf{WAut}_o can be represented as a triple*

$$S = (V, T : \mathcal{A} \rightarrow \text{End}(V), \eta \in V^*)$$

where $V \in \mathbf{FVect}$, \mathcal{A} is a finite set (input alphabet), and T maps input letters $a \in \mathcal{A}$ to linear transformations $T(a) : V \rightarrow V$. The vector $\eta \in V^*$ is the final vector.

The dimension of V is called the dimension of the automaton and is denoted $\dim(S)$. Given a weighted automata $\mathcal{S} \in \mathbf{WAut}_o$ we extend the map T to a map $T : \mathcal{A}^* \rightarrow \text{End}(V)$ inductively by putting $T(\epsilon) = \text{id}_V$ and $T(a \cdot w) = T(a) \cdot T(w)$.

Given two linear weighted automata $\mathcal{S}_1 = (V_1, T_1 : \mathcal{A} \rightarrow \text{End}(V_1), \eta_1 \in V_1^*)$ and $\mathcal{S}_2 = (V_2, T_2 : \mathcal{A} \rightarrow \text{End}(V_2), \eta_2 \in V_2^*)$, a \mathbf{WAut}_o -morphism from \mathcal{S}_1 to \mathcal{S}_2 is a linear function $M : V_1 \rightarrow V_2$ such that $M^*(\eta_2) = \eta_1$ and for all $a \in \mathcal{A}$ we have $M \circ T_1(a) = T_2(a) \circ M$.

Based on the self-duality of \mathbf{FVect} we can define a duality of output linear weighted automata with linear weighted automata with initial states.

Definition 7. The category \mathbf{WAut}_i of linear weighted automata with initial state over alphabet \mathcal{A} has as objects triples $(V, T : \mathcal{A} \rightarrow \text{End}(V), \alpha)$ where $V \in \mathbf{FVect}$ is a finite-dimensional real vector space, $T(a) : V \rightarrow V$ is a linear function for all $a \in \mathcal{A}$ and $\alpha \in V$ is an initial state vector. A \mathbf{WAut}_i -morphism from (V_1, T_1, α_1) to (V_2, T_2, α_2) is a linear function $M : V_1 \rightarrow V_2$ such that $M(\alpha_1) = \alpha_2$ and such that for all $a \in \mathcal{A}$ we have $M \circ T_1(a) = T_2(a) \circ M$.

Remark 3. Again the dual category \mathbf{WAut}_i can be represented as a category of algebras for a functor. We define a functor $L : \mathbf{FVect} \rightarrow \mathbf{FVect}$ by putting $LV = \coprod_{a \in \mathcal{A}} V + F_{\mathbf{FVect}}(1)$ where $F_{\mathbf{FVect}}(1) = \mathbb{R}$ denotes the free real vector space generated by the one-element set. It is now not difficult to see that $\mathbf{Alg}(L)$ is isomorphic to \mathbf{WAut}_i .

We now use the duality from Fact 4 to define the dual of a linear weighted automaton in order to establish a duality $\mathbf{WAut}_o \cong \mathbf{WAut}_i^{\text{op}}$.

Definition 8. Let $\mathcal{S} = (V, T, \eta)$ be an output linear weighted automaton. The dual of \mathcal{S} is defined as $\hat{F}(\mathcal{S}) = (V^*, T^*, \eta)$ where $T^*(w) := (T(w))^*$ for $w \in \mathcal{A}^*$. We extend \hat{F} to a contravariant functor from \mathbf{WAut}_o to \mathbf{WAut}_i by putting $\hat{F}(M) := M^*$ for a given morphism $M : \mathcal{S}_1 \rightarrow \mathcal{S}_2 \in \mathbf{WAut}_o$. Likewise, the dual of a linear weighted automaton with initial state $\mathcal{S} = (V, T, \alpha)$ is defined as $\hat{G}(\mathcal{S}) = (V^*, T^*, \tau_V(\alpha))$ where $\tau_V : V \rightarrow V^{**}$ is the V -component of the natural isomorphism from Fact 4 and we extend \hat{G} to a contravariant functor from \mathbf{WAut}_i to \mathbf{WAut}_o by putting $\hat{G}(M) = M^*$.

Proposition 4. The natural isomorphism $\tau : \text{Id}_{\mathbf{FVect}} \rightarrow (-)^{**}$ extends to natural isomorphisms $\tau : \text{Id}_{\mathbf{WAut}_i} \rightarrow \hat{F} \circ \hat{G}$ and $\tau : \text{Id}_{\mathbf{WAut}_o} \rightarrow \hat{G} \circ \hat{F}$. Consequently, the contravariant functors $\hat{F} : \mathbf{WAut}_o \rightarrow \mathbf{WAut}_i$ and $\hat{G} : \mathbf{WAut}_i \rightarrow \mathbf{WAut}_o$ form a dual equivalence of \mathbf{WAut}_o and \mathbf{WAut}_i .

Proof. We show that for any linear weighted automaton $\mathcal{S} = (V, T : \mathcal{A}^* \rightarrow \text{End}(V), \alpha \in V, \eta \in V^*)$ the map $\tau_V : V \rightarrow V^{**}$ is an isomorphism between \mathcal{S} and $\hat{G}\hat{F}(\mathcal{S})$. Clearly it is an isomorphism of the underlying vector spaces.

We need to check that $\tau_V : \mathcal{S} \rightarrow \hat{G}\hat{F}(\mathcal{S}) \in \mathbf{WAut}_o$, where

$$\mathcal{S}^{dd} = (V^{**}, T^{**}, \tau_{V^*}(\eta))$$

by definition.

Obviously τ_V satisfies the morphism condition for the initial vector. For the condition regarding the final vector, one can easily see that

$$\begin{aligned} \tau_V^*(\tau_{V^*}(\eta))(v) &\stackrel{(\text{Def. of } \omega^*)}{=} (\tau_{V^*}(\eta))(\tau_V(v)) \\ &\stackrel{(\text{Def. of } \tau_{V^*})}{=} \tau_V(v)(\eta) \\ &\stackrel{(\text{Def. of } \tau_V)}{=} \eta(v) \end{aligned}$$

for any $v \in V$. Therefore $\tau_V^*(\tau_{V^*}(\eta)) = \eta$ as required. Furthermore, for $w \in \mathcal{A}^*$, $v \in V$ and $\phi \in V^*$, we calculate:

$$\begin{aligned} (T^{**}(w)(\tau_V(v)))(\phi) &= ((T^*(w))^*(\tau_V(v)))(\phi) \\ &= \tau_V(T(w)^*(\phi)) = (T(w)^*(\phi))(v) \\ &= \phi(T(w)(v)) = (\tau_V(T(w)(v)))(\phi) \end{aligned}$$

which implies that $T^{**}(w)(\tau_V(v)) = \tau_V(T(w)(v))$ as required by the diagram in the definition of a \mathbf{WAut}_o -morphism. Similarly one can check that τ gives rise to a natural isomorphism from $\text{Id}_{\mathbf{WAut}_i}$ to $\hat{F} \circ \hat{G}$. ■

In order to apply our general minimization theorem we have to describe the minimal subobjects of an object $\mathcal{S} \in \mathbf{WAut}_i$. As in the previous case this can be done using formulas. One can think of these formulas as *tests* which can be applied to a given weighted automaton.

Definition 9. Let \mathcal{A} be a finite alphabet. The set of tests \mathcal{T} over \mathcal{A} is inductively defined by

$$\mathcal{T} \ni t ::= \epsilon \mid a \cdot t, a \in \mathcal{A}.$$

Given some $\mathcal{S} = (V, T, \eta \in V^*) \in \mathbf{WAut}_o$ the semantics of a test $t \in \mathcal{T}$ is a linear function $\llbracket t \rrbracket : V \rightarrow \mathbb{R}$ that is defined by putting $\llbracket \epsilon \rrbracket(v) = \eta(v)$ and $\llbracket a \cdot t \rrbracket(v) = \llbracket t \rrbracket(T(a)(v))$.

We can use these test functions in order to define elements of $\hat{F}(\mathcal{S})$ and in order to obtain a way to compute minimal subobjects in the category \mathbf{WAut}_i :

Lemma 2. Let $\mathcal{S} = (V, T, \eta) \in \mathbf{WAut}_o$ and let $\hat{F}(\mathcal{S}) = (V^*, T^*, \eta)$. The smallest subobject of $\hat{F}(\mathcal{S})$ is equal to $(U, T_{\uparrow U}^*, \eta)$ where $U = \text{span}(\{\llbracket t \rrbracket \mid t \in \mathcal{T}\})$ denotes the subspace of V generated from test functions and $T_{\uparrow U}^*$ denotes the restriction of T^* to U , i.e., $T_{\uparrow U}^*(a) = (T(a)^*)_{\uparrow U}$.

Proof. In order to see that $T_{\uparrow U}^*$ is well-defined we have to check that for any test $t \in \mathcal{T}$ and any $a \in \mathcal{A}$ we have $T(a)^*(\llbracket t \rrbracket) \in U$. For $v \in V$ we have $T(a)^*(\llbracket t \rrbracket)(v) = \llbracket t \rrbracket(T(a)(v)) = \llbracket a \cdot t \rrbracket(v)$ and thus $T(a)^*(\llbracket t \rrbracket) = \llbracket a \cdot t \rrbracket \in U$ as required. It is not difficult to see that any subobject of $\hat{F}(\mathcal{S})$ has to contain $\{\llbracket t \rrbracket \mid t \in \mathcal{T}\}$ and therefore also U , which shows that $(U, T_{\uparrow U}^*, \eta)$ is indeed the smallest subobject of $\hat{F}(\mathcal{S})$. ■

By Theorem 1 the following is immediate:

Theorem 5. Let $\mathcal{S} = (V, T, \eta) \in \mathbf{WAut}_o$ be an output linear weighted automaton. The minimization of \mathcal{S} can be computed as $\hat{G}(S')$ where S' is the minimal subobject of $\hat{F}(\mathcal{S}) \in \mathbf{WAut}_i$ as described in Lemma 2. ■

Spelling out the description of minimal subobjects provided by Lemma 2 one can now obtain a minimization procedure for linear weighted automata. Owing to space limitations, however, we cannot provide the details of this construction here. Finally note that in [5] it is proven that behavioural equivalence in the category that we call \mathbf{WAut}_o is precisely language equivalence. This shows that minimization in \mathbf{WAut}_o coincides with the standard notion of minimization for linear weighted automata.

5 Belief Automata

A fundamental example of our approach is probabilistic transition systems. These are not a special case of weighted automata for the simple reason that the set of probability distributions on a set does not form a vector space. One needs a different theory. There are two possible approaches: one is to use Stone's version of Gelfand duality [18,19] and the other is to exploit convexity ideas, which we will not do here but will follow up in future work. For textbook presentations of Stone's version of Gelfand duality we recommend Johnstone [10]. Some key categorical constructions are due to Joan Wick-Pelletier [14].

There are several versions of probabilistic transition systems that one could use. A basic model is *partially observable Markov decision processes* (POMDPs) [17,16]. This is a model invented in the operations research community but now of major importance in the artificial intelligence community, especially in machine learning; see, for example, the paper by Kaelbling et al. [11]. Usually POMDPs have a notion of *reward* associated with transitions and the main interest is in optimizing the reward by finding a suitable policy. However, we will ignore the reward for this paper. POMDPs without rewards are rather like Labelled Markov Processes [15] except that the state is not observable. A very interesting duality theory for LMPs also based on Gelfand duality has been developed by Mislove et al. [13]. Our constructions are somewhat different because we deal with partial observability. We use the left adjoint of a particular forgetful functor to construct C^* -algebras [14], they construct C^* -algebras directly from tests. They use their theory to discuss composition of LMPs while we are interested in minimization.

In the AI literature one does not deal with POMDPs directly, rather one deals with what are called belief automata. This is an automaton where the states are probability distributions over the states of the original POMDP and the transitions are interpreted as changes in the distribution. One works directly with this object for analysis, learning prediction etc. and the original POMDP is not used. The belief automaton becomes a set of probability distributions over a finite set which is just a simplex in a finite dimensional real vector space and hence a compact Hausdorff space.

Definition 10. We define $\mathbf{CHA}(\mathcal{A}, \mathcal{O})$ as the category of compact Hausdorff automata over \mathcal{A} and \mathcal{O} . The objects are triples $\mathcal{K} = (K, \Delta : K \rightarrow K^{\mathcal{A}}, \Gamma : K \rightarrow \mathbf{Sub}(\mathcal{O}))$, where K is a compact Hausdorff space and Δ is a transition function over the set of actions \mathcal{A} while Γ maps a state to a subdistribution over the set of observations \mathcal{O} , ie. to a function $f : \mathcal{O} \rightarrow [0, 1]$ such that $\sum_{\omega \in \mathcal{O}} f(\omega) \leq 1$. For fixed $a \in \mathcal{A}$, the function $\Delta(\cdot)(a) : K \rightarrow K$ is continuous and Γ is continuous when $\mathbf{Sub}(\mathcal{O})$, the set of subdistributions over \mathcal{O} , is topologized with the topology it inherits as a subset of $[0, 1]^{|\mathcal{O}|}$.

The morphisms of $\mathbf{CHA}(\mathcal{A}, \mathcal{O})$, $f : (K, \Delta, \Gamma) \rightarrow (K', \Delta', \Gamma')$ are continuous functions $f : K \rightarrow K'$ such that $\Delta'(f(k))(a) = f(\Delta(k)(a))$ and $\Gamma'(f(k))(\omega) = \Gamma(k)(\omega)$ for $k \in K$, $a \in \mathcal{A}$ and $\omega \in \mathcal{O}$. In other words, $\mathbf{CHA}(\mathcal{A}, \mathcal{O})$ is isomorphic to the category of coalgebras for the functor $T = (-)^{\mathcal{A}} \times \text{Sub}(\mathcal{O}) : \mathbf{KHaus} \rightarrow \mathbf{KHaus}$ on the category of compact Hausdorff spaces.

We will usually just write \mathbf{CHA} instead of $\mathbf{CHA}(\mathcal{A}, \mathcal{O})$. We employ the well known Gelfand duality between the category of compact Hausdorff spaces and the category $\mathbf{C}^*\mathbf{Alg}$ of real commutative C^* -algebras in order to develop a duality for \mathbf{CHA} (cf. [10, Chapter IV.4]). For the dual of a compact Hausdorff automaton we take commutative C^* -algebras as the underlying structure but we need to equip them with operators as we did with Boolean algebras with operators. We will assume fixed sets \mathcal{A} of actions and \mathcal{O} of observations.

Definition 11. An object of the category \mathbf{CAO} is a real commutative C^* -algebra C together with a set of operators $(a) : C \rightarrow C$, for each $a \in A$ and a distinguished set of elements $\underline{\omega}$ of C for each $\omega \in \mathcal{O}$. The operators (a) are C^* -algebra morphisms (i.e. ring homomorphisms). We require that $0 \leq \underline{\omega} \leq 1$ and $\sum_{\omega \in \mathcal{O}} \underline{\omega} \leq 1$. The morphisms are morphisms of C^* -algebras, which also preserve the constants and the additional operators.

Remark 4. Again it is possible to view \mathbf{CAO} as a category L-algebras. There is, however, a slight problem that needs to be solved: the forgetful functor from the category of C^* -algebras does not have a left adjoint, i.e., it is not possible to always construct the free C^* -algebra for a given set of generators. Luckily there is a slight variant of the forgetful functor that does have a left adjoint: Consider the functor from $\mathbf{C}^*\mathbf{Alg}$ to \mathbf{Set} that maps a given C^* -algebra C to its unit interval $I = \{c \in C \mid 0 \leq c \leq 1\}$. This functor does have a left-adjoint ([14]) which we denote with $F_{\mathbf{C}^*\mathbf{Alg}}$. We now define a functor $L : \mathbf{C}^*\mathbf{Alg} \rightarrow \mathbf{C}^*\mathbf{Alg}$ by putting $LC = \coprod_{a \in \mathcal{A}} C + F_{\mathbf{C}^*\mathbf{Alg}}(\mathcal{O})/J$ where $F_{\mathbf{C}^*\mathbf{Alg}}(\mathcal{O})/J$ is a quotient of $F_{\mathbf{C}^*\mathbf{Alg}}(\mathcal{O})$ that ensures that the sum of the elements of $F_{\mathbf{C}^*\mathbf{Alg}}(\mathcal{O})/J$ that correspond to the constants $\underline{\omega}$ is smaller or equal to 1. A number of details have to be checked; owing to space limitations we omit them here and will give them in the full version of the paper.

There are contravariant functors between these categories that establish a dual equivalence. We describe these functors before stating the duality theorem. We name the functors $\mathbf{A} : \mathbf{CHA} \rightarrow \mathbf{CAO}^{\text{op}}$ and $\mathbf{H} : \mathbf{CAO}^{\text{op}} \rightarrow \mathbf{CHA}$.

We are given an object $\mathcal{K} = (K, \Delta, \Gamma)$ of \mathbf{CHA} . We define an object $\mathbf{A}(\mathcal{K}) = (C, \{(a)\}, \{\underline{\omega}\})$ of \mathbf{CAO} as follows. We define C to be the C^* -algebra of continuous functions from K to \mathbb{R} with the pointwise order.

The operators and constants are defined by $(a)f := \lambda x : K.f(\Delta(x)(a))$ for $a \in \mathcal{A}$ and $f \in C$ and $\underline{\omega} := \lambda x : K.\Gamma(x)(\omega)$ for $\omega \in \mathcal{O}$. Clearly $(a) : C \rightarrow C$ is a ring homomorphism for any $a \in \mathcal{A}$. Also,

$$\sum_{\omega \in \mathcal{O}} \underline{\omega} = \sum_{\omega \in \mathcal{O}} \lambda x : K.\Gamma(x)(\omega) \leq \lambda x : K.1.$$

Given a morphism $h : \mathcal{K} \rightarrow \mathcal{K}'$, where $\mathcal{K}' = (K', \Delta', \Gamma')$ we define $\mathbf{A}(h) : \mathbf{A}(\mathcal{K}') \rightarrow \mathbf{A}(\mathcal{K})$ by $\mathbf{A}(h)(f) = f \circ h$. It is easy to verify that this is a functor.

For the reverse direction we proceed as follows: Given a **CAO** $(C, \{(a)\}, \{\underline{\omega}\})$ we define the dual compact Hausdorff automaton $\mathcal{K} = (K, \Delta, \Gamma)$ by putting $K = \text{Hom}(C, \mathbb{R})$, $\Delta(f)(a) = \lambda x.f((a)x)$ and $\Gamma(f)(\omega) = f(\underline{\omega})$ for $a \in \mathcal{A}$, $\omega \in \mathcal{O}$ and $f \in \text{Hom}(C, \mathbb{R})$. The set K is turned into a compact Hausdorff space by adding the topology that is generated by the sets $\hat{c} = \{f \in \text{Hom}(C, \mathbb{R}) \mid f(c) \neq 0\}$.

Lemma 3. *With these definitions the maps Δ and Γ are continuous.*

Proof. It is not difficult to see that Δ is continuous:

$$\Delta_a^{-1}(\hat{c}) = \{g \mid \Delta_a(g)(c) \neq 0\} = \{g \mid g((a)(c)) \neq 0\} = \widehat{(a)(c)}.$$

The fact that Γ is well-defined follows because any C^* -algebra morphism $f \in \text{Hom}(C, \mathbb{R})$ has the property that $\|f(x)\| \leq \|x\|$ for all $x \in C$ and thus $\|\sum_{\omega \in \mathcal{O}} f(\underline{\omega})\| \leq \sum_{\omega \in \mathcal{O}} \|\underline{\omega}\| \leq 1$. The fact that Γ is continuous can be seen as follows: By Gelfand duality we can assume w.l.o.g. that $C \cong C(X)$ for some compact Hausdorff space X . Also by Gelfand duality we know that the map

$$\begin{aligned} \tau_X : X &\rightarrow (C(X) \rightarrow \mathbb{R}) \\ x &\mapsto \lambda f.f(x) \end{aligned}$$

is a homeomorphism. In particular, for any $g : C(X) \rightarrow \mathbb{R}$ there is $x_g \in X$ such that $\tau_X(x_g) = g$. Our goal is to prove that the map

$$\begin{aligned} \Gamma_\omega : \mathcal{HC}(X) &\rightarrow \mathbb{R} \\ (g : C(X) \rightarrow \mathbb{R}) &\mapsto g(\underline{\omega}) \end{aligned}$$

is continuous. Let $U \subseteq \mathbb{R}$ be an open set. Then

$$\begin{aligned} \Gamma_\omega^{-1}(U) &= \{g : C(X) \rightarrow \mathbb{R} \mid g(\underline{\omega}) \in U\} \\ &= \tau_X(\{x \mid \underline{\omega}(x) \in U\}) = \tau_X(\underline{\omega}^{-1}(U)) \end{aligned}$$

Because $\underline{\omega} \in C(X)$ is continuous we have $\underline{\omega}^{-1}(U)$ is open and by the fact that τ_X is a homeomorphism we have $\Gamma_\omega^{-1}(U)$ is open as well. As $U \subseteq \mathbb{R}$ was an arbitrary open set this shows that Γ_ω is continuous as required. ■

This operation can be extended to a functor $\mathbf{H} : \mathbf{CAO} \rightarrow \mathbf{CHA}^{\text{op}}$ that maps a function $h : C_1 \rightarrow C_2 \in \mathbf{CAO}$ to the function $\mathbf{H}(h) : \mathbf{H}(C_2) \rightarrow \mathbf{H}(C_1)$ given by $\mathbf{H}(h)(g) := g \circ h$.

Theorem 6. *The functors $\mathbf{H} : \mathbf{CAO} \rightarrow \mathbf{CHA}^{\text{op}}$ and $\mathbf{A} : \mathbf{CHA}^{\text{op}} \rightarrow \mathbf{CAO}$ form an equivalence of categories, i.e. the categories **CHA** and **CAO** are dually equivalent: $\mathbf{CHA} \simeq \mathbf{CAO}^{\text{op}}$.*

Proof. Let $\mathcal{K} = (K, \Delta, \Gamma)$ be a compact Hausdorff automaton and let $\mathbf{HA}\mathcal{K} = (K', \Delta', \Gamma')$. It is an immediate consequence of Gelfand duality that the carrier spaces K and K' are isomorphic via the isomorphism $\tau_{\mathcal{K}} : K \rightarrow K'$ given by $\tau_{\mathcal{K}}(x)(f) := f(x)$ (note that

$K' = \text{Hom}(C(K), \mathbb{R})$, i.e. $\tau_{\mathcal{K}}(x)$ has to map continuous real-valued functions on K to a real number). The fact that $\tau_{\mathcal{K}} : \text{Id} \rightarrow \mathbf{H} \circ \mathbf{A}$ is natural is also a consequence of Gelfand duality. The only thing which remains to be checked is that $\tau_{\mathcal{K}}$ is an automaton morphism. In order to see this we calculate for $a \in \mathcal{A}$, $x \in K$ and $g \in C(K)$ that

$$\begin{aligned} (\Delta'(\tau_{\mathcal{K}}(x))(a))(g) &= \tau_{\mathcal{K}}(x)((a)g) = \tau_{\mathcal{K}}(x)(\lambda y.g(\Delta(y)(a))) \\ &= g(\Delta(x)(a)) = \tau_{\mathcal{K}}(\Delta(x)(a))(g) \end{aligned}$$

where all equalities are immediate consequences of the definitions. This shows that $\tau_{\mathcal{K}}$ is an automaton morphism regarding the transition structures Δ and Δ' . Let us now check that $\tau_{\mathcal{K}}$ also preserves all observations $\omega \in \mathcal{O}$:

$$\Gamma'(\tau_{\mathcal{K}}(x))(\omega) = \tau_{\mathcal{K}}(x)(\underline{\omega}) = \underline{\omega}(x) = \Gamma(x)(\omega).$$

For the other direction of the equivalence we have to show that the C^* -algebra isomorphism $\alpha_C : C \rightarrow C(\text{Hom}(C, \mathbb{R}))$ that exists by Gelfand duality for each $C = (C, \{(a)\}, \{\underline{\omega}\}) \in \mathbf{CAO}$ and that is given by $\alpha_C(c)(f) := f(c)$ for $f \in \text{Hom}(C, \mathbb{R})$ is in fact a \mathbf{CAO} -morphism. It then follows that $\alpha : \text{Id} \rightarrow \mathbf{A} \circ \mathbf{H}$ is a natural isomorphism. In order to see that α_C is indeed a \mathbf{CAO} -morphism we calculate

$$\begin{aligned} (\alpha_C((a)c)(f)) &= f((a)c) = (\Delta(f)(a))(c) \\ &= \alpha_C(c)(\Delta(f)(a)) = (a)'(\alpha_C(c)) \\ (\alpha_C(\underline{\omega})(f)) &= \underline{\omega}(f) = \Gamma(f)(\omega) = \underline{\omega}'(f) \end{aligned}$$

where the $(a)'$ and $\underline{\omega}'$ denote the operators and constants of \mathbf{AHC} which is the C^* -algebra with operators obtained by following both functors. ■

We are now going to use the duality in order to minimize compact Hausdorff automata. Let us first define a set of tests.

Definition 12. *The set of tests for automata in \mathbf{CHA} is defined as follows*

$$\mathcal{T} \ni t ::= \underline{\omega}, \omega \in \mathcal{O} \mid (a)t, a \in \mathcal{A}.$$

A test t gives rise to a function $\llbracket t \rrbracket : K \rightarrow \mathbb{R}$:

$$\begin{aligned} \llbracket \underline{\omega} \rrbracket(k) &:= \Gamma(k)(\omega) && \text{for } \omega \in \mathcal{O} \\ \llbracket (a)t \rrbracket(k) &:= \llbracket t \rrbracket(\Delta(k)(a)) && \text{for } \omega \in \mathcal{O}, w \in \mathcal{A}^*. \end{aligned}$$

We say two states $k, k' \in K$ in an automaton (K, Δ, Γ) are test equivalent (notation: $k \sim k'$) if for all tests $t \in \mathcal{T}$ we have $\llbracket t \rrbracket(k) = \llbracket t \rrbracket(k')$.

The tests can be used in order to obtain a better understanding of the smallest subobject (subalgebra) of a given C^* -algebra with operators $\mathbf{A}(\mathcal{K}) \in \mathbf{CAO}$.

Proposition 5. *Let $\mathcal{K} \in \mathbf{CHA}$ be a compact Hausdorff automaton, let $\mathbf{A}(\mathcal{K})$ be its dual C^* -algebra with operators and let $C = (C, \{(a)\}, \{\underline{\omega}\})$ be the smallest subobject (subalgebra) of $\mathbf{A}(\mathcal{K})$. Then $C \subseteq \{f \in C(K) \mid \forall k, k'. k \sim k' \text{ implies } f(k) = f(k')\}$, i.e. all functions in C agree on the equivalence classes defined by test equivalence on \mathcal{K} .*

Proof.(Sketch) It is clear that $\llbracket \mathcal{T} \rrbracket = \{\llbracket t \rrbracket \mid t \in \mathcal{T}\} \subseteq C$. We can generate a C^* -algebra with operators from $\llbracket \mathcal{T} \rrbracket$ by adding the constant functions and by closing under $+$, \cdot and under limits in the sup norm. All functions generated in that way, and thus all functions in C , will be constant on the \sim -equivalence classes. ■

As in the other cases duality gives us the following theorem.

Theorem 7. *Let $\mathcal{K} \in \mathbf{CHA}$ be a compact Hausdorff automaton. The minimization of \mathcal{K} can be computed as $\mathbf{H}(C)$ where C is the minimal subobject of $\mathbf{A}(\mathcal{K}) \in \mathbf{CAO}$ as described in Proposition 5*

As a corollary of the previous proposition and theorem we obtain a description of the minimization of a given compact Hausdorff automaton.

Corollary 1. *Given a compact Hausdorff automaton $\mathcal{K} = (K, \Delta, \Gamma)$, let $\mathcal{K}' = (K, \Delta', \Gamma')$ be a compact Hausdorff automaton such that the states of \mathcal{K}' are \sim -equivalence classes of \mathcal{K} and $\Delta'([k])(a) = \Delta(k)(a)$, $\Gamma'([k]) = \Gamma(k)$ for $k \in K$ where $[k]$ denotes the \sim -equivalence class of k . Then \mathcal{K}' is the minimization of \mathcal{K} .*

6 Conclusions

In this paper we have developed a theory of minimization via duality and showed how it works in three examples. There are some variations of these cases that we have omitted from this abstract for lack of space. For example, we have worked out a version of the weighted automaton case in which there are both initial and final “states” and in which the minimization is achieved by dualizing and using reachability twice. This version is actually being used by Stefan Kiefer and he has implemented the algorithm.

The idea of this kind of double duality was first worked out, in a completely non-categorical way in 2006 [9] and was used as a way of representing systems where one does not know the state space. Closely connected to what we have been doing is the work in [6]: they give a duality-based explanation of the Brzozowski algorithm for deterministic finite automata, but their duality is very different from our Stone duality and takes reachability into account, something we have not done here.

We note that Mislove et al. [13] have used this duality to study labelled Markov processes but with different motivations; their work, while very interesting, is not about minimization but about how to compose systems. Furthermore, the details of their work are somewhat different from ours.

It is clear that there are a number of cases very close to automata, for example tree automata, nondeterministic automata, alternating automata etc. that fall under the same rubric and could be developed as straightforward extensions of the present work. We have begun a collaboration with the authors of [6] and others to unify our points of view and collect several more examples.

Acknowledgments. We have benefitted from discussions with Marcello Bonsangue, Marcelo Fiore, Dexter Kozen, Alexander Kurz, Mike Mislove, Robert Myers, Jan Rutten, Alexandra Silva and James Worrell on this and related topics. We are very grateful to Vincenzo Marra who shared with us his expert knowledge on C^* -algebras and to Stefan Kiefer for explaining to us his algorithm for minimizing weighted automata. The third author would like to thank Chris Hundt, Monica Dinculescu, Joelle Pineau and Doina Precup for the collaboration that led to the precursor of this paper.

This research is supported by a grant from the Natural Sciences and Engineering Research Council of Canada to the third author. The second author acknowledges support by the EU FP7 project SEALS and by the EPSRC projects ConDOR, ExODA and LogMap. The work of the first author was partially supported by the EPSRC grants EP/F032102/1 and EP/F031173/1 and by Rustaveli Science Foundation of Georgia grant FR/489/5-105/11.

References

1. Adámek, J., Bonchi, F., Hülsbusch, M., König, B., Milius, S., Silva, A.: A Coalgebraic Perspective on Minimization and Determinization. In: Birkedal, L. (ed.) FOSSACS 2012. LNCS, vol. 7213, pp. 58–73. Springer, Heidelberg (2012)
2. Adamek, J., Herrlich, H., Strecker, G.E.: Abstract and Concrete Categories: The Joy of Cats. Dover (1990)
3. Arbib, M., Manes, E.: Adjoint machines, state behavior machines and duality. *J. Pure Appl. Algebra* 6, 313–343 (1975)
4. Beimel, A., Bergadano, F., Bshouty, N.H., Kushelevitz, E., Varricchio, S.: Learning functions represented as multiplicity automata. *Journal of the ACM* 47(5), 506–530 (2000)
5. Bonchi, F., Bonsangue, M., Boreale, M., Rutten, J., Silva, A.: A coalgebraic perspective on linear weighted automata. *Information and Computation* 211, 77–105 (2012)
6. Bonchi, F., Bonsangue, M.M., Rutten, J.J.M.M., Silva, A.: Brzozowski’s Algorithm (Co)Algebraically. In: Constable, R.L., Silva, A. (eds.) *Logic and Program Semantics, Kozen Festschrift*. LNCS, vol. 7230, pp. 12–23. Springer, Heidelberg (2012)
7. Brzozowski, J.A.: Canonical regular expressions and minimal state graphs for definite events. In: Fox, J. (ed.) *Proceedings of the Symposium on Mathematical Theory of Automata*. MRI Symposia Series, vol. 12, pp. 529–561. Polytechnic Press of the Polytechnic Institute of Brooklyn (April 1962); book appeared in 1963
8. Givant, S., Halmos, P.: *Introduction to Boolean Algebras*. Undergraduate Texts in Mathematics. Springer (2009)
9. Hundt, C., Panangaden, P., Pineau, J., Precup, D.: Representing systems with hidden state. In: *The Twenty-First National Conference on Artificial Intelligence, AAAI* (2006)
10. Johnstone, P.: *Stone Spaces*. Cambridge Studies in Advanced Mathematics, vol. 3. Cambridge University Press (1982)
11. Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101 (1998)
12. Kiefer, S.: Minimization of weighted automata (2011) (unpublished private communication)
13. Mislove, M., Ouaknine, J., Pavlovic, D., Worrell, J.B.: Duality for Labelled Markov Processes. In: Walukiewicz, I. (ed.) FOSSACS 2004. LNCS, vol. 2987, pp. 393–407. Springer, Heidelberg (2004)
14. Negrepointis, J.W.: Duality in analysis from the point of view of triples. *Journal of Algebra* 19, 228–253 (1971)
15. Panangaden, P.: *Labelled Markov Processes*. Imperial College Press (2009)
16. Smallwood, R.D., Sondik, E.J.: The optimal control of partially observable Markov decision processes over a finite horizon. *Operations Research* 21(5), 1071–1088 (1973)
17. Sondik, E.J.: The optimal control of partially observable Markov processes. Ph.D. thesis, Stanford University (1971)
18. Stone, M.H.: A general theory of spectra I. *Proc. Nat. Acad. Sci. USA* 26, 280–283 (1940)
19. Stone, M.H.: A general theory of spectra II. *Proc. Nat. Acad. Sci. USA* 27, 83–87 (1941)

On Some Subclasses of the Fodor-Roubens Fuzzy Bi-implication

Claudio Callejas, João Marcos, and Benjamín René Callejas Bedregal

LoLITA and DIMAp, UFRN, Brazil

Abstract. The paper deals with fuzzy versions of the classical bi-implication, that is, extensions of classical bi-implication to the canonical domain of mathematical fuzzy logics, the real-valued unit interval $[0, 1]$. Our approach to fuzzy bi-implication may be summarized as follows: first, we recall a well-known approach to bi-implications, by Fodor and Roubens, via the direct axiomatization of the properties of the corresponding class of operators; next, we investigate a particular defining standard of bi-implication in terms of t-norms and r-implications. We study four prospective classes of bi-implications based on such defining standard, by varying the properties of its composing operators, and show that these classes collapse into precisely two increasingly weaker subclasses of the Fodor-Roubens bi-implication.

1 Introduction

The investigation of fuzzy logic in a narrow sense, as subfield of multi-valued logic, was initiated by Petr Hájek in [9], much after the introduction of fuzzy set theory by Lotfi Zadeh in [20]. Since then, however, a lot of debate has happened over the ‘most reasonable way’ of extending the most usual operators from the discrete classical domain $\{0, 1\}$ into the continuum represented by the real-valued unit interval $[0, 1]$. In the meanwhile, if some classical connectives have found well-accepted fuzzy counterparts, others remained largely as a matter of contention. The fuzzy interpretation of conjunction, for instance, is well settled in terms of the triangular norm operator, and similarly for disjunction as its dual [11–14]. Classical negation, in weak and strong forms, also has a reasonably well-studied associated fuzzy operator [6]. Furthermore, there are many competing fuzzy versions of implication, of which [1] seems to be the most widely used in the literature.

Classical bi-implication also does not fall short of fuzzy interpretations, and one may find it studied in the literature under the appellations of T-indistinguishability operator [18], fuzzy bi-implication [2, 4], fuzzy equality [17], fuzzy bi-residuation [11, 15], fuzzy equivalence [7, 8], T-equivalence [16], fuzzy similarity [9] and restricted equivalence function [5].

As it happens, it is relatively common to find in the recent literature investigations that study the relations between different classes of the most common fuzzy operators. This happens for instance with the relation between different classes of triangular norms [12] and with the intersections of different classes of

fuzzy implications [1]. For fuzzy bi-implication, however, the same investigation still remains to be done. In this paper we contribute to fill this gap, by studying the relation between the more well-known definition proposed by Fodor and Roubens and other appealing definitions, old or new, of fuzzy operators that extend the interpretation of the classical bi-implication.

The plan of the paper is as follows: in section 2 we recall some basic definitions and facts about t-norms and r-implications; subsection 3.1 recalls the definitions and proves some important results about the so-called Fodor-Roubens fuzzy bi-implications; section 3.2 studies four classes of bi-implications produced by the defining standard based on the classical equivalence in between $\alpha \Leftrightarrow \beta$ and $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$, and shows that these classes amount to two distinct classes of which one is a subclass of the other, and both are subclasses of Fodor-Roubens bi-implications; we conclude by some considerations concerning open problems and interesting lines for future research.

2 Conjunction and Implication from a Fuzzy Perspective

There are infinite ways in which the interpretation of *conjunction* \wedge may be extended from the classical $\{0, 1\}$ domain to the unit interval $[0, 1]$, but not all of them behave as what is intuitively expected from a generalization of the Boolean conjunction to the unit square. The fuzzy logic community, as a matter of fact, has by and large agreed to impose the properties of t-norms to any extension of the classical conjunction.

The following definitions and examples may be found in [12].

Definition 1. A triangular norm (in short t-norm) is a binary operator T on the unit interval $[0, 1]$ that: (T0) agrees with classical conjunction on $\{0, 1\}$, (T1) is commutative, (T2) is associative, (T3) is monotone on both arguments, and (T4) has 1 as neutral element.

In fact, it is easy to check that (T0) follows from the remaining properties.

The associated notions of continuity are the usual ones. In particular:

Definition 2. A t-norm T is left-continuous if for all non-decreasing sequences $(x_n)_{n \in \mathbb{N}}$ we have that $\lim_{n \rightarrow \infty} T(x_n, y) = T(\lim_{n \rightarrow \infty} x_n, y)$.

There are uncountably many t-norms, but below we mention some of the most well-known among them.

Example 1.

1. $T_M(x, y) = \min(x, y)$ (minimum t-norm)
2. $T_P(x, y) = x \cdot y$ (product t-norm)
3. $T_L(x, y) = \max(x + y - 1, 0)$ (Łukasiewicz t-norm)
4. $T_D(x, y) = \begin{cases} 0 & \text{if } (x, y) \in [0, 1]^2 \\ \min(x, y) & \text{otherwise} \end{cases}$ (drastic t-norm)

Notice, in particular, that T_M , T_P and T_L are all left-continuous, yet T_D is not.

The following constitutes a generalization of the transitivity property in the context t-norms:

Definition 3. *Let T be a left-continuous t-norm, and F a binary operator on $[0, 1]$. We say that F is T -transitive if $T(F(x, y), F(y, z)) \leq F(x, z)$.*

As regards *implication* \Rightarrow , there are several competing approaches to what should constitute its fuzzy counterpart (see for example [6, 10, 19]). Below we propose an axiomatization equivalent to the ones that may be found in [1, 8, 10], which characterize the most common fuzzy implication found in the literature.

Definition 4. *A fuzzy implication is a binary operator I on the unit interval $[0, 1]$ that: (I0) agrees with classical implication on $\{0, 1\}$, (I1) is antitone on the first argument and (I2) is monotone on the second argument.*

The following are examples of fuzzy implications:

Example 2.

1. $I_M(x, y) = \begin{cases} 1 & \text{if } x \leq y \\ y & \text{otherwise} \end{cases}$ (minimum / Gödel implication)
2. $I_P(x, y) = \begin{cases} 1 & \text{if } x \leq y \\ \frac{y}{x} & \text{otherwise} \end{cases}$ (product / Goguen implication)
3. $I_L(x, y) = \min(1 - x + y, 1)$ (contractionless / Łukasiewicz implication)
4. $I_D(x, y) = \begin{cases} y & \text{if } x = 1 \\ 1 & \text{otherwise} \end{cases}$ (drastic / Weber implication)
5. $I_B^1(x, y) = \begin{cases} 0 & \text{if } x = 1 \text{ and } y \neq 1 \\ 1 & \text{otherwise} \end{cases}$ (boolean 1-implication)

The first four examples are well-known, but for the last one, introduced here, we have to check that the corresponding definition satisfies the properties in Definition 4. In any case, (I0) is obvious. Consider now $x_a < x_b$. So, $x_a < 1$, thus $I_B^1(x_a, y) = 1 \geq I_B^1(x_b, y)$, satisfying thus (I1). Next, assume $y_a < y_b \leq 1$, and suppose that $I_B^1(x, y_a) > I_B^1(x, y_b)$. Notice that this is only possible in case $I_B^1(x, y_a) = 1$ and $I_B^1(x, y_b) = 0$. From $I_B^1(x, y_b) = 0$ one may conclude in particular that $x = 1$, and from this and $I_B^1(x, y_a) = 1$ it follows that $y_a = 1$. Contradiction, for $y_a < y_b$. Thus, (I2) is also satisfied.

Definition 5. *A fuzzy implication I is said to satisfy:*

- the identity principle, if $I(x, x) = 1$ (IP)
- the left-ordering property, if $I(x, y) = 1$ whenever $x \leq y$ (LOP)
- the right-ordering property, if $I(x, y) \neq 1$ whenever $x > y$ (ROP)

Given an arbitrary t-norm T and an arbitrary fuzzy implication I , the pair (T, I) is said to satisfy:

- modus ponens, if $T(x, I(x, y)) \leq y$ (MP)

Notice that all implications in Ex. 2 satisfy (LOP), and *a fortiori* also (IP). There are well-known examples of implications failing (LOP) (check the first chapter of [1]), but they will be of no particular interest to us here.

The following operation is used to generalize the Deduction Metatheorem:

Definition 6. *The residuum of a left-continuous t-norm T is the (unique) operation I such that $I(x, y) \geq z$ iff $T(z, x) \leq y$.*

A particularly interesting class of fuzzy implications is the one based on residua:

Definition 7. *A binary operator I on $[0, 1]$ is called an r-implication if there is a t-norm T such that:*

$$I(x, y) = \sup\{z \in [0, 1] \mid T(z, x) \leq y\}$$

In such case we may say also that I is an r-implication based on T , and denote it by I^T . We say that I^T is of type \mathbb{LC} in case T is left-continuous. In such case we also say that (T, I^T) form an adjoint pair, or that I^T is the adjoint companion of T .

Given a left-continuous t-norm T , it should be clear from the above that its residuum I^T is the pointwise largest operation such that (T, I^T) satisfies modus ponens.

Note that:

Proposition 1. *(T_X, I^{T_X}) form adjoint pairs, for each $X \in \{M, P, L\}$.*

It is also the case that (cf. [1, 2]):

Proposition 2. *Let I^T be an r-implication. Then: (i) $I^T(1, y) \geq y$; (ii) I^T satisfies the identity principle; (iii) I^T satisfies the left-ordering property. Assume I^T to be of type \mathbb{LC} . Then: (iv) I^T is T -transitive; (v) I^T satisfies the right-ordering property.*

Even though we will not need the following results here, it is interesting to mention that for r-implications we can immediately count on $I^T(1, y) \leq y$ as well, thus validating (MP), and to mention also the characteristic strengthening of the above result according to which any r-implication that satisfies both (LOP) and (ROP) is of type \mathbb{LC} . It might also be interesting to notice how Prop. 2(v) shows that I_B^1 cannot be the residuum of a left-continuous t-norm, as it obviously fails (ROP). While neither I_D nor I_B^1 are of the type \mathbb{LC} , and on the one hand it is easy to see that I_D is indeed the residuum of T_D , on the other hand it is not at all obvious which t-norm, if any, would I_B^1 be the residuum of.

3 Fuzzy Bi-implication

3.1 Via Axiomatization

As it happens with implication, for the bi-implication \Leftrightarrow there is also no universal agreement on what should constitute its fuzzy counterpart. The most well-known class of fuzzy bi-implications was investigated by Fodor and Roubens and is characterized by the following properties (see [8]):

Definition 8. *The class of f -bi-implications contains binary operators B on the unit interval $[0, 1]$ respecting the following axioms:*

- (B1) $B(x, y) = B(y, x)$ (B -commutativity)
- (B2) $B(x, x) = 1$ (B -identity)
- (B3) $B(0, 1) = 0$
- (B4) *If $w \leq x \leq y \leq z$, then $B(w, z) \leq B(x, y)$*

In view of (B1), (B2) and (B3), it is easy to see that any Fodor-Roubens fuzzy bi-implication is bound to agree with classical bi-implication on $\{0, 1\}$. We will refer to this ‘boundary’ property as (B0).

Here are some examples of f -bi-implications:

Example 3

1. $B_M(x, y) = \begin{cases} 1 & \text{if } x = y \\ \min(x, y) & \text{otherwise} \end{cases}$
2. $B_P(x, y) = \begin{cases} 1 & \text{if } x = y \\ \frac{\min(x, y)}{\max(x, y)} & \text{otherwise} \end{cases}$
3. $B_L(x, y) = 1 - |x - y|$
4. $B_D(x, y) = \begin{cases} y & \text{if } x = 1 \\ x & \text{if } y = 1 \\ 1 & \text{otherwise} \end{cases}$
5. $B_B^{TI1}(x, y) = \begin{cases} 1 & \text{if } x = y \text{ or } x, y \neq 1 \\ 0 & \text{otherwise} \end{cases}$

Definition 9. *A fuzzy bi-implication B is said to satisfy:*

- *the diagonal principle, if $B(x, y) \neq 1$ whenever $x \neq y$* (DP)

It should be clear that:

Theorem 1. (i) *Not all f -bi-implications satisfy the diagonal principle.* (ii) *There are f -bi-implications that are T -intransitive, that is, that fail T -transitivity for every t -norm T . Indeed, B_D is a convenient witness to both these facts.*

Proof. Part (i). If $x < y < 1$ then $B_D(x, y) = 1$. Since we have $x \neq y$ and $B_D(x, y) = 1$ then B_D does not satisfy (DP). Part (ii). Let T be an arbitrary t -norm. Then, in view of (T4) and the definition of B_D , $T(B_D(1, .9), B_D(.9, .8)) = T(.9, 1) = .9 \not\leq .8 = B_D(1, .8)$. Therefore, B_D fails to be T -transitive. □

Moreover:

Theorem 2. *The following property holds good for any r -implication I^T :*

- $\min(I^T(x, y), I^T(y, x)) = I^T(\max(x, y), \min(x, y))$

Proof. Let I^T be of type $\mathbb{L}\mathbb{C}$. Assume without loss of generality that $x \leq y$. Recall that, by Prop. 2(iii), I^T satisfies (LOP), thus $\min(I^T(x, y), I^T(y, x)) = \min(1, I^T(y, x)) = I^T(y, x)$. Notice, in addition, that $y = \max(x, y)$ and $x = \min(x, y)$, once $x \leq y$. □

3.2 Via a Defining Standard over t-Norms and Fuzzy Implications

Inspired by the classical (in fact, intuitionistic) equivalence in between $\alpha \Leftrightarrow \beta$ and $(\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)$, in this section we explore the fuzzy bi-implication obtained by setting as defining standard $B(x, y) = T(I(x, y), I(y, x))$. We shall call this *TI defining standard* for bi-implication. As a matter of fact, a very general result may be proven about such defining standard, when r-implications are involved:

Theorem 3. *Given $B(x, y) = T(I(x, y), I(y, x))$, where I is an r-implication, the specific choice of t-norm T is inconsequential. Indeed, the following property holds good in general:*

$$- B(x, y) = \min(I(x, y), I(y, x))$$

Proof. Assume without loss of generality that $x \leq y$. Since, by Prop. 2(iii), the r-implication I satisfies (LOP), we have that $B(x, y) = T(I(x, y), I(y, x)) = T(1, I(y, x))$, and by (T4), we know that $T(1, I(y, x)) = I(y, x)$. Using (LOP) again, we conclude that $I(y, x) = \min(I(x, y), I(y, x))$. \square

In the definitions that follow, we fix the *TI* defining standard for fuzzy bi-implications, assume that T is a t-norm and I an r-implication, and experiment with properties associated to left-continuity. We start by proposing the following very generous class of fuzzy bi-implications:

Definition 10. *The class of aa-bi-implications contains all binary operators B on $[0, 1]$ following the *TI* defining standard and based on arbitrary t-norms and arbitrary r-implications, that is, operators defined by setting $B(x, y) = T_1(I^{T_2}(x, y), I^{T_2}(y, x))$, where T_1 and T_2 are arbitrary t-norms.*

One may readily prove that:

Theorem 4. *Every aa-bi-implication B satisfies the equation $B(1, y) \geq y$.*

Proof. By Theor. 3, we know that $B(1, y) = \min(I^T(1, y), I^T(y, 1))$, for some appropriate r-implication I_T , which by Prop. 2(iii) must satisfy (LOP). From the latter property we conclude that $I^T(y, 1) = 1$, thus, $B(1, y) = I^T(1, y)$. The proof is completed by recalling from Prop. 2(i) that $I^T(1, y) \geq y$. \square

Theorem 5. *Every aa-bi-implication is an f-bi-implication.*

Proof. Let T be a t-norm, I be an r-implication and B be the aa-bi-implication based on T and I . It is obvious that B satisfies (B1) and (B3), and (B2) follows from Prop. 2(ii). Now, recall by Prop. 2(iii) that I satisfies (LOP), and assume $w \leq x \leq y \leq z$. So:

$$\begin{aligned}
 B(w, z) &= T(I(w, z), I(z, w)) \\
 &= T(1, I(z, w)) && \text{by (LOP), once } w \leq z \\
 &= I(z, w) && \text{by (T4)} \\
 &\leq I(y, x) && \text{by (I1), once } z \geq y, \text{ and (I2), once } w \leq x \\
 &= T(I(x, y), I(y, x)) && \text{by (LOP), once } x \leq y \\
 &= B(x, y)
 \end{aligned}$$

Therefore, B satisfies (B4). \square

A restricted version of the above definition may be found in [2]:

Definition 11. *The class of a -bi-implications contains all aa -bi-implications in which $T_1 = T_2$, that is, in which I^{T_2} is precisely the residuum of T_1 .*

Example 4. The ‘drastic’ bi-implication B_D (Ex. [3]4) is an a -bi-implication. Indeed, $B_D(x, y) = T_D(I_D(x, y), I_D(y, x))$.

In view of Ex. [4] and Theor. [1] we know that there are a -bi-implications (thus, *a fortiori*, aa -bi-implications) that are intransitive and fail the diagonal principle.

As a corollary of Theor. [3], however, it is easy to see that the classes of aa -bi-implications and a -bi-implications are coextensive, even though the former class might have initially seemed to be more inclusive than the latter. So:

Theorem 6. *Every aa -bi-implication is an a -bi-implication.*

In what follows we restrict a bit further the preceding definitions of fuzzy bi-implication.

Definition 12. *The class of al -bi-implications contains all binary operators B on $[0, 1]$ following the TI defining standard and based on arbitrary t -norms and r -implications of type \mathbb{LC} , that is, operators defined through the equation $B(x, y) = T_1(I^{T_2}(x, y), I^{T_2}(y, x))$, where T_1 is an arbitrary t -norm and I^{T_2} an r -implication of type \mathbb{LC} .*

The following specialization of al -bi-implications was studied in [11]:

Definition 13. *The class of ℓ -bi-implications contains all al -bi-implications in which $T_1 = T_2$, that is, in which I^{T_2} is precisely the adjoint companion of T_1 .*

Example 5. B_M , B_P and B_L are ℓ -bi-implications.

Again, as an immediate corollary of Theor. [3], we know that the two latter classes of bi-implications are coextensive, that is:

Theorem 7. *Every al -bi-implication is an ℓ -bi-implication.*

As a more interesting side-effect of Theor. [3], the following results from [3] on ℓ -bi-implications may also be generalized to al -bi-implications:

Theorem 8. *Every al -bi-implication based on an r -implication I^T of type \mathbb{LC} enjoys both the diagonal principle and T -transitivity.*

Proof. Let T_1 be a t -norm, I^T be an r -implication of type \mathbb{LC} and B be the al -bi-implication $B(x, y) = T_1(I^T(x, y), I^T(y, x))$ based on T_1 and I^T . By Theor. [3], we know that $B(x, y) = \min(I^T(x, y), I^T(y, x))$. So, $B(x, y) = 1$ iff both $I^T(x, y) = 1$ and $I^T(y, x) = 1$. Given Prop. [2](v), I^T satisfies (ROP), so $I^T(x, y) = 1$ and $I^T(y, x) = 1$ imply that $x \leq y$ and $y \leq x$. It follows that $x = y$ whenever $B(x, y) = 1$, in other words, that B satisfies (DP).

Now we are going to check that $T(B(x, y), B(y, z)) \leq I^T(x, z)$. Recall that, by Prop. 2(iv), I^T is T -transitive, once T is left-continuous. So:

$$\begin{aligned} T(B(x, y), B(y, z)) &= \\ &= T(\min(I^T(x, y), I^T(y, x)), \min(I^T(y, z), I^T(z, y))) && \text{by Theor. 3} \\ &\leq T(I^T(x, y), I^T(y, z)) && \text{by (T3)} \\ &\leq I^T(x, z) && \text{by } T\text{-transitivity of } I^T \end{aligned}$$

For analogous reasons, it is also true that $T(B(x, y), B(y, z)) \leq I^T(z, x)$. Therefore, $T(B(x, y), B(y, z)) \leq \min(I^T(x, z), I^T(z, x))$ and again by Prop. 2(iii) and Theor. 3, it follows that $\min(I^T(x, z), I^T(z, x)) = B(x, z)$. Thus, B is T -transitive. \square

Last but not least, for the sake of comparing the above classes of bi-implication, we may observe that:

Theorem 9. *Not all a -bi-implications are ℓ -bi-implications. Again, B_D bears witness to this fact.*

Proof. Recall from Ex. 4 that B_D is an a -bi-implication and that in Theor. 11 we proved that B_D does not satisfy neither the diagonal principle nor the T -transitivity property for no t -norm T . Since by the definition of the class of ℓ -bi-implications, any creature from this class is in particular an $a\ell$ -bi-implication, and in Theor. 8 we have seen that every $a\ell$ -bi-implication satisfies both (DP) and T -transitivity, the drastic bi-implication B_D gives us two good reasons to conclude that not every a -bi-implication is an ℓ -bi-implication. \square

While the latter distinguishing result should be contrasted with the ordinary facts mentioned in Ex. 5, the next result should be contrasted with Ex. 35.

Theorem 10. *Not all f -bi-implications are a -bi-implications. Indeed, this statement has B_B^{TI1} as witness.*

Proof. Consider any $y \in (0, 1)$. Then, $B_B^{TI1}(1, y) = 0$. Yet, in view of Theor. 4 we know that $B(1, y) \geq y$ for any aa -bi-implication B .

4 Conclusions

There are basically three classes of fuzzy bi-implications to be found in this paper: (B1) f -bi-implications; (B2) a -bi-implications (which we have shown to be coextensive with the apparently more general class of aa -bi-implications); (B3) ℓ -bi-implications (which we have shown to be coextensive with the apparently more general class of $a\ell$ -bi-implications) We have seen that (B3) is a proper subclass of (B2), and that (B2) is a proper subclass of (B1). The full picture may be appreciated in Fig. 11.

In 5 a class $\mathcal{B}4$ of ‘restricted equivalence functions’ is introduced via axiomatization as a subclass of $\mathcal{B}1$. It has not been shown, however this consists

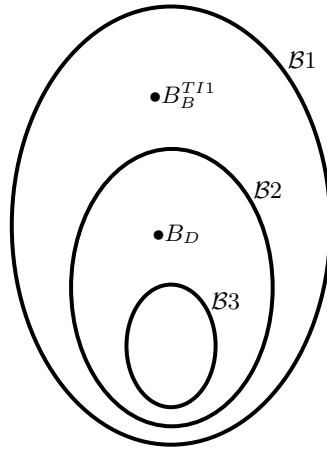


Fig. 1. Subclasses of Fodor-Roubens fuzzy bi-implication

in a proper subclass. An interesting line of investigation would be thus to determine which are the relations that hold between $\mathcal{B}4$ and the other classes of fuzzy bi-implications that we have studied here.

Finally, to get a better view over the possibilities it is also very important to investigate other defining standards for bi-implication, such as the one that sets $B(x, y) = I(S(x, y), T(x, y))$, where T is a t-norm, I a convenient fuzzy implication, and S a t-conorm (the dual of a t-norm, used for interpreting disjunction). Some results have already been found that characterize some classes, based on such an alternative defining standard, that properly extend $\mathcal{B}3$ yet are not extended by $\mathcal{B}1$, providing thus a legitimate alternative to the Fodor-Roubens paradigm. Presenting these results in detail is left as matter for future work.

To the authors of this paper, the class of Fodor-Roubens implications is too inclusive. In particular, satisfaction of the equation $I(1, y) \leq y$ is not enforced, and that seems to us rather inadvisable, at least if one wants to count on (fuzzy) modus ponens. This defect is appropriately fixed by r-implications. In exporting the intuitions behind (Fodor-Roubens) fuzzy implications into the class of f -bi-implications, the defects of the former are inherited, and there will be no way of guaranteeing that, say, $B(1, y) \leq y$. Not by coincidence, the alternative classes of bi-implications we have studied here are based precisely on r-implications, and the fact that they turned out to define proper subclasses of the f -bi-implications containing the most natural examples of fuzzy bi-implications from the literature would seem to lend support to our decision of concentrating our attention on such classes. However, if one takes into account, on a closer look, the additional fact that our main theorems concerning both the class of bi-implications following the TI defining standard and the class of a -bi-implications are based directly on the left-ordering property, rather than on other properties of fuzzy implications, there seems to be some chance that an interesting class of bi-implications might

still lurk somewhere in between $\mathcal{B}1$ and $\mathcal{B}2$. We close our present study by leaving the investigation of this thread open for the interested researcher.

Acknowledgments. This study was partially supported by CNPq (under projects 480832/2011-0 and 553393/2009-0). The authors would like to thank four anonymous referees for their remarks on a preliminary version of this paper.

References

1. Baczyński, M., Jayaram, B.: Fuzzy Implications. STUDDFUZZ. Springer (2008)
2. Bedregal, B.R.C., Cruz, A.P.: A characterization of classic-like fuzzy semantics. *Logic Journal of the IGPL* 16(4), 357–370 (2008)
3. Bodenhofer, U., De Baets, B., Fodor, J.: General Representation Theorems for Fuzzy Weak Orders. In: de Swart, H., Orłowska, E., Schmidt, G., Roubens, M. (eds.) TARSKI II. LNCS (LNAI), vol. 4342, pp. 229–244. Springer, Heidelberg (2006)
4. Bodenhofer, U., De Baets, B., Fodor, J.: A compendium of fuzzy weak orders: Representations and constructions. *Fuzzy Sets and Systems* 158(8), 811–829 (2007)
5. Bustince, H., Barrenechea, E., Pagola, M.: Restricted equivalence functions. *Fuzzy Sets and Systems* 157(17), 2333–2346 (2006)
6. Bustince, H., Burillo, P., Soria, F.: Automorphisms, negations and implication operators. *Fuzzy Sets and Systems* 134, 209–229 (2003)
7. Ćirić, M., Ignjatović, J., Bogdanović, S.: Fuzzy equivalence relations and their equivalence classes. *Fuzzy Sets and Systems* 158, 1295–1313 (2007)
8. Fodor, J., Roubens, M.: Fuzzy Preference Modelling and Multicriteria Decision Support. Theory and Decision Library. Kluwer (1994)
9. Hájek, P.: Metamathematics of fuzzy logic. Trends in Logic. Kluwer (1998)
10. Kitainik, L.: Fuzzy decision procedures with binary relations: towards a unified theory. Theory and Decision Library: System Theory, Knowledge Engineering, and Problem Solving. Kluwer (1993)
11. Klement, E.P., Mesiar, R., Pap, E.: Triangular Norms. Trends in Logic: Studia Logica Library. Kluwer (2000)
12. Klement, E.P., Mesiar, R., Pap, E.: Triangular norms. Position paper I: basic analytical and algebraic properties. *Fuzzy Sets and Systems* 143(1), 5–26 (2004)
13. Klement, E.P., Mesiar, R., Pap, E.: Triangular norms. Position paper II: general constructions and parameterized families. *Fuzzy Sets and Systems* 145(3), 411–438 (2004)
14. Klement, E.P., Mesiar, R., Pap, E.: Triangular norms. Position paper III: continuous t-norms. *Fuzzy Sets and Systems* 145(3), 439–454 (2004)
15. Mesiar, R., Novák, V.: Operations fitting triangular-norm-based biresiduation. *Fuzzy Sets and Systems* 104, 77–84 (1999)
16. Moser, B.: On the T-transitivity of kernels. *Fuzzy Sets and Systems* 157(13), 1787–1796 (2006)
17. Novák, V., De Baets, B.: EQ-algebras. *Fuzzy Sets and Systems* 160(20), 2956–2978 (2009)
18. Recasens, J.: Indistinguishability Operators: Modelling fuzzy equalities and fuzzy equivalence relations. STUDDFUZZ. Springer (2010)
19. Yager, R.: On the implication operator in fuzzy logic. *Information Sciences* 31(2), 141–164 (1983)
20. Zadeh, L.A.: Fuzzy sets. *Information and Control* 8(3), 338–353 (1965)

Linearity in the Non-deterministic Call-by-Value Setting

Alejandro Díaz-Caro^{1,*} and Barbara Petit²

¹ Université Paris 13, Sorbonne Paris Cité, LIPN, F-93430, Villetaneuse, France

² FOCUS (INRIA) – Università di Bologna, Italy

Abstract. We consider the non-deterministic extension of the call-by-value lambda calculus, which corresponds to the additive fragment of the linear-algebraic lambda-calculus. We define a fine-grained type system, capturing the right linearity present in such formalisms. After proving the subject reduction and the strong normalisation properties, we propose a translation of this calculus into the System F with pairs, which corresponds to a non linear fragment of linear logic. The translation provides a deeper understanding of the linearity in our setting.

Introduction

Several non-deterministic extensions of λ -calculus have been proposed in the literature, e.g. [1–4]. In these approaches, the sometimes called *must-convergent parallel composition*, is such that if \mathbf{t} and \mathbf{u} are two λ -terms, $\mathbf{t} + \mathbf{u}$ (also written $\mathbf{t} \parallel \mathbf{u}$) represents the computation that runs either \mathbf{t} or \mathbf{u} non-deterministically. Therefore, $(\mathbf{t} + \mathbf{u})\mathbf{s}$ can run either $\mathbf{t}\mathbf{s}$ or $\mathbf{u}\mathbf{s}$, which is exactly what $\mathbf{t}\mathbf{s} + \mathbf{u}\mathbf{s}$ expresses. Extra rewriting rules (or equivalences, depending on the presentation) are set up to account for such an interpretation, e.g. $(\mathbf{t} + \mathbf{u})\mathbf{s} \rightarrow \mathbf{t}\mathbf{s} + \mathbf{u}\mathbf{s}$.

This right distributivity can alternatively be seen as the one of the function sum: $(\mathbf{f} + \mathbf{g})(x)$ is defined as $\mathbf{f}(x) + \mathbf{g}(x)$. This is the approach of the algebraic lambda-calculi presented in [5] and [6], that were introduced independently but that resulted afterwards to be strongly related [7, 8]. In these algebraic calculi, a scalar pondering each ‘choice’ is considered in addition to the sum of terms.

In the call-by-value (or CBV) version of these algebraic/non-deterministic calculi, e.g. [1, 4, 5], it is natural to consider also the left distributivity of application over sums: $\mathbf{t}(\mathbf{u} + \mathbf{s}) \rightarrow \mathbf{t}\mathbf{u} + \mathbf{t}\mathbf{s}$. To our knowledge, this was first observed in [9]. Indeed, a sum $\mathbf{u} + \mathbf{s}$ is not a value, in the sense that it represents a non-deterministic choice that remains to be done, and therefore cannot substitute the argument x . In algebraic terms, it means that functions are linear: $\mathbf{f}(x + y) = \mathbf{f}(x) + \mathbf{f}(y)$.

The work we present here is motivated by a better understanding of this linearity, and so our first attempt was to interpret such a CBV calculus in Linear Logic [10] (indeed linear functions can be precisely characterised in this logic). Surprisingly, it appeared that the target calculus was a non linear fragment of the intuitionistic multiplicative exponential Linear Logic (IMELL), shining a light

* Supported by grants from DIGITEO and Région Île-de-France.

on the difference between the linearity in these non-deterministic calculi, and the common algebraic notion of linear functions. Since the non linear fragment of IMELL corresponds to the System F with pairs [11, Sec. 1.5], and this latter might be better known by the reader, we present in this paper a (reversible) translation into the System F with pairs.

Notice also that the left distributivity of application over sum induces a completely different computational behaviour compared to the one in CBN calculi. Consider for instance the term $\delta = \lambda x.xx$ applied to a sum $\mathbf{t} + \mathbf{u}$. In the first case, it reduces to $\delta\mathbf{t} + \delta\mathbf{u}$ and then to $\mathbf{t}\mathbf{t} + \mathbf{u}\mathbf{u}$, whereas a CBN reduction would lead to $(\mathbf{t} + \mathbf{u})(\mathbf{t} + \mathbf{u})$ and then to $\mathbf{t}(\mathbf{t} + \mathbf{u}) + \mathbf{u}(\mathbf{t} + \mathbf{u})$. In particular, the CBV algebraic calculus we mentioned above (Lineal, [5]) was originally meant to express quantum computing, where a superposition $\mathbf{t} + \mathbf{u}$ is seen as a quantum superposition. Hence reducing $\delta(\mathbf{t} + \mathbf{u})$ into $(\mathbf{t} + \mathbf{u})(\mathbf{t} + \mathbf{u})$ is considered as the forbidden quantum operation of “cloning” [12], while the alternative reduction to $\mathbf{t}\mathbf{t} + \mathbf{u}\mathbf{u}$ is seen as a “copy”, or CNOT, a fundamental quantum operation [13].

Outline. In this paper we propose (in Sec. 1) a type system, called *Additive*, capturing the linear CBV behaviour of the sum operator that we discussed above. Then we prove its correctness properties, namely subject reduction and strong normalisation in Sec. 2. Its logical interpretation (that is, the translation into System F with pairs) is developed in Sec. 3. We conclude with a discussion about the linearity of the call-by-value setting. We leave in the appendices extra examples and some technical details such as auxiliary lemmas.

1 The Calculus

1.1 The Language

We consider the call-by-value λ -calculus [14] extended with a non-deterministic operator in the spirit of the parallel composition from [2]. This setting can be seen as the additive fragment of Lineal [5]. The set of *terms* and the set of *values* are defined by mutual induction as follows (where variables range over a countable set and are denoted by x, y, z):

$$\begin{array}{ll} \text{Terms:} & \mathbf{t}, \mathbf{u}, \mathbf{s} ::= \mathbf{v} \mid \mathbf{t}\mathbf{u} \mid \mathbf{t} + \mathbf{u} \mid \mathbf{0} \\ \text{Values:} & \mathbf{v} ::= x \mid \lambda x.\mathbf{t} \end{array}$$

Intuitively $\mathbf{t} + \mathbf{u}$ denotes the *non-deterministic choice* between \mathbf{t} and \mathbf{u} , and hence, as discussed in the introduction, $(\mathbf{t} + \mathbf{u})\mathbf{s}$ reduces to the non-deterministic choice $\mathbf{t}\mathbf{s} + \mathbf{u}\mathbf{s}$. Analogously, in this call-by-value setting, $\mathbf{t}(\mathbf{u} + \mathbf{s})$ reduces to $\mathbf{t}\mathbf{u} + \mathbf{t}\mathbf{s}$. The term $\mathbf{0}$ is introduced to express the *impossible computation*, and hence $\mathbf{t} + \mathbf{0}$ always reduces to \mathbf{t} , while $\mathbf{t}\mathbf{0}$ and $\mathbf{0}\mathbf{t}$ reduce to $\mathbf{0}$, because none of them continue reducing (notice that $\mathbf{0}$ is not a value), and have an impossible computation on them. Since the operator $+$ represents a non deterministic choice, where no one have precedence, terms are considered modulo associativity and commutativity of $+$ (that is an *AC-rewrite system* [15]). Notice that considering $\mathbf{t} + \mathbf{u}$ either as

a sum of functions or as a sum of arguments—depending on its position—is also natural with the previous definitions, where $\mathbf{0}$ becomes the sum of 0 elements.

The α -conversion and the set $fv(\mathbf{t})$ of *free variables of \mathbf{t}* are defined as usual (cf. [16, Sec. 2.1]). We say that a term \mathbf{t} is closed whenever $fv(\mathbf{t}) = \emptyset$. Given a term \mathbf{t} and a value \mathbf{v} , we denote by $\mathbf{t}\{\mathbf{v}/x\}$ to the term obtained by simultaneously substituting \mathbf{v} for all the free occurrences of x in \mathbf{t} , taking care to rename bound variables when needed in order to prevent variable capture. Hereafter, terms are considered up to α -conversion. The five rewrite rules plus the β -reduction are summarised as follows.

$$\begin{array}{lll}
 \textit{Distributivity rules:} & \textit{Zero rules:} & \textit{\beta-reduction:} \\
 (\mathbf{t} + \mathbf{u})\mathbf{s} \rightarrow \mathbf{t}\mathbf{s} + \mathbf{u}\mathbf{s}, & \mathbf{0}\mathbf{t} \rightarrow \mathbf{0}, \quad \mathbf{t} + \mathbf{0} \rightarrow \mathbf{t}, & (\lambda x.\mathbf{t})\mathbf{v} \rightarrow \mathbf{t}\{\mathbf{v}/x\}. \\
 \mathbf{t}(\mathbf{u} + \mathbf{s}) \rightarrow \mathbf{t}\mathbf{u} + \mathbf{t}\mathbf{s}, & \mathbf{t}\mathbf{0} \rightarrow \mathbf{0}, &
 \end{array}$$

1.2 The Additive Type System

Our objective is to define a type system, capturing as much as possible the behaviour of $+$. Roughly speaking, we want a system where, if \mathbf{t} has type T and \mathbf{u} has type R , then $\mathbf{t} + \mathbf{u}$ has type $T + R$. So the natural typing rule for such a construction is “ $\Gamma \vdash \mathbf{t} : T$ and $\Gamma \vdash \mathbf{u} : R$ entails $\Gamma \vdash \mathbf{t} + \mathbf{u} : T + R$ ”. We also want a special type distinguishing the impossible computation $\mathbf{0}$, which we call $\bar{0}$. Due to the associative and commutative nature of $+$, we consider an equivalence between types taking into account its commutative nature. Hence if $T + R$ is a type, $R + T$ is an equivalent type. Also the neutrality of $\mathbf{0}$ with respect to $+$ is captured by an equivalence between $T + \bar{0}$ and T . Finally, as usual the arrow type $T \rightarrow R$ characterises the functions taking an argument in T and returning an element of R . However, notice that the type $(T + R) \rightarrow S$ captures a behaviour that is not appearing in our setting: there is no function taking a non-deterministic superposition as argument. Indeed, if \mathbf{v}_1 has type T and \mathbf{v}_2 type R , any function \mathbf{t} distributes $\mathbf{t}(\mathbf{v}_1 + \mathbf{v}_2)$ as $\mathbf{t}\mathbf{v}_1 + \mathbf{t}\mathbf{v}_2$, so \mathbf{t} needs to be characterised by a function taking both T and R , but not simultaneously. In order to capture such a behaviour, we introduce a *unit* type U (i.e. an atomic type with respect to $+$), capturing elements which are not sums of elements, and hence the arrow types have the shape $U \rightarrow T$, where the different arguments to which the function can be applied, are captured by polymorphic types with variables ranging on unit types. For example, the previous term \mathbf{t} can have type $\forall X.(X \rightarrow S)$, where if \mathbf{t} is applied to the above discussed $\mathbf{v}_1 + \mathbf{v}_2$ of type $T + R$, it reduces to $\mathbf{t}\mathbf{v}_1 + \mathbf{t}\mathbf{v}_2$ of type $S[T/X] + S[R/X]$.

To take into account the above discussion, the grammar of the *Additive* type system is defined by mutual induction as follows (where type variables range over a countable set and are denoted by X, Y, Z):

$$\begin{array}{ll}
 \text{Types:} & T, R, S ::= U \mid T + R \mid \bar{0} \\
 \text{Unit types:} & U, V, W ::= X \mid U \rightarrow T \mid \forall X.U
 \end{array}$$

Contexts are denoted by Γ, Δ and are defined as sets of pairs $x : U$, where each term variable appears at most once. The substitution of X by U in T is defined

analogously to the substitution in terms, and is written $T[U/X]$. We also use the vectorial notation $T[\vec{U}/\vec{X}]$ for $T[U_1/X_1] \cdots [U_n/X_n]$ if $\vec{X} = X_1, \dots, X_n$ and $\vec{U} = U_1, \dots, U_n$. To avoid capture, we consider that X_i cannot appear free in U_j , with $j < i$. Free and bound variables of a type are assumed distinct.

The above discussed equivalence relation \equiv on types, is defined as the least congruence such that:

$$T + R \equiv R + T, \quad T + (R + S) \equiv (T + R) + S, \quad T + \bar{0} \equiv T.$$

Within this equivalence, it is consistent to use the following notation:

Notation: $\sum_{i=1}^0 T = \bar{0}$; $\sum_{i=1}^\alpha T_i = \sum_{i=1}^{\alpha-1} T_i + T_\alpha$ if $\alpha \geq 1$.

Remark 1. Every type is equivalent to a sum of unit types.

Returning to the previous example, $\mathbf{t}(\mathbf{v}_1 + \mathbf{v}_2)$ reduces to $\mathbf{t}\mathbf{v}_1 + \mathbf{t}\mathbf{v}_2$ and its type have to be an arrow with a polymorphic unit type at the left. Such a type must allow to be converted into both the type of \mathbf{v}_1 and the type of \mathbf{v}_2 . Hence, consider V_1 and V_2 to be the respective types of \mathbf{v}_1 and \mathbf{v}_2 , we need \mathbf{t} to be of type $\forall X.(U \rightarrow S)$ for some S and where $U[W_1/X] = V_1$ and $U[W_2/X] = V_2$ for some unit types W_1 and W_2 . That is, we need that if \mathbf{t} has such a type, then \mathbf{v}_1 has type $U[W_1/X]$ and \mathbf{v}_2 type $U[W_2/X]$. We can express this with the following rule

$$\frac{\Gamma \vdash \mathbf{t} : \forall X.(U \rightarrow S) \quad \Gamma \vdash \mathbf{v}_1 + \mathbf{v}_2 : U[W_1/X] + U[W_2/X]}{\Gamma \vdash \mathbf{t}(\mathbf{v}_1 + \mathbf{v}_2) : S[W_1/X] + S[W_2/X]}$$

In the same way, for the right distributivity, if \mathbf{t} and \mathbf{u} are two functions of types $U \rightarrow T$ and $V \rightarrow R$ respectively, then the application $(\mathbf{t} + \mathbf{u})\mathbf{v}$ needs U and V to be the type of \mathbf{v} . Therefore, the polymorphism plays a role again, and if \mathbf{t} has type $\forall X.(U \rightarrow T)$ and \mathbf{u} has type $\forall X.(V \rightarrow R)$ such that $U[W_1/X] = V[W_2/X]$ and also equal to the type of \mathbf{v} , then $(\mathbf{t} + \mathbf{u})\mathbf{v}$ has a type. It can be expressed by

$$\frac{\Gamma \vdash \mathbf{t} + \mathbf{u} : \forall X.(U \rightarrow S) + \forall X.(V \rightarrow R) \quad \Gamma \vdash \mathbf{v} : U[W_1/X] = V[W_2/X]}{\Gamma \vdash (\mathbf{t} + \mathbf{u})\mathbf{v} : S[W_1/X] + R[W_2/X]}$$

Notice that when combining both cases, for example in $(\mathbf{t} + \mathbf{u})(\mathbf{v}_1 + \mathbf{v}_2)$, we need the type of \mathbf{t} to be an arrow accepting both the type of \mathbf{v}_1 and the type of \mathbf{v}_2 as arguments, and the same happens with the type of \mathbf{u} . So, the combined rule is

$$\frac{\Gamma \vdash \mathbf{t} + \mathbf{u} : \forall X.(U \rightarrow S) + \forall X.(U \rightarrow R) \quad \Gamma \vdash \mathbf{v}_1 + \mathbf{v}_2 : U[V/X] + U[W/X]}{\Gamma \vdash (\mathbf{t} + \mathbf{u})(\mathbf{v}_1 + \mathbf{v}_2) : S[V/X] + R[W/X]}$$

The arrow elimination has become also a forall elimination. For the general case however it is not enough with the previous rule. We must consider bigger sums, which are not typable with such a rule, as well as arrows with more than one \forall , e.g. $\forall X.\forall Y.(U \rightarrow R)$, where $U[V/X][W/Y]$ has the correct type. Since it is under a sum, and the elimination must be done simultaneously in all the members of the sum, it is not possible with a traditional forall elimination.

The generalised arrow elimination as well as the rest of the typing rules are summarised in Fig. 1. Rules for the universal quantifier, axiom and introduction of arrow are the usual ones. As discussed before, any sum of typable terms can be typed using rule $+_I$. Notice that there is no elimination rule for $+$ since the actual non-deterministic choice step (which eliminates one branch) is not considered here. For similar calculi where the elimination is present in the operational semantics, see e.g. [2, 17]. Finally, a rule assigns equivalent types to the same terms.

$$\begin{array}{c}
\frac{}{\Gamma, x : U \vdash x : U} \text{ax} \quad \frac{}{\Gamma \vdash \mathbf{0} : \bar{\mathbf{0}}} \text{ax}_{\bar{\mathbf{0}}} \quad \frac{\Gamma \vdash \mathbf{t} : T \quad T \equiv R}{\Gamma \vdash \mathbf{t} : R} \equiv \\
\\
\frac{\Gamma, x : U \vdash \mathbf{t} : T}{\Gamma \vdash \lambda x. \mathbf{t} : U \rightarrow T} \rightarrow_I \quad \frac{\Gamma \vdash \mathbf{t} : \sum_{i=1}^{\alpha} \forall \vec{X}. (U \rightarrow T_i) \quad \Gamma \vdash \mathbf{u} : \sum_{j=1}^{\beta} U[\vec{V}_j/\vec{X}]}{\Gamma \vdash \mathbf{t} \mathbf{u} : \sum_{i=1}^{\alpha} \sum_{j=1}^{\beta} T_i[\vec{V}_j/\vec{X}]} \rightarrow_E \\
\\
\frac{\Gamma \vdash \mathbf{t} : T \quad \Gamma \vdash \mathbf{u} : R}{\Gamma \vdash \mathbf{t} + \mathbf{u} : T + R} +_I \quad \frac{\Gamma \vdash \mathbf{t} : \forall X. U}{\Gamma \vdash \mathbf{t} : U[V/X]} \forall_E \quad \frac{\Gamma \vdash \mathbf{t} : U \quad X \notin FV(\Gamma)}{\Gamma \vdash \mathbf{t} : \forall X. U} \forall_I
\end{array}$$

Fig. 1. Typing rules of *Additive*

Example 2. Let $V_1 = U[W_1/X]$, $V_2 = U[W_2/X]$, $\Gamma \vdash \mathbf{v}_1 : V_1$, $\Gamma \vdash \mathbf{v}_2 : V_2$, $\Gamma \vdash \lambda x. \mathbf{t} : \forall X. (U \rightarrow T)$ and $\Gamma \vdash \lambda y. \mathbf{u} : \forall X. (U \rightarrow R)$. Then

$$\frac{\Gamma \vdash \lambda x. \mathbf{t} + \lambda y. \mathbf{u} : \forall X. (U \rightarrow T) + \forall X. (U \rightarrow R) \quad \Gamma \vdash \mathbf{v}_1 + \mathbf{v}_2 : V_1 + V_2}{\Gamma \vdash (\lambda x. \mathbf{t} + \lambda y. \mathbf{u})(\mathbf{v}_1 + \mathbf{v}_2) : T[W_1/X] + T[W_2/X] + R[W_1/X] + R[W_2/X]} \rightarrow_E$$

Notice that this term reduces to $\underbrace{(\lambda x. \mathbf{t})\mathbf{v}_1}_{T[W_1/X]} + \underbrace{(\lambda x. \mathbf{t})\mathbf{v}_2}_{T[W_2/X]} + \underbrace{(\lambda y. \mathbf{u})\mathbf{v}_1}_{R[W_1/X]} + \underbrace{(\lambda y. \mathbf{u})\mathbf{v}_2}_{R[W_2/X]}$.

Example 3. Let $\Gamma \vdash \mathbf{v}_1 : U$ and $\Gamma \vdash \mathbf{v}_2 : V$. Then the term $(\lambda x. x)(\mathbf{v}_1 + \mathbf{v}_2)$, which reduces to $(\lambda x. x)\mathbf{v}_1 + (\lambda x. x)\mathbf{v}_2$, can be typed in the following way:

$$\frac{\Gamma \vdash \lambda x. x : \forall X. X \rightarrow X \quad \Gamma \vdash \mathbf{v}_1 + \mathbf{v}_2 : U + V}{\Gamma \vdash (\lambda x. x)(\mathbf{v}_1 + \mathbf{v}_2) : U + V} \rightarrow_E$$

Notice that without the simultaneous forall/arrow elimination, it is not possible to type such a term.

2 Main Properties

The *Additive* type system is consistent, in the sense that typing is preserved by reduction (Theorem 4). Moreover, only terms with no infinite reduction are typable (Theorem 11).

The preservation of types by reduction, or *subject reduction* property, is proved by adapting the proof of Barendregt [18, Section 4.2] for the System F : we first define a binary relation \preceq on types, and then prove the usual generation and substitution lemmas (cf. Appendix A for more details).

Theorem 4 (Subject Reduction). *For any terms \mathbf{t}, \mathbf{t}' , any context Γ and any type T , if $\mathbf{t} \rightarrow^* \mathbf{t}'$ then $\Gamma \vdash \mathbf{t} : T \Rightarrow \Gamma \vdash \mathbf{t}' : T$.*

We also prove the strong normalisation property (i.e. no typable term has an infinite reduction) by adapting the standard method of *reducibility candidates* [19, Chap. 14] to the *Additive* type system. The idea is to interpret types by reducibility candidates, which are sets of strongly normalising terms. Then we show that as soon as a term has a type, it is in its interpretation, and thereby is strongly normalising.

We define here candidates as sets of *closed* terms. The set of all the closed terms is written Λ_0 , and SN_0 denotes the set of *strongly normalising* closed terms. In the following, we write $Red(\mathbf{t})$ for the set of reducts in one step of a term \mathbf{t} (with any of the six rules given in Sec. 1.1), and $Red_*(\mathbf{t})$ for the set of its reducts in any number of steps (including itself). Both notations are naturally extended to sets of terms. A term is a *pseudo value* when it is an abstraction or a sum of them: $\mathbf{b}, \mathbf{b}' ::= \lambda x. \mathbf{t} \mid \mathbf{b} + \mathbf{b}'$. A term that is not a pseudo value is said to be *neutral*, and we denote by \mathcal{N} the set of *closed neutral* terms.

Definition 5. *A set $\mathcal{S} \subseteq \Lambda_0$ is a reducibility candidate if it satisfies the three following conditions: (CR₁) Strong normalisation: $\mathcal{S} \subseteq SN_0$. (CR₂) Stability under reduction: $\mathbf{t} \in \mathcal{S} \Rightarrow Red(\mathbf{t}) \subseteq \mathcal{S}$. (CR₃) Stability under neutral expansion: If $\mathbf{t} \in \mathcal{N}$, then $Red(\mathbf{t}) \subseteq \mathcal{S}$ implies $\mathbf{t} \in \mathcal{S}$.*

We denote the reducibility candidates by \mathcal{A}, \mathcal{B} , and the set of all the reducibility candidates by \mathcal{RC} . Note that SN_0 is in \mathcal{RC} . In addition, the term $\mathbf{0}$ is a neutral term with no reduct, so it is in every reducibility candidate by (CR₃). Hence every reducibility candidate is non-empty.

Let $\overline{\mathcal{S}}$ be the closure of a set of terms \mathcal{S} by (CR₃). It can be defined inductively as follows: If $\mathbf{t} \in \mathcal{S}$, then $\mathbf{t} \in \overline{\mathcal{S}}$, and if $\mathbf{t} \in \mathcal{N}$ and $Red(\mathbf{t}) \subseteq \overline{\mathcal{S}}$, then $\mathbf{t} \in \overline{\mathcal{S}}$.

We can actually use this closure operator to define reducibility candidates:

Lemma 6. *If $\mathcal{S} \subseteq SN_0$, then $\overline{Red_*(\mathcal{S})} \in \mathcal{RC}$.*

In order to interpret types with reducibility candidates, we define the operators ‘arrow’, ‘plus’ and ‘intersection’ in \mathcal{RC} : Let $\mathcal{A}, \mathcal{B} \in \mathcal{RC}$. We define: $\mathcal{A} \rightarrow \mathcal{B} = \{\mathbf{t} \in \Lambda_0 / \forall \mathbf{u} \in \mathcal{A}, \mathbf{t}\mathbf{u} \in \mathcal{B}\}$ and $\mathcal{A} \mp \mathcal{B} = \overline{(\mathcal{A} + \mathcal{B}) \cup \mathcal{A} \cup \mathcal{B}}$ where $\mathcal{A} + \mathcal{B} = \{\mathbf{t} + \mathbf{u} / \mathbf{t} \in \mathcal{A} \text{ and } \mathbf{u} \in \mathcal{B}\}$.

Proposition 7. *Let $\mathcal{A}, \mathcal{B} \in \mathcal{RC}$. Then both $\mathcal{A} \rightarrow \mathcal{B}$ and $\mathcal{A} \mp \mathcal{B}$ are reducibility candidates. Moreover, if $(\mathcal{A}_i)_{i \in I}$ is a family of \mathcal{RC} , then $\bigcap_{i \in I} \mathcal{A}_i$ is a reducibility candidate.*

The operator $+$ is commutative and associative on terms, and hence so is the operator $+$ defined on sets of terms. Therefore, \mp is commutative and associative on reducibility candidates. In addition, $\bar{\emptyset}$ (a reducibility candidate according to Lemma 6) is neutral with respect to \mp . Lemma 8 formalises these properties.

Lemma 8. *Let $\mathcal{A}, \mathcal{B}, \mathcal{C} \in \mathcal{RC}$. Then $\mathcal{A} \mp \mathcal{B} = \mathcal{B} \mp \mathcal{A}$, $(\mathcal{A} \mp \mathcal{B}) \mp \mathcal{C} = \mathcal{A} \mp (\mathcal{B} \mp \mathcal{C})$ and $\mathcal{A} \mp \bar{\emptyset} = \mathcal{A}$.*

Type variables are interpreted using *valuations*, i.e. partial functions from type variables to reducibility candidates: $\rho := \emptyset \mid \rho, X \mapsto \mathcal{A}$. The interpretation $\llbracket T \rrbracket_\rho$ of a type T in a valuation ρ (that is defined for each free type variable of T) is given by

$$\begin{aligned} \llbracket X \rrbracket_\rho &= \rho(X) & \llbracket \bar{0} \rrbracket_\rho &= \bar{\emptyset} \\ \llbracket U \rightarrow T \rrbracket_\rho &= \llbracket U \rrbracket_\rho \rightarrow \llbracket T \rrbracket_\rho & \llbracket T + R \rrbracket_\rho &= \llbracket T \rrbracket_\rho \mp \llbracket R \rrbracket_\rho \\ \llbracket \forall X. T \rrbracket_\rho &= \bigcap_{\mathcal{A} \in \mathcal{RC}} \llbracket T \rrbracket_{\rho, X \mapsto \mathcal{A}} \end{aligned}$$

Lemma 6 and Proposition 7 ensure that each type is interpreted by a reducibility candidate. Furthermore, Lemma 8 entails that this interpretation is well defined with respect to the type equivalences.

Lemma 9. *For any types T, T' , and any valuation ρ , if $T \equiv T'$ then $\llbracket T \rrbracket_\rho = \llbracket T' \rrbracket_\rho$.*

Adequacy Lemma. We show that this interpretation complies with typing judgements. Reducibility candidates deal with closed terms, whereas proving the adequacy lemma by induction requires the use of open terms with some assumptions on their free variables (which are ensured by the context). Therefore we use *substitutions* σ to close terms:

$$\sigma := \emptyset \mid x \mapsto u; \sigma \qquad \mathbf{t}_\emptyset = \mathbf{t} \quad , \quad \mathbf{t}_{x \mapsto u; \sigma} = \mathbf{t}\{u/x\}_\sigma.$$

Given a context Γ , we say that a substitution σ *satisfies* Γ for the valuation ρ (notation: $\sigma \in \llbracket \Gamma \rrbracket_\rho$) when $(x : T) \in \Gamma$ implies $\sigma(x) \in \llbracket T \rrbracket_\rho$. A typing judgement $\Gamma \vdash \mathbf{t} : T$ is said to be *valid* (notation $\Gamma \vDash \mathbf{t} : T$) if for every valuation ρ , and for every substitution σ satisfying Γ for ρ , we have $\mathbf{t}_\sigma \in \llbracket T \rrbracket_\rho$.

Proposition 10 (Adequacy). *Every derivable typing judgement is valid: for each Γ , each term \mathbf{t} and each type T , we have that $\Gamma \vdash \mathbf{t} : T$ implies $\Gamma \vDash \mathbf{t} : T$.*

This immediately provides the strong normalisation result:

Theorem 11 (Strong normalisation). *Every typable term in Additive is strongly normalising.*

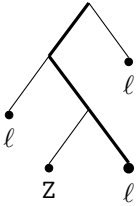
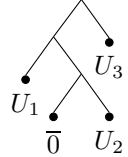
Proof. If a term \mathbf{t} is typable by a type T , then the adequacy lemma ensures that $\mathbf{t} \in \llbracket T \rrbracket_\emptyset$. As a reducibility candidate, $\llbracket T \rrbracket_\emptyset$ is included in SN_0 , and thus \mathbf{t} is strongly normalising. \square

3 Logical Interpretation

In this section, we interpret the *Additive* type system into System F with pairs (System F_P for short). Sum types are interpreted with Cartesian products. Since this product is neither associative nor commutative in System F_P , we first consider *Additive* without type equivalences. This involves a slightly modified but equivalent type system, that we call Add_{str} . We then translate every term of Add_{str} into a term of System F_P . Finally, we show that our translation is correct with respect to typing in *Additive* (Theorem 17) and reduction (Theorem 18).

Structured Additive Type System. The system Add_{str} is defined with the same grammar of types as *Additive*, and the same rules $ax, ax_{\bar{0}}, \rightarrow_I, +_I, \forall_I$ and \forall_E . There is no type equivalence, and thereby no commutativity nor associativity for sums (also $\bar{0}$ is not neutral for sums). Hence rule \rightarrow_E , has to be precised. To specify what an n -ary sum is, we introduce the structure of trees for types.

Example 12. In Add_{str} , the type $(U_1 + (\bar{0} + U_2)) + U_3$ is no longer equivalent to $U_1 + (U_2 + U_3)$. We can represent the first one by the labelled tree on the right.



To formalise Add_{str} , we use the standard representation of binary trees, with some special leaves ℓ (which can be labelled by a unit type): $\mathcal{T}, \mathcal{T}' := \ell \mid \mathbf{Z} \mid \mathcal{S}(\mathcal{T}, \mathcal{T}')$.

Each leaf is denoted by the finite word on the alphabet $\{1, r\}$ (for *left* and *right*) representing the path from the root of the tree. For instance, the type $(U_1 + (\bar{0} + U_2)) + U_3$ is obtained using the labelling $\{11 \mapsto U_1, 1rr \mapsto U_2, r \mapsto U_3\}$, with the tree of the left.

We say that a labelling function s (formally, a partial function from $\{1, r\}^*$ to unit types) *labels a tree* \mathcal{T} when each of its leaves ℓ is in the domain of s . In this case, we write $\mathcal{T}[s]$ the type of Add_{str} obtained by labelling \mathcal{T} with s . Notice that conversely, for any type T , there exists a unique tree \mathcal{T}_T and a labelling function s_T such that $T = \mathcal{T}_T[s_T]$. The tree composition $\mathcal{T} \circ \mathcal{T}'$ consists in “branching” \mathcal{T}' to each leaf ℓ of \mathcal{T} (cf. Example 25 in Appendix B.1). By extending the definition of labelling functions to functions from leaves to types, we have $\mathcal{T}[w \mapsto \mathcal{T}'[s]] = \mathcal{T} \circ \mathcal{T}'[wv \mapsto s(v)]$, where w denotes a ℓ -leaf of \mathcal{T} , and v a ℓ -leaf of \mathcal{T}' . Then the rule for the arrow elimination in Add_{str} is:

$$\frac{\Gamma \vdash \mathbf{t} : \mathcal{T}[w \mapsto \forall \vec{X}. (U \rightarrow T_w)] \quad \Gamma \vdash \mathbf{u} : \mathcal{T}'[v \mapsto U[\vec{V}_v / \vec{X}]]}{\Gamma \vdash \mathbf{tu} : \mathcal{T} \circ \mathcal{T}'[wv \mapsto T_w[\vec{V}_v / \vec{X}]]} \rightarrow_{E'}$$

where wv is a word whose prefix w represents a leaf of \mathcal{T} (cf. Example 26).

Proposition 13 (Additive equivalent to Add_{str}). $\Gamma \vdash \mathbf{t} : T$ is derivable in *Additive* if and only if there is a type $T' \equiv T$ such that $\Gamma \vdash \mathbf{t} : T'$ is derivable in Add_{str} .

Translation into the System F with Pairs. We recall the syntax of System F_P [11]:

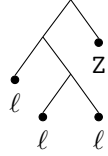
$$\begin{array}{ll} \mathbf{Terms} : & t, u := x \mid \lambda x.t \mid tu \mid \star \mid \langle t, u \rangle \mid \pi_1(t) \mid \pi_r(t) \\ \mathbf{Types} : & A, B := X \mid A \Rightarrow B \mid \forall X.A \mid \mathbf{1} \mid A \times B \end{array}$$

(reduction and typing rules are well known, cf. Fig. 2 on Appendix B).

In the same way than for the types, we define a term of System F_P with a tree (whose binary nodes S are seen as pairs) and a partial function τ from $\{\mathbf{1}, \mathbf{r}\}^*$ to F_P -terms. We write $\pi_{\alpha_1 \dots \alpha_n}(t)$ for $\pi_{\alpha_1}(\pi_{\alpha_2}(\dots \pi_{\alpha_n}(t)))$ (with $\alpha_i \in \{\mathbf{1}, \mathbf{r}\}$). Remark that if $t = \mathcal{T}[\tau]$ and w is a ℓ -leaf of \mathcal{T} , then $\tau(w)$ is a subterm of t that can be obtained by reducing $\pi_{\bar{w}}(t)$, where \bar{w} is the mirror word of w (cf. Example 14).

Example 14 (Representation of F_P -terms with trees).

Let $t = \langle \langle u_1, \langle u_2, u_3 \rangle \rangle, \star \rangle$. Then $t = \mathcal{T}[\mathbf{1}\mathbf{l} \mapsto u_1, \mathbf{1}\mathbf{r}\mathbf{l} \mapsto u_2, \mathbf{1}\mathbf{r}\mathbf{r} \mapsto u_3]$ (where \mathcal{T} is the tree on the right) and u_3 reduces from $\pi_{221}(t)$.



Every type T is interpreted by a type $|T|$ of System F_P .

$$\begin{array}{lll} |X| = X, & |\bar{\mathbf{0}}| = \mathbf{1}, & |\forall X.U| = \forall X.|U|, \\ |U \rightarrow T| = |U| \Rightarrow |T|, & |T + R| = |T| \times |R|. \end{array}$$

Then any term \mathbf{t} typable with a derivation \mathcal{D} is interpreted by a F_P -term $[\mathbf{t}]_{\mathcal{D}}$:

$$\text{If } \mathcal{D} = \frac{}{\Gamma, x : T \vdash x : T} ax, \text{ then } [x]_{\mathcal{D}} = x.$$

$$\text{If } \mathcal{D} = \frac{}{\Gamma \vdash \mathbf{0} : \bar{\mathbf{0}}} ax\bar{\mathbf{0}}, \text{ then } [\mathbf{0}]_{\mathcal{D}} = \star.$$

$$\text{If } \mathcal{D} = \frac{\mathcal{D}_1 \quad \mathcal{D}_2}{\Gamma \vdash \mathbf{t} + \mathbf{u} : T + R} +_I, \text{ then } [\mathbf{t} + \mathbf{u}]_{\mathcal{D}} = \langle [\mathbf{t}]_{\mathcal{D}_1}, [\mathbf{u}]_{\mathcal{D}_2} \rangle.$$

$$\text{If } \mathcal{D} = \frac{\mathcal{D}'}{\Gamma \vdash \lambda x.\mathbf{t} : U \rightarrow T} \rightarrow_I, \text{ then } [\lambda x.\mathbf{t}]_{\mathcal{D}} = \lambda x.[\mathbf{t}]_{\mathcal{D}'}$$

$$\text{If } \mathcal{D} = \frac{\mathcal{D}_1 \quad \mathcal{D}_2}{\Gamma \vdash \mathbf{t}\mathbf{u} : \mathcal{T} \circ \mathcal{T}'[wv \mapsto T_w[\bar{V}_v/\bar{X}]]} \rightarrow_{E'},$$

then $[\mathbf{t}\mathbf{u}]_{\mathcal{D}} = \mathcal{T} \circ \mathcal{T}'[wv \mapsto \pi_{\bar{w}}([\mathbf{t}]_{\mathcal{D}_1})\pi_{\bar{v}}([\mathbf{u}]_{\mathcal{D}_2})]$.

$$\text{If } \mathcal{D} = \frac{\mathcal{D}'}{\Gamma \vdash \mathbf{t} : \forall X.U} \forall_I, \text{ then } [\mathbf{t}]_{\mathcal{D}} = [\mathbf{t}]_{\mathcal{D}'}$$

$$\text{If } \mathcal{D} = \frac{\mathcal{D}'}{\Gamma \vdash \mathbf{t} : U[V/X]} \forall_E, \text{ then } [\mathbf{t}]_{\mathcal{D}} = [\mathbf{t}]_{\mathcal{D}'}$$

This interpretation is in fact a direct translation of sums by pairs at each step of the derivation, except for the application: informally, all the distributivity redexes are reduced before the translation of a term $\mathbf{t}\mathbf{u}$, which requires to ‘know’ the sum structure of \mathbf{t} and \mathbf{u} . This structure is actually given by their type, and that is why we can only interpret typed terms.

Example 15. If \mathbf{t} has type $(U \rightarrow T_1) + (U \rightarrow T_2)$ and \mathbf{u} has type $(U + \bar{\mathbf{0}}) + U$, then we see them as terms of shape $\mathbf{t}_1 + \mathbf{t}_2$ and $(\mathbf{u}_1 + \mathbf{0}) + \mathbf{u}_2$ respectively (the reducibility model of section 2 ensures that they actually reduce to

terms of this shape). Indeed, the translation of \mathbf{tu} reduces to the translation of $((\mathbf{t}_1\mathbf{u}_1) + \mathbf{0}) + \mathbf{t}_1\mathbf{u}_2) + (((\mathbf{t}_2\mathbf{u}_1) + \mathbf{0}) + \mathbf{t}_2\mathbf{u}_2)$:

$$[\mathbf{tu}]_{\mathcal{D}} = \langle \langle \langle t_1u_1, \star \rangle, t_1u_2 \rangle, \langle \langle t_2u_1, \star \rangle, t_2u_2 \rangle \rangle,$$
 where $t_1 = \pi_{11}([\mathbf{t}]_{\mathcal{D}_1})$, $t_2 = \pi_{21}([\mathbf{t}]_{\mathcal{D}_1})$, $u_1 = \pi_1([\mathbf{u}]_{\mathcal{D}_2})$, and $u_2 = \pi_{12}([\mathbf{u}]_{\mathcal{D}_2})$

Theorem 16 (Correction with respect to typing). *If a judgement $\Gamma \vdash \mathbf{t} : T$ is derivable in Add_{str} with derivation \mathcal{D} , then $|\Gamma| \vdash_F [\mathbf{t}]_{\mathcal{D}} : |T|$.*

The technical details for its proof are given in Appendix B.2. In Appendix B.3 it is given a theorem showing that the translation is not trivial since it is reversible.

To return back to *Additive*, observe that if $T \equiv T'$, their translations are equivalent in System F_P (in the sense that there exists two terms establishing an isomorphism between them), and conclude with Proposition I.3.

Theorem 17. *If a judgement $\Gamma \vdash \mathbf{t} : T$ is derivable in *Additive*, then there is a term t' of System F_P such that $|\Gamma| \vdash_F t' : |T|$.*

To some extent, the translation from Add_{str} to System F_P is also correct with respect to reduction (technical details for its proof in Appendix B.4).

Theorem 18 (Correction with respect to reduction). *Let $\Gamma \vdash \mathbf{t} : T$ be derivable (by \mathcal{D}) in Add_{str} , and $\mathbf{t} \rightarrow \mathbf{u}$. If the reduction is not due to rule $\mathbf{t} + \mathbf{0} \rightarrow \mathbf{t}$, then there is \mathcal{D}' deriving $\Gamma \vdash \mathbf{u} : T$, and $[\mathbf{t}]_{\mathcal{D}} \rightarrow^+ [\mathbf{u}]_{\mathcal{D}'}$.*

Notice that the associativity and commutativity of types have their analogous in the term equivalences. However, the equivalence $T + \mathbf{0} \equiv T$ has its analogous with a reduction rule, $\mathbf{t} + \mathbf{0} \rightarrow \mathbf{t}$. Since Add_{str} has no equivalences, this reduction rule is not correct in the translation. However, if $\Gamma \vdash \mathbf{t} + \mathbf{0} : T + \bar{\mathbf{0}}$ is derivable by \mathcal{D} in Add_{str} , then there is some $\mathcal{D}' = \Gamma \vdash \mathbf{t} : T$ such that $\varepsilon_{|T+\bar{\mathbf{0}}|,|T|}[\mathbf{t} + \mathbf{0}]_{\mathcal{D}} \rightarrow^* [\mathbf{t}]_{\mathcal{D}'}$, where $\varepsilon_{|T+\bar{\mathbf{0}}|,|T|}$ and $\varepsilon_{|T|,|T+\bar{\mathbf{0}}|}$ are the terms establishing the isomorphism between $|T|$ and $|T + \bar{\mathbf{0}}|$ in System F_P .

4 Conclusion

In this paper we considered an extension to call-by-value lambda calculus with a non-deterministic (or algebraic) operator $+$, and we mimiced its behaviour at the level of types. As we discussed in the introduction, this operator behaves like the algebraic sum with linear functions: $\mathbf{f}(x + y) = \mathbf{f}(x) + \mathbf{f}(y)$. However, our system is simulated by System F with pairs, which corresponds to the *non linear* fragment of IMELL.

This puts in the foreground the deep difference between the linearity in the algebraic sense (the one of Linear Logic), and the linearity of *Additive* (which is the same, for instance, as Lineal [5]). In the first case, a function is linear if it does not *duplicate* its argument x (that is, x^2 –or xx – will not appear during the computation), whereas in *Additive* a linear behaviour is achieved by banning sum terms substitutions: while computing $(\lambda x.\mathbf{t})(\mathbf{u} + \mathbf{s})$, the argument $(\mathbf{u} + \mathbf{s})$

will never be duplicated even if \mathbf{t} is not linear in x . We can only duplicate values (that intuitively correspond to constants in the algebraic setting, so their duplication does not break linearity). Actually, in *Additive*, the application is always distributed over the sum before performing the β -reduction, and these both reductions do not interact. This is what our translation shows: all distributivity rules are simulated *during* the translation (of the application), and then the β -reduction is simulated in System F , without paying any attention to the linearity.

As mentioned in the introduction, Lineal was meant for quantum computing and forcing the left distributivity is useful to prevent cloning. Moreover, it makes perfectly sense to consider any function as linear in this setting, since every quantum operator is given by a matrix, and thereby is linear. A CBV reduction for this kind of calculus is thus entirely appropriate.

Acknowledgements. We would like to thank Olivier Laurent for the useful advice he gave us about the interpretation we present in this paper, as well as Pablo Arrighi for the fruitful discussions about Lineal and its linearity.

References

1. Boudol, G.: Lambda-calculi for (strict) parallel functions. *Information and Computation* 108(1), 51–127 (1994)
2. Bucciarelli, A., Ehrhard, T., Manzonetto, G.: A relational semantics for parallelism and non-determinism in a functional setting. *Annals of Pure and Applied Logic* 163(7), 918–934 (2012)
3. Dezani-Ciancaglini, M., de'Liguoro, U., Piperno, A.: Filter models for conjunctive-disjunctive lambda-calculi. *Theoretical Computer Science* 170(1-2), 83–128 (1996)
4. Dezani-Ciancaglini, M., de'Liguoro, U., Piperno, A.: A filter model for concurrent lambda-calculus. *SIAM Journal on Computing* 27(5), 1376–1419 (1998)
5. Arrighi, P., Dowek, G.: Linear-algebraic Lambda-calculus: Higher-order, Encodings, and Confluence. In: Voronkov, A. (ed.) *RTA 2008. LNCS*, vol. 5117, pp. 17–31. Springer, Heidelberg (2008)
6. Vaux, L.: The algebraic lambda calculus. *Mathematical Structures in Computer Science* 19(5), 1029–1059 (2009)
7. Assaf, A., Perdrix, S.: Completeness of algebraic cps simulations. In: *Proceedings of the 7th International Workshop on Developments of Computational Models (DCM 2011)*, Zurich, Switzerland (2011), http://www.pps.univ-paris-diderot.fr/~jkrivine/conferences/DCM2011/DCM_2011.html
8. Díaz-Caro, A., Perdrix, S., Tasson, C., Valiron, B.: Equivalence of algebraic λ -calculi. In: *Informal Proceedings of the 5th International Workshop on Higher-Order Rewriting, HOR 2010*, Edinburgh, UK, pp. 6–11 (July 2010)
9. Hennessy, M.: The semantics of call-by-value and call-by-name in a nondeterministic environment. *SIAM Journal on Computing* 9(1), 67–84 (1980)
10. Girard, J.Y.: Linear logic. *Theoretical Computer Science* 50, 1–102 (1987)
11. Cosmo, R.D.: *Isomorphisms of Types: From Lambda-Calculus to Information Retrieval and Language Design*. Progress in Theoretical Computer Science. Birkhäuser (1995)

12. Wootters, W.K., Zurek, W.H.: A single quantum cannot be cloned. *Nature* 299, 802–803 (1982)
13. Monroe, C., Meekhof, D.M., King, B.E., Itano, W.M., Wineland, D.J.: Demonstration of a fundamental quantum logic gate. *Physical Review Letters* 75(25), 4714–4717 (1995)
14. Plotkin, G.D.: Call-by-name, call-by-value and the λ -calculus. *Theoretical Computer Science* 1(2), 125–159 (1975)
15. Jouannaud, J.P., Kirchner, H.: Completion of a set of rules modulo a set of equations. *SIAM Journal on Computing* 15(4), 1155–1194 (1986)
16. Barendregt, H.P.: The lambda calculus: its syntax and semantics. *Studies in Logic and the Foundations of Mathematics*, vol. 103. Elsevier (1984)
17. Díaz-Caro, A., Dowek, G.: Non determinism through type isomorphism. Draft (April 2012), <http://diaz-caro.info/ndti.pdf>
18. Barendregt, H.P.: Lambda calculi with types. In: *Handbook of Logic in Computer Science*, vol. 2. Oxford University Press (1992)
19. Girard, J.-Y., Lafont, Y., Taylor, P.: *Proofs and Types*. Cambridge University Press (1989)
20. Krivine, J.L.: *Lambda-calcul: types et modèles. Études et recherches en informatique*, Masson (1990)

A Formalisation of the Proof of Subject Reduction

The preservation of types by reduction, or *subject reduction* property, is proved by adapting the proof of Barendregt [18, Section 4.2] for the Sytem F : we first define a binary relation \preccurlyeq on types, and then we give the usual generation and substitution lemmas. Finally, we give a needed property (Lemma 24) for the typing of $\mathbf{0}$ and values.

Definition 19 (Relation \preccurlyeq on types)

- Given two types U_1 and U_2 , we write $U_1 \prec U_2$ if either
 - $U_2 \equiv \forall X.U_1$ or
 - $U_1 \equiv \forall X.U'$ and $U_2 \equiv U'[T/X]$ for some type T .
- We write \preccurlyeq the reflexive (with respect to \equiv) transitive closure of \prec .

The following property says that if two arrow types are related by \preccurlyeq , then they are equivalent up to substitutions.

Lemma 20 (Arrow comparison). *For any unit types U, U' and types T, T' , if $U' \rightarrow T' \preccurlyeq U \rightarrow T$, then there exist \vec{V}, \vec{X} such that $U \rightarrow T \equiv (U' \rightarrow T')[\vec{V}/\vec{X}]$.*

As a pruned version of a subtyping system, we can prove the subsumption rule:

Lemma 21 (\preccurlyeq -subsumption). *For any context Γ , any term \mathbf{t} and any unit types U, U' such that $U \preccurlyeq U'$ and no free type variable in U occurs in Γ , if $\Gamma \vdash \mathbf{t} : U$ then $\Gamma \vdash \mathbf{t} : U'$.*

Generation lemmas allows to study the conclusion of a derivation so as to understand where it may come from, thereby decomposing the term in its basic constituents.

Lemma 22 (Generation lemmas). *For any context Γ , any terms \mathbf{t}, \mathbf{u} , and any type T ,*

1. $\Gamma \vdash \mathbf{t}\mathbf{u}:T$ implies $\Gamma \vdash \mathbf{t}: \sum_{i=1}^n \forall \vec{X}.(U \rightarrow T_i)$ and $\Gamma \vdash \mathbf{u}: \sum_{j=1}^m U[\vec{V}_j/\vec{X}]$ for some integers n, m , some types T_1, \dots, T_n , and some unit types $U, \vec{V}_1, \dots, \vec{V}_m$ such that $\sum_{i=1}^n \sum_{j=1}^m T_i[\vec{V}_j/\vec{X}] \preceq T$.
2. $\Gamma \vdash \lambda x.\mathbf{t}:T$ implies $\Gamma, x:U \vdash \mathbf{t}:R$ for some types U, R such that $U \rightarrow R \preceq T$.
3. $\Gamma \vdash \mathbf{t} + \mathbf{u}:T$ implies $\Gamma \vdash \mathbf{t}:R$ and $\Gamma \vdash \mathbf{u}:S$ with for some types R, S such that $R + S \equiv T$.

The following lemma is standard in proofs of subject reduction, and can be found for example in [18, Prop. 4.1.19] and [20, Props. 8.2 and 8.5]. It ensures than by substituting type variables for types or term variables for terms in an adequate manner, the type derived is still valid.

Lemma 23 (Substitution). *For any Γ, T, U, \mathbf{v} and \mathbf{t} ,*

1. $\Gamma \vdash \mathbf{t}:T$ implies $\Gamma[U/X] \vdash \mathbf{t}:T[U/X]$.
2. If $\Gamma, x:U \vdash \mathbf{t}:T$, and $\Gamma \vdash \mathbf{v}:U$, then $\Gamma \vdash \mathbf{t}\{\mathbf{v}/x\}:T$.

Finally we need a property showing that $\mathbf{0}$ is only typed by $\bar{\mathbf{0}}$ and its equivalent types, and values are always typed by unit types or equivalent.

Lemma 24 (Typing $\mathbf{0}$ and values)

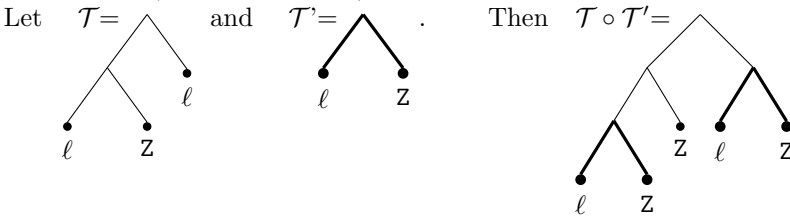
1. For any Γ , if $\Gamma \vdash \mathbf{0}:T$ then $T \equiv \bar{\mathbf{0}}$.
2. For any value \mathbf{v} (i.e. a variable or an abstraction), if $\Gamma \vdash \mathbf{v}:T$ then T is necessarily equivalent to a unit type.

Using all the previous lemmas, the proof of subject reduction is made by induction on typing derivation.

B Formalisation of the Translation into System F

B.1 Some Examples

Example 25 (Tree composition).

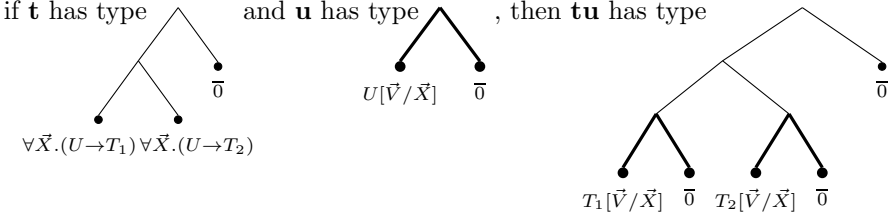


Example 26 (Arrow elimination rule in Add_{str}).

The following derivation is correct:

$$\frac{\Gamma \vdash \mathbf{t}: (\forall \vec{X}.(U \rightarrow T_1) + \forall \vec{X}.(U \rightarrow T_2)) + \bar{\mathbf{0}} \quad \Gamma \vdash \mathbf{u}: U[\vec{V}/\vec{X}] + \bar{\mathbf{0}}}{\Gamma \vdash \mathbf{t}\mathbf{u}: ((T_1[\vec{V}/\vec{X}] + \bar{\mathbf{0}}) + (T_2[\vec{V}/\vec{X}] + \bar{\mathbf{0}})) + \bar{\mathbf{0}}} \rightarrow_{E'}$$

Graphically, we can represent this rule as follows:



B.2 Soundness with Respect to Typing

We need first some lemmas and definitions. It can be immediately checked that the tree structure of a type is preserved by translation, as expressed in the following lemma.

Lemma 27. *If $T = \mathcal{T}[w \mapsto U_w]$ is a type of Add_{str} , then $|T| = \mathcal{T}[w \mapsto |U_w|]$.*

Definition 28. *We call F-labelling a function defined from leaves to types of System F_P . Given ϕ , an F-labelling, and \mathcal{T} , a tree, the type $\mathcal{T}[\phi]$ of System F_P is defined as expected:*

$$\ell[\phi] = \phi(\varepsilon), \quad \mathbf{Z}[\phi] = \mathbf{1}, \quad \mathbf{S}(\mathcal{T}, \mathcal{T}')[\phi] = \mathcal{T}[w \mapsto \phi(\mathbf{1}w)] \times \mathcal{T}'[w \mapsto \phi(\mathbf{r}w)]$$

There is a trivial relation between the term-labelling of a tree, and its F-labelling, that we give in the following lemma.

Reduction rules :

$$\begin{array}{l} (\lambda x.t)u \rightarrow t\{u/x\} \quad ; \quad \pi_i(\langle t_1, t_2 \rangle) \rightarrow t_i \\ \lambda x.tx \rightarrow t \quad (\text{if } x \notin FV(t)) \quad ; \quad \langle \pi_1(p), \pi_r(p) \rangle \rightarrow p \end{array}$$

Typing rules :

$$\begin{array}{l} \frac{}{\Delta, x : A \vdash_F x : A}^{Ax} \quad ; \quad \frac{}{\Delta \vdash_F \star : \mathbf{1}}^{\mathbf{1}} \quad ; \quad \frac{\Delta, x : A \vdash_F t : B}{\Delta \vdash_F \lambda x.t : A \Rightarrow B}^{\Rightarrow I} \\ \frac{\Delta \vdash_F t : A \Rightarrow B \quad \Delta \vdash_F u : A}{\Delta \vdash_F tu : B}^{\Rightarrow E} \quad ; \quad \frac{\Delta \vdash_F t : A \quad \Delta \vdash_F u : B}{\Delta \vdash_F \langle t, u \rangle : A \times B}^{\times I} \\ \frac{\Delta \vdash_F t : A \times B}{\Delta \vdash_F \pi_1(t) : A}^{\times E_1} \quad ; \quad \frac{\Delta \vdash_F t : A \times B}{\Delta \vdash_F \pi_r(t) : B}^{\times E_r} \\ \frac{\Delta \vdash_F t : A \quad X \notin FV(\Delta)}{\Delta \vdash_F t : \forall X.A}^{\forall I} \quad ; \quad \frac{\Delta \vdash_F t : \forall X.A}{\Delta \vdash_F t : A[B/X]}^{\forall E} \end{array}$$

Fig. 2. System F with pairs

Lemma 29. *Let \mathcal{T} be a tree.*

1. *If $\Gamma \vdash_F t_w : A_w$ for each ℓ -leave w , then $\Gamma \vdash_F \mathcal{T}[w \mapsto t_w] : \mathcal{T}[w \mapsto A_w]$.*
2. *If $\Gamma \vdash_F t : \mathcal{T}[w \mapsto A_w]$, then for each ℓ -leaf of \mathcal{T} , $\Gamma \vdash_F \pi_{\overline{w}}(t) : A_w$.*

Theorem 16 (Correction with respect to typing). *If a judgement $\Gamma \vdash \mathbf{t} : T$ is derivable in Add_{str} with derivation \mathcal{D} , then $|\Gamma| \vdash_F [\mathbf{t}]_{\mathcal{D}} : |T|$.*

Proof. We prove this proposition by induction on the derivation \mathcal{D} . If it ends with rule ax or $ax_{\overline{0}}$, we use rule Ax or **1** respectively in System F_P . If the last rule of \mathcal{D} is $+_I$ or \rightarrow_I we can conclude by induction. If the last rule is \forall_I , we just need to note that $X \notin FV(\Gamma)$ implies $X \notin FV(|\Gamma|)$. If it is the rule \forall_E , we just have to note that $|U[V/X]| = |U[|V|/X]|$ to conclude with induction hypothesis. The only interesting case is when \mathcal{D} ends with rule $\rightarrow_{E'}$:

$$\mathcal{D} = \frac{\Gamma \vdash \mathbf{t} : \mathcal{T}[w \mapsto \forall \vec{X}.(U \rightarrow T_w)] \quad \Gamma \vdash \mathbf{u} : \mathcal{T}'[v \mapsto U[\vec{V}_v/\vec{X}]]}{\Gamma \vdash \mathbf{tu} : \mathcal{T} \circ \mathcal{T}'[wv \mapsto T_w[\vec{V}_v/\vec{X}]}}$$

By induction hypothesis, $|\Gamma| \vdash_F [\mathbf{t}]_{\mathcal{D}_1} : |\mathcal{T}[w \mapsto \forall \vec{X}.(U \rightarrow T_w)]|$ and $|\Gamma| \vdash_F [\mathbf{u}]_{\mathcal{D}_2} : |\mathcal{T}'[v \mapsto U[\vec{V}_v/\vec{X}]]|$. By Lemma 27, it means that $|\Gamma| \vdash_F [\mathbf{t}]_{\mathcal{D}_1} : \mathcal{T}[w \mapsto \forall \vec{X}.|U| \Rightarrow |T_w|]$ and $|\Gamma| \vdash_F [\mathbf{u}]_{\mathcal{D}_2} : \mathcal{T}'[v \mapsto |U[|\vec{V}_v|/\vec{X}]]|$. By Lemma 29.2, for every ℓ -leaf w of \mathcal{T} , and every ℓ -leaf v of \mathcal{T}' , we can derive

$$\frac{\frac{|\Gamma| \vdash_F \pi_{\overline{w}}([\mathbf{t}]_{\mathcal{D}_1}) : \forall \vec{X}.|U| \Rightarrow |T_w|}{|\Gamma| \vdash_F \pi_{\overline{w}}([\mathbf{t}]_{\mathcal{D}_1}) : |U[|\vec{V}_v|/\vec{X}] \Rightarrow |T_w[|\vec{V}_v|/\vec{X}]} \quad |\Gamma| \vdash_F \pi_{\overline{v}}([\mathbf{u}]_{\mathcal{D}_2}) : |U[|\vec{V}_v|/\vec{X}]|}{|\Gamma| \vdash_F \pi_{\overline{w}}([\mathbf{t}]_{\mathcal{D}_1}) \pi_{\overline{v}}([\mathbf{u}]_{\mathcal{D}_2}) : |T_w[|\vec{V}_v|/\vec{X}]|}$$

Since $[\mathbf{tu}]_{\mathcal{D}} = \mathcal{T} \circ \mathcal{T}'[wv \mapsto \pi_{\overline{w}}([\mathbf{t}]_{\mathcal{D}_1}) \pi_{\overline{v}}([\mathbf{u}]_{\mathcal{D}_2})]$, by Lemma 29.1 we can conclude $|\Gamma| \vdash_F [\mathbf{tu}]_{\mathcal{D}} : \mathcal{T} \circ \mathcal{T}'[wv \mapsto |T_w[|\vec{V}_v|/\vec{X}]]|$, and then conclude using Lemma 27 again. \square

B.3 Partial Translation from System F_P to Add_{str}

To show that the translation from Add_{str} to System F_P is meaningful and non trivial, we define a partial encoding from System F_P to Add_{str} , and prove that it is the inverse of the previous translation. We define inductively the partial function (\cdot) from the types of System F_P to those of Add_{str} , as follows.

$$(\!|X|\!) = X \quad \text{and} \quad (\!|\mathbf{1}|\!) = \overline{0};$$

if $(\!|A|\!)$, $(\!|A'|\!)$ and $(\!|B|\!)$ are defined, then

$$(\!|\forall X.A|\!) = \forall X.(\!|A|\!) \quad \text{and} \quad (\!|A \times B|\!) = (\!|A|\!) + (\!|B|\!);$$

and if also $(\!|A'|\!) \in U$, then $(\!|A' \Rightarrow B|\!) = (\!|A'|\!) \rightarrow (\!|B|\!)$.

This translation is extended to contexts in the usual way. Similarly, we define a partial function from terms of System $F_{\mathcal{P}}$ to those of Add_{str} :

$$\begin{aligned} \langle x \rangle &= x \quad ; \quad \langle \lambda x.t \rangle = \lambda x.\langle t \rangle \quad ; \quad \langle tu \rangle = \langle t \rangle \langle u \rangle \quad ; \quad \langle \star \rangle = \mathbf{0} \quad ; \\ \langle \mathcal{T}[wv \mapsto \pi_{\overline{w}}(t)\pi_{\overline{v}}(u)] \rangle &= \langle t \rangle \langle u \rangle \quad \text{if } \mathcal{T} \neq \mathbf{Z} \text{ and } \mathcal{T} \neq \ell \quad ; \\ \langle \langle t_1, t_2 \rangle \rangle &= \langle t_1 \rangle + \langle t_2 \rangle \quad \text{if } \langle t_1, t_2 \rangle \neq \mathcal{T}[wv \mapsto \pi_{\overline{w}}(u)\pi_{\overline{v}}(u')] \text{ for any } \mathcal{T}, u, u' \end{aligned}$$

This defines the inverse of $[\cdot]_{\mathcal{D}}$, as specified by the following theorem.

Theorem 30. *If $\Gamma \vdash \mathbf{t} : T$ is derivable in Add_{str} with derivation \mathcal{D} , then $\langle \langle \Gamma \rangle \rangle \vdash \langle \langle \mathbf{t} \rangle_{\mathcal{D}} \rangle : \langle \langle T \rangle \rangle$ is syntactically the same sequent.*

B.4 Soundness with Respect to Reduction

First we need a substitution lemma for the translation of terms.

Lemma 31. *Let $\mathcal{D}_1 = \Gamma, x:U \vdash \mathbf{t}:T$ and $\mathcal{D}_2 = \Gamma \vdash \mathbf{b}:U$, then $\exists \mathcal{D}_3$ such that $[\mathbf{t}]_{\mathcal{D}_1} \{[\mathbf{b}]_{\mathcal{D}_2}/x\} = [\mathbf{t}\{\mathbf{b}/x\}]_{\mathcal{D}_3}$.*

Theorem 18 (Correction with respect to reduction). *Let $\Gamma \vdash \mathbf{t} : T$ be derivable (by \mathcal{D}) in Add_{str} , and $\mathbf{t} \rightarrow \mathbf{u}$. If the reduction is not due to rule $\mathbf{t} + \mathbf{0} \rightarrow \mathbf{t}$, then there is \mathcal{D}' deriving $\Gamma \vdash \mathbf{u} : T$, and $[\mathbf{t}]_{\mathcal{D}} \rightarrow^+ [\mathbf{u}]_{\mathcal{D}'}$.*

Proof. The proof is long but straightforward using the previous lemmas. It follows by induction over \mathcal{D} . \square

Polynomial-Time Solution of Initial Value Problems Using Polynomial Enclosures

Amin Farjudian

Division of Computer Science
University of Nottingham Ningbo, China
Amin.Farjudian@nottingham.edu.cn
www.cs.nott.ac.uk/~avf/

Abstract. Domain theory has been used with great success in providing a semantic framework for Turing computability, over both discrete and continuous spaces. On the other hand, classical approximation theory provides a rich set of tools for computations over real functions with (mainly) polynomial and rational function approximations.

We present a semantic model for computations over real functions based on *polynomial enclosures*. As an important case study, we analyse the convergence and complexity of Picard's method of initial value problem solving in our framework. We obtain a geometric rate of convergence over Lipschitz fields and then, by using Chebyshev truncations, we modify Picard's algorithm into one which runs in *polynomial-time* over a set of polynomial-space representable fields, thus achieving a reduction in complexity which would be impossible in the step-function based domain models.

Keywords: computable analysis, computational complexity, initial value problem, Picard's method, approximation theory, Chebyshev series.

1 Introduction

In classical mathematical analysis, for the most part one abstracts away from effective representations of objects. The constructive view of analysis [6] brings in a distinction between finitely representable objects and those that can only be approximated with finitely representable ones. In computable analysis [25] one works at an even lower level of abstraction, where claims of existence are required to be proven via procedures that are implementable on a Turing machine.

For instance, whereas in classical analysis it is true that any initial value problem (IVP) with a continuous field has a solution, in computable analysis proper this claim is only valid when a solution can be worked out using some algorithm which generates approximations to the solution to within any desired accuracy.

As such, in partial orders one finds a natural setting in which essential concepts such as approximation and convergence can be formulated [13]. As a special subclass of partial orders, domains [2] have been studied extensively as a semantic model of computation. Cartesian closed categories of domains with an interval domain object [11] provide a denotational semantic framework for computations over continuous spaces.

There is a canonical way of constructing function spaces in the category of domains via *step functions* [2 Chapter 4]. For real function spaces this approach is essentially equivalent to approximating functions via piece-wise constant enclosures. This is theoretically sufficient as with this construction many of the concepts from classical analysis can be reformulated in a domain theoretic setting [8,9].

On the other hand, there is a long tradition in *approximation theory* [3,21] with a very rich literature, in which computations over real functions are reduced to those over their relatively simpler polynomial (or even rational function) approximations. By the classic theorems of Jackson [15] and Bernstein [4] there is a tight link between the analytic properties of a function and how it can be approximated with polynomials.

At a practical level, almost all widely used maths software libraries provide some kind of functionality based on approximation theory. In fact there are some that have been written exclusively based on function approximations. These libraries are mainly geared towards fast numerical computations based on the machine’s floating-point unit. This means that a rigorous analysis of (floating-point) inaccuracies or the theoretical complexity of computations beyond the reach of the machine’s resources is typically of a secondary concern.

Our aim is to present a semantic model *along the lines* of domain based ones, in which both the convergence and the complexity of computations can be studied. We will make sure that our framework is complete, i. e. all results can be approximated to within any desired accuracy. We pick IVP solving as an important case to study, and analyse the convergence and complexity of Picard’s method as adapted to our framework.

2 Polynomial Enclosures

In what follows, \mathbb{N}_+ denotes the set of positive natural numbers and $C[a, b]^n$ denotes the set of continuous functions from the n -dimensional cube $[a, b]^n$ to \mathbb{R} , where $a, b \in \mathbb{R}$, $n \in \mathbb{N}_+$ and $a \leq b$.

For $n \in \mathbb{N}_+$ and $f, g \in C[a, b]^n$ we define the function enclosure $[f, g]$ by

$$[f, g] := \{h \in C[a, b]^n \mid \forall \bar{x} \in [a, b]^n : f(\bar{x}) \leq h(\bar{x}) \leq g(\bar{x})\}$$

If $\exists \bar{x}_0 \in [a, b]^n : g(\bar{x}_0) < f(\bar{x}_0)$ then $[f, g]$ will be empty. The functions f and g are called the *lower* and the *upper boundaries* of the enclosure $[f, g]$, respectively. For each point $t \in [a, b]^n$, the (local) *width* of the enclosure $[f, g]$ at t is defined as $w_{[f,g]}(t) := g(t) - f(t)$. The (global) width of the enclosure is defined as $w([f, g]) := \max\{w_{[f,g]}(t) \mid t \in [a, b]^n\}$. Note that $w([f, g])$ is well defined as $[a, b]^n$ is compact and both f and g are assumed to be continuous.

Definition 1 ($\Gamma(h)$: graph of a function enclosure). Let $h = [f, g]$ be an enclosure where $f, g \in C[a, b]^n$. By the graph of h we mean the set of all points in $[a, b]^n \times \mathbb{R}$ lying between the graphs of its lower and upper boundaries, i. e.

$$\Gamma(h) := \{(\bar{x}, r) \in [a, b]^n \times \mathbb{R} \mid f(\bar{x}) \leq r \leq g(\bar{x})\}$$

¹ Such as the free and open source MATLAB library *chebfun* available from <http://www2.maths.ox.ac.uk/chebfun>

Definition 2 (<P, R>: centered rational polynomial enclosure). Let $n \in \mathbb{N}_+$ and assume that P and R are polynomials with rational coefficients in n variables X_1, \dots, X_n , i. e. $P, R \in \mathbb{Q}[X_1, \dots, X_n]$. By the centered rational polynomial enclosure <P, R> we mean the non-empty \square function enclosure $[f, g]$ in which $f, g \in C[a, b]^n$ and $\forall \bar{x} \in [a, b]^n : g(\bar{x}) = P(\bar{x}) + R(\bar{x}) \wedge f(\bar{x}) = P(\bar{x}) - R(\bar{x})$. We will refer to P and R as the center and the radius of the enclosure <P, R>, respectively.

As we have imposed the non-emptiness condition, then $\forall \bar{x} \in [a, b]^n : R(\bar{x}) \geq 0$. The purpose of restricting the coefficients of P and R to rational numbers has been to ensure that all of our enclosures are finitely representable.

Note that the notations $[f, g]$ and <P, R> for function enclosures are indeed interchangeable as we have $[f, g] = \langle (g + f)/2, (g - f)/2 \rangle$. In this paper there will be no decoupling of the boundaries of the enclosures in the sense that we will always add the error estimates to both the upper and the lower boundaries of an enclosure. Thus, we will opt for using the centered-enclosure notation as in Definition \square .

Remark 1. Throughout this paper, by a polynomial enclosure we will always mean a centered rational one.

For each $n \in \mathbb{N}_+$ we denote the set of all non-empty enclosures with boundaries in $C[a, b]^n$ by $\mathbb{FE}[a, b]^n$, i. e. $\mathbb{FE}[a, b]^n := \{[f, g] \mid f, g \in C[a, b]^n, \forall t \in [a, b]^n : f(t) \leq g(t)\}$. We define the order \sqsubseteq over this set as follows: $\forall h_1, h_2 \in \mathbb{FE}[a, b]^n : h_1 \sqsubseteq h_2 \Leftrightarrow h_2 \subseteq h_1$. The pair $(\mathbb{FE}[a, b]^n, \sqsubseteq)$ is a partial order which we simply denote by $\mathbb{FE}[a, b]^n$. The pair $(\mathbb{PE}[a, b]^n, \sqsubseteq)$ in which $\mathbb{PE}[a, b]^n$ is the set of polynomial enclosures also forms a poset under the order inherited from $\mathbb{FE}[a, b]^n$, which we simply write as $\mathbb{PE}[a, b]^n$.

A sequence $\langle [f_i, g_i] \rangle_{i \in \mathbb{N}}$ of enclosures is said to converge to $[f, g]$ if $\forall k \in \mathbb{N} : [f_k, g_k] \sqsubseteq [f, g], f = \lim_{i \rightarrow \infty} f_i$ and $g = \lim_{i \rightarrow \infty} g_i$, where the limits are taken with respect to the supremum norm, which is defined for each $f \in C[a, b]^n$ as $\|f\| = \sup\{f(\bar{x}) \mid \bar{x} \in [a, b]^n\}$. An element $h \in \mathbb{FE}[a, b]^n$ is said to be *maximal* if $w(h) = 0$, in which case $h = [f, f]$ for some $f \in C[a, b]^n$. For simplicity, we will identify the maximal element $[f, f]$ with f . Using this convention one may talk about sequences of *function enclosures* in $\mathbb{FE}[a, b]^n$ that converge to *functions* in $C[a, b]^n$.

Note that neither $\mathbb{FE}[a, b]^n$ nor $\mathbb{PE}[a, b]^n$ is complete under the notion of convergence just defined. For instance, consider the sequence $h_i = [f_i, g_i]$ of enclosures in $\mathbb{PE}[0, 1]$ defined as $\forall i \in \mathbb{N}, x \in [0, 1] : f_i(x) = 0, g_i(x) = x^i$. This sequence forms a chain as $\forall i \in \mathbb{N} : h_i \sqsubseteq h_{i+1}$, but ‘the limit’ of $\langle g_i \rangle_{i \in \mathbb{N}}$ is the non-continuous function $\gamma : [0, 1] \rightarrow \mathbb{R}$ which satisfies $\gamma(x) = 0$ if $x \in [0, 1)$ and $\gamma(1) = 1$.

3 Solving Initial Value Problems Using an Oracle Machine

Let $m \in \mathbb{N}_+$ and consider the initial value problem (IVP)

$$\begin{cases} y'(t) = F(t, y(t)) \\ y(t_0) = y_0 \end{cases} \tag{1}$$

² It will be interesting to see (in our future work) what more we may achieve by removing the non-emptiness condition.

in which $F : \Omega \rightarrow \mathbb{R}^m$ will be referred to as the *field* of the IVP, and for which we seek a solution $y : [t_0, a] \rightarrow \mathbb{R}^m$ for a suitable $a \geq t_0$. We assume that $\Omega \subseteq \mathbb{R}^{m+1}$ is open and includes the initial point, i. e. $(t_0, y_0) \in \Omega$. Peano’s existence theorem states that the mere continuity of the field F guarantees the *existence* of a solution [22]. Furthermore, if F is *Lipschitz continuous* in its second argument, i. e.

$$\exists L \in \mathbb{R} : \forall t \in \mathbb{R}, r_1, r_2 \in \mathbb{R}^m : \|F(t, r_1) - F(t, r_2)\|_{\text{sup}} \leq L\|r_1 - r_2\|_{\text{sup}}$$

then by Picard-Lindelöf theorem the IVP has a *unique* solution. The Lipschitz condition is sufficient but not necessary for the uniqueness [22]. A *necessary and sufficient condition* for the uniqueness of the solution was provided by Hiroshi Okamura, a generalisation of which can be found in [26].

If we integrate both sides of the differential equation in (1) and incorporate the initial condition, we obtain the following integral equation:

$$y(x) = y_0 + \int_{t_0}^x F(t, y(t)) dt \tag{2}$$

Assume that for some suitable $a \geq t_0$ and $b > 0$ satisfying $[t_0, a] \times [-b, b]^n \subseteq \Omega$ the operator $Pic_F(g) = \lambda x. y_0 + \int_{t_0}^x F(t, g(t)) dt$ is an endofunction over $[t_0, a] \times [-b, b]^n$. Then any fixed-point of Pic_F —if any does indeed exist—would be a solution to both the integral equation (2) and the IVP (1). In fact, Lipschitz continuity guarantees that the conditions for the Banach fixed-point theorem are satisfied, so all one needs to do is to apply Pic_F repeatedly to obtain better approximations of the solution, a process famously known as *Picard’s method of IVP solving*.

Assume that the field F and the initial point (t_0, y_0) are computable. Then under the Lipschitz assumption Picard’s method yields a computable solution, whereas without the Lipschitz assumption all of the solutions could turn out to be non-computable [1].

Remark 2. Throughout this paper, the only notion of computability over real numbers that we will consider will be that of the Type-2 Theory of Effectivity (TTE) [25].

From now on, without loss of generality, we will focus on the one-dimensional case and will consider the following IVP:

$$\begin{cases} y'(t) = F(t, y(t)) \\ y(0) = 0 \end{cases} \tag{3}$$

for which we seek a solution $y : [0, a] \rightarrow [-b, b]$, for suitable $a \geq 0$ and $b \geq 0$.

We draw inspiration from Ko’s work [17] and adopt an *oracle machine* model for a clear complexity analysis (as in Fig. 1). As we have simplified the initial condition to $(0, 0)$, the *solver* machine will only need:

1. a socket into which one plugs in a *field* oracle, i. e. an oracle which provides information about the field to the solver machine;
2. a pair of input and output tapes for interaction with the user (i. e. the outside world).

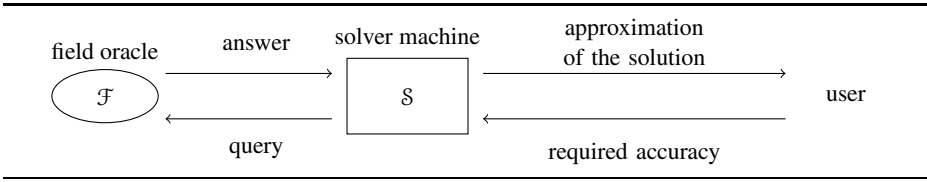


Fig. 1. An oracle machine for solving IVPs

The user sends the required accuracy to the solver machine \mathcal{S} in the form of a natural number k , and in return will be expecting a polynomial enclosure $\langle y_k, r_k \rangle$ of the solution y whose width over $[0, a]$ is smaller than 2^{-k} . The solver machine in turn tries to obtain enough information about the field from its oracle \mathcal{F} until it can provide the required approximation of the solution.

The advantage of using an oracle machine model is that we can abstract away from the cost of computing the field, and instead focus on the actual cost of IVP solving as performed by the solver machine. Nonetheless, the cost of handling queries and answers, and the cost of operations inside the solver machine *will* be taken into account.

Let us recount a rough sketch of how algorithms such as those of [10,12]—where step-functions are used to approximate real functions—would work in this setting. The solver starts with the most conservative rectangle enclosing the graph of the solution, i. e. $([0, a], [-b, b])$. At iteration n , the solver possesses an enclosure of the graph of the solution made up of 2^n rectangles $\{B_i^n \mid 0 \leq i < 2^n\}$, in which each rectangle B_i^n has got the horizontal side $[ai/2^n, a(i + 1)/2^n]$. The solver machine sends each of these rectangles as queries to the field oracle.

The field oracle \mathcal{F} possesses an ‘encoding’ of a domain theoretic extension \mathcal{IF} of the field F . As such, in response to each query B_i^n it returns an interval enclosure $\mathcal{IF}(B_i^n)$ of $F(B_i^n)$, i. e. $F(B_i^n) \subseteq \mathcal{IF}(B_i^n)$.

Next, the solver performs a domain-theoretic integration on the step-function enclosure $\{([ai/2^n, a(i + 1)/2^n], \mathcal{IF}(B_i^n)) \mid 0 \leq i < 2^n\}$, and works out an enclosure of this integral by 2^{n+1} rectangles $\{B_i^{n+1} \mid 0 \leq i < 2^{n+1}\}$, which forms the next approximation of the solution y , and then the solver moves on to iteration $n + 1$.

This means that the mere number of queries and answers grows exponentially in n , which puts a heavy burden on the communications between the machine and its oracle. To address this problem, we will consider a different protocol for communications in which queries from the solver machine \mathcal{S} to the field oracle \mathcal{F} are of the form $(n, \langle P, R \rangle)$, where $n \in \mathbb{N}$, and $\langle P, R \rangle$ is a polynomial enclosure in $\mathbb{PE}[0, a]$, and answers from \mathcal{F} to \mathcal{S} come in the form of polynomial enclosures in $\mathbb{PE}[0, a]$, so do the approximations to the solution y as sent out to the user by \mathcal{S} .

A denotational semantic model of communication protocols (such as ours) over real numbers has been studied in a broader context in [19,18]. Here we focus on the complexity theoretic implications of using one specific protocol.

³ See e. g. [22] page 79].

⁴ Generalising our results to the m -dimensional case and $y : [\alpha, \beta] \rightarrow [-b, b]^m$ will be straightforward.

3.1 Semantic Behaviour of the Field Oracle \mathcal{F}

It is instructive to begin by viewing the input to \mathcal{F} as a pair consisting of a number and a *function* enclosure. Assume that \mathcal{F} receives a query $(n, [\alpha, \beta])$ on its input tape, where $n \in \mathbb{N}$ and $[\alpha, \beta] \in \mathbb{PE}[0, a]$. We expect the respective output to be an approximation of the image of $[\alpha, \beta]$ under the field F , to within 2^{-n} accuracy. So, consider the set $\Phi := \{(t, s) \in [0, a] \times \mathbb{R} \mid \exists z \in \mathbb{R} : \alpha(t) \leq z \leq \beta(t) \wedge s = F(t, z)\}$. It is easy to see that there are two functions $f, g \in C[0, a]$ for which we have $\Phi = \Gamma([f, g])$, i. e. the graph of $[f, g]$ (see Definition [1](#)). We denote the enclosure $[f, g]$ as $F([\alpha, \beta])$, i. e.

$$F([\alpha, \beta]) := [f, g] \tag{4}$$

Next, we consider two polynomials ϕ and ψ that approximate f and g from below and above to within 2^{-n} accuracy, respectively, i. e. $[\phi, \psi] \sqsubseteq [f, g] \sqsubseteq [\phi + 2^{-n}, \psi - 2^{-n}]$. Remember that according to our protocols, the input to \mathcal{F} is a pair $(n, \langle P, R \rangle)$, in which $\langle P, R \rangle \in \mathbb{PE}[0, a]$. Hence we define:

Definition 3 (Lipschitz field oracle). *The field oracle \mathcal{F} is said to be Lipschitz with constant L if for each $n \in \mathbb{N}$ and $\langle P, R \rangle \in \mathbb{PE}[0, a]$, the input query $(n, \langle P, R \rangle)$ is responded with some $\langle \tilde{P}, \tilde{R} \rangle \in \mathbb{PE}[0, a]$, in which $\langle \tilde{P}, \tilde{R} \rangle$ is an enclosure of $F(\langle P, R \rangle)$ (see [4](#) above) and $\forall t \in [0, a] : 0 \leq \tilde{R}(t) \leq 2^{-n} + LR(t)$.*

Even though we abstract away from what goes on inside the oracle \mathcal{F} , yet to demonstrate that Definition [3](#) is a plausible one, we have presented a way of obtaining Lipschitz oracles from analytic fields using polynomial approximations in Appendix [A](#).

4 Convergence of Picard’s Method

Let us reformulate Picard’s method in our setting. The idea is for the solver machine to work out a sequence $\{\langle y_n, r_n \rangle \mid n \in \mathbb{N}\}$ of polynomial enclosures in such a way that they all enclose the final solution y , i. e. $\forall n \in \mathbb{N} : y \in \langle y_n, r_n \rangle$:

base case: $y_0(x) = 0, r_0(x) = b$.

recursive step: Assume that the solver \mathcal{S} has worked out the enclosure $\langle y_{n-1}, r_{n-1} \rangle$.

At this point \mathcal{S} sends the query $(n, \langle y_{n-1}, r_{n-1} \rangle)$ to the field oracle \mathcal{F} , and in return it receives the polynomial enclosure $\langle \tilde{P}_n, \tilde{R}_n \rangle$. To obtain the next approximation $\langle y_n, r_n \rangle$, all \mathcal{S} needs to do is symbolic integration over polynomials, i. e.

$$y_n(t) = \int_0^t \tilde{P}_n(x) dx, \quad r_n(t) = \int_0^t \tilde{R}_n(x) dx \tag{5}$$

Theorem 1 (geometric convergence). *Consider the IVP [3](#) and the configuration of Fig. [1](#). Moreover, assume that the field oracle \mathcal{F} is Lipschitz. Then the successive polynomial enclosures $\langle y_n, r_n \rangle$ sent on the output tape by the solver machine converge to the solution y of the IVP geometrically, that is, for some constant $c > 1$, the width of $\langle y_n, r_n \rangle$ decreases in $O(c^{-n})$.*

Proof. The main idea of the proof is similar to the usual one for the classical Picard-Lindelöf theorem. A detailed proof is given in Appendix [B](#). □

5 Reducing the Complexity to Polynomial-Time

One of the many ways with which the oracle \mathcal{F} can work out a polynomial enclosure in response to any query from the solver machine \mathcal{S} is via polynomial compositions (see Appendix A for a detailed account). Let $\langle \tilde{P}_n, \tilde{R}_n \rangle$ be the answer that \mathcal{F} sends to \mathcal{S} in response to the query $(n, \langle y_{n-1}, r_{n-1} \rangle)$ where $\tilde{P}_n(X) = P_n(X, y_{n-1}(X))$ and $P_n(X_1, X_2) \in \mathbb{Q}[X_1, X_2]$ is a polynomial approximation of the field F to within 2^{-n} accuracy.

Let us write q_n for the maximum degree of X_2 in $P_n(X_1, X_2)$, and d_n for the degree of the n^{th} polynomial approximation y_n of the solution y generated by the solver machine. It is straightforward to verify that the degrees of the polynomials $\{y_n \mid n \in \mathbb{N}\}$ satisfy $d_{n+1} \geq 1 + q_{n+1}d_n$ for all $n \in \mathbb{N}$, and $d_0 = 0$.

This should give us an idea of how the degrees of the approximants to the solution grow. For instance, even if $\forall n \in \mathbb{N} : q_n = 2$, then d_n grows at least exponentially. One way to address this inefficiency is to first approximate each of the polynomials \tilde{P}_n and \tilde{R}_n by lower degree polynomials \tilde{y}_n and \tilde{r}_n , respectively, and then proceed to work out the next approximation to the solution $\langle y_n, r_n \rangle$ accordingly.

There are various ways of approximating one polynomial with another of a lower degree. The approach which we consider here is based on using *truncated Chebyshev series*. The main reason for choosing this method over (say) the straightforward truncated Taylor series is that in numerical work, the Chebyshev basis is a much more well-behaved one compared with the monomial basis, a well-known fact spelled out in any standard text on function approximation [24][20].

Another reason is that for any rational polynomial, the calculation of Chebyshev truncation involves only arithmetic over rational numbers, in contrast to methods such as the Carathéodory-Fejér [14] which involves computations of eigenvalues that are not necessarily rational.⁵

Assume that a polynomial p is written as $\sum_{i=0}^n a_i x^i$ in the monomial basis, and as $\sum_{i=0}^n b_i T_i(x)$ in the Chebyshev basis, in which $T_i(x) = \sum_{j=0}^i t_{ij} x^j$ is the i -th Chebyshev polynomial of the first kind ($i \geq 0$).⁶ Going from the latter to the former is straightforward, and in the other direction, the coefficients b_i can be calculated as the entries of the vector B , which is the solution of the system of linear equations $S \times B = A$, in which $A = [a_0 \dots a_n]^T$, $B = [b_0 \dots b_n]^T$ and $S = [s_{i,j}]_{0 \leq i, j \leq n}$ is upper triangular with entries as follows:

$$s_{i,j} = \begin{cases} t_{ji} & \text{if } i \leq j \\ 0 & \text{if } i > j \end{cases} \tag{6}$$

For each $k \in \mathbb{N}$, the *Chebyshev-truncation* of p up to degree k is defined to be the polynomial $\text{ct}(p, k) := \sum_{i=0}^{k'} b_i T_i(x)$ in which $k' = \min\{k, n\}$. As $\forall i \in \mathbb{N}, x \in [-1, 1] : |T_i(x)| \leq 1$ and as we will focus only on the values of x in $[0, 1]$, then we define the *Chebyshev-bound* of this truncation to be $\text{cb}(p, k) := \sum_{i=k+1}^n |b_i|$.

For $\rho \geq 0$, let $C_\rho = \{\rho e^{i\theta} \mid 0 \leq \theta < 2\pi\}$ be the circle of radius ρ in the complex plain. We write E_ρ to denote the open region bounded by the ellipse which is obtained as the

⁵ Nonetheless, we will be considering the Carathéodory-Fejér method in our future work due to its remarkable performance in practice.

⁶ One can find a comprehensive treatment of Chebyshev polynomials in any standard book on approximation theory, e.g. [24][21].

image of the circle C_ρ under the Joukowski map $z \mapsto (z + z^{-1})/2$. The following is a corollary of Bernstein’s theorem [5]:

Theorem 2. *Assume that $p \in \mathbb{R}[x]$ is a real polynomial and choose $\rho > 1$ and $C \geq 0$ in such a way that $\forall z \in E_\rho : |p(z)| \leq C$. Then the Chebyshev coefficients of p satisfy $b_k \leq 2C\rho^{-k}$. As a consequence $\forall n \in \mathbb{N} : \text{cb}(p, n) \leq 2C\rho^{-n}/(\rho - 1)$.*

Now we focus on the solution of the equation (3) for values of $t \in [0, 1]$ and modify the algorithm of Section 4 to reduce the growth of the size of the polynomials y_n and r_n , i. e. both the degree and the bit size of coefficients. More precisely, after receiving the polynomial enclosure $\langle \tilde{P}_n, \tilde{R}_n \rangle$ from the field oracle, the solver first works out the Chebyshev truncation of \tilde{R}_n up to degree n to obtain the polynomial $\hat{r}_n := \text{ct}(\tilde{R}_n, n)$, and adds the bound $e(\hat{r}_n) := \text{cb}(\tilde{R}_n, n)$ to \hat{r}_n . Next it considers the Chebyshev truncation of \tilde{P}_n up to degree n , which we write as $\hat{y}_n := \text{ct}(\tilde{P}_n, n)$, and adds the bound $e(\hat{y}_n) := \text{cb}(\tilde{P}_n, n)$ to $\hat{r}_n + e(\hat{r}_n)$. At this point our polynomials satisfy $\text{deg}(\hat{y}_n) \leq n$ and $\text{deg}(\hat{r}_n) \leq n$.

The final concern is that the sizes of the representations of the coefficients of \hat{y}_n and \hat{r}_n may have grown too large. So, assume that $\hat{y}_n(x) = \sum_{i=0}^n a_i x^i$ in monomial basis and for each $i \leq n$, let $p_i/2^n$ be the largest dyadic number with denominator 2^n which is not greater than $|a_i|$, and define \hat{a}_i as $p_i/2^n$ if $a_i \geq 0$, and $-p_i/2^n$ otherwise. We define $\tilde{y}_n(x) := \sum_{i=0}^n \hat{a}_i x^i$. It should be clear that, as $\forall i \leq n : |a_i - \hat{a}_i| \leq 2^{-n}$, we will have $\|\tilde{y}_n - \hat{y}_n\| \leq (n + 1)2^{-n}$. Using the same method, we obtain \tilde{r}_n from \hat{r}_n by approximating its coefficients with appropriate dyadic numbers, with an added error of $(n + 1)2^{-n}$. So, the reduction of the sizes of the coefficients have added another $2(n + 1)2^{-n} = (n + 1)2^{-(n-1)}$ to our error estimates. Now we define:

$$y_n(t) = \int_0^t \tilde{y}_n(x) dx, \quad r_n(t) = \int_0^t (\tilde{r}_n(x) + e(\hat{r}_n) + e(\hat{y}_n) + (n + 1)2^{-(n-1)}) dx \quad (7)$$

In order to be able to use Theorem 2 we add the following assumption which states that in the modified algorithm:

$$\exists C \in \mathbb{N} : \forall n \in \mathbb{N} : \forall z \in E_5 : |\tilde{P}_n(z)| < C \wedge |\tilde{R}_n(z)| < C \quad (8)$$

We have chosen the elliptic region E_5 as it fully contains the circle C_2 . This ensures that the sizes of the coefficients \hat{a}_i ($i \geq 0$) remain within the desired bounds. To see this, assume that the complex function f is analytic in a neighbourhood D of C_ρ . Then inside D , the function f is equal to its Taylor series about 0, i. e. $f(z) = \sum_{i=0}^\infty a_i z^i$. In particular, we have:

$$\forall i \in \mathbb{N} : a_i = \frac{1}{2\pi i} \int_{C_\rho} \frac{f(z)}{z^{i+1}} dz \quad (9)$$

Proposition 1. *Assume that $\rho > 1$ and $\exists C > 0 : \forall z \in C_\rho : |f(z)| < C$. Then $\forall k \in \mathbb{N} : |a_k| < C/\rho^k$.*

Proof. Straightforward from (9). □

Under assumption (8) and by using Proposition 1 (and the fact that C_2 is contained in E_5), the coefficients of the polynomial $\hat{y}_n(x) = \sum_{i=0}^n a_i x^i$ satisfy $\forall i : |a_i| \leq C$, which implies that each \hat{a}_i will be representable in $(n \lceil \log(C) \rceil + 1)$ bits, i. e. $O(n)$ bits. A similar

argument holds for the coefficients of \hat{r}_n and \tilde{r}_n . As such we have reduced the degrees of our polynomials to $O(n)$, and by dyadic approximations, we have reduced the sizes of their representations to $O(n^2)$. Nonetheless, how much \mathcal{F} blows up the sizes of its answers is not under our control. Therefore, we consider the field oracles for which, the size of the answer to a query is bounded by a polynomial function of the size of the query:

Definition 4 (p-representable field oracle). *Assume that the size of the representation of each finite object x is written as $s(x)$. We say that the field oracle \mathcal{F} is polynomial-space representable, p-representable for short, if there exists a polynomial $q \in \mathbb{N}[X]$ such that for each input query $(n, \langle P, R \rangle)$, the answer polynomial enclosure $\langle \tilde{P}, \tilde{R} \rangle$ satisfies $s(\langle \tilde{P}, \tilde{R} \rangle) \leq q(m)$, where $m = \max\{n, s(\langle P, R \rangle)\}$.⁷*

Remark 3. It is worth mentioning that according to Jackson-Bernstein Theorem [17, page 254], by assuming the field oracle to be p-representable, we are effectively excluding the fields that are not infinitely differentiable. Yet by another theorem of Bernstein [21, Theorem 5.2.1, page 148], we are effectively including all analytic fields.

We can prove that for all values of $\tau \leq \min(a, 1)$, the enclosures $\langle y_n, r_n \rangle$ do indeed enclose the solution y over $[0, \tau]$, and in fact:

Theorem 3 (polynomial-time complexity). *For any Lipschitz p-representable field oracle \mathcal{F} and under assumption (8), the modified algorithm solves the associated IVP in polynomial-time over $[0, \tau]$, in which $\tau \leq \min(a, 1)$.*

Proof (Sketch). The fact that our modified algorithm converges follows from (7) and (8) and Theorem 2.

To prove polynomial-time complexity, first note that the volume of data on the channels is capped as a result of the truncations on the one-hand, and the assumption that the field is p-representable on the other hand. Regarding the cost of computations inside the solver, symbolic integration in monomial basis can be done in polynomial-time, and the rewriting of polynomials in Chebyshev basis is also polynomial-time as the matrix S in (6) is upper triangular with non-zero diagonal entries. For the details of this proof, please see Appendix C. \square

5.1 Analysis of Our Assumption

Admittedly, assumption (8) lacks the elegance that one would hope for. More importantly, it is not clear how restrictive this assumption is.

Note that for equation (3), Picard's method guarantees a lifetime of $r \leq \min(a, b/M)$ for the solution, in which M is the maximum norm of F over $[0, a] \times [-b, b]$. In Subsection 3.1, we required the field to return enclosures $\langle \tilde{P}, \tilde{R} \rangle$ that were 'tight enough' in response to queries $(n, \langle P, R \rangle)$. We considered the field and the polynomials as defined over real numbers. If we strengthen this requirement by interpreting the field and the polynomials involved over a certain subset of *complex* numbers, which in our case

⁷ For a more detailed account of this definition, including the precise account of the representation of a polynomial, see Appendix C.

should contain E_5 , we can guarantee polynomial-time complexity for our algorithm assuming $5 \leq \min(a, b/(2M))$ ⁸

But this seems to leave out some very simple equations such as $y'(t) = 2y(t) + 2$. The problem is that the lifetime as guaranteed by Picard’s method can be a gross underestimate of the real one. For instance, for the IVP $y'(t) = 2y(t) + 2$ with the initial condition $y(0) = 0$, the unique solution $y(t) = \exp(2t) - 1$ is defined over *the whole real line*, and it can be easily verified that our method converges in polynomial-time over this IVP⁹. Yet Picard’s method only guarantees a lifetime of $r \leq 1/2$. Therefore, we will need to analyse the strength of assumption (8) using a different method.

Picard’s method may be seen as a competition between the field oracle on the one hand, which may be highly expanding and thus may amplify tiny fluctuations, and the solver on the other hand, which smooths the results over by integration. Bournez et al. [7] have proven polynomial-time complexity of computing the Taylor coefficients of the solution under a cap on the growth of the field, which is called *poly-boundedness*. This condition is implied by p-representability of the field in our setting. Thus, in our future work, we will try to answer the following question:

Problem 1. Assume that the field F is extendable to a p-representable analytic function over a ‘suitably large’ compact subset of \mathbb{C}^2 , and the field oracle’s estimates are tight enough *over complex numbers*. Will it be true then that condition (8) is always satisfied and hence the IVP can be solved in polynomial-time using our modified method?

6 Summary

We have presented a denotational framework for analysing the semantics of computations over real functions based on polynomial enclosures. We focused on the case of Picard’s method of IVP solving. An oracle machine model was considered to separate the actual procedure of IVP solving from the field application. In Theorem 1, we showed that Picard’s method converges under an assumption which is equivalent to the classical Lipschitz condition on the field.

In general, IVPs cannot be solved in polynomial-time under the mere Lipschitz continuity condition on the field unless P=PSPACE [16]. On the other hand, from [23,7] we know that under the stronger analyticity condition on the field, the Taylor coefficients of the solutions can be approximated in polynomial-time. But this does not provide any explicit bounds on the solution.

In contrast, in this paper, we have presented a semantic model which provides explicit error bounds at each stage in the form of polynomial enclosures of the result. In that respect, our model is comparable to the usual domain-models, with the difference that ours makes polynomial-time solution of IVPs possible, as opposed to the step-function models considered in [10,12].

⁸ We do not present a formal proof of this claim here as it can be verified easily with a bit of calculation.

⁹ In fact, our implementation of the algorithm successfully converges over non-linear equations such as $y'(t) = \cos(t)y(t)$, $y(0) = 1$ and $y''(t) - 0.2(1 - y(t)^2)y'(t) + y(t) - 0.1 \cos(1.1t)$, $y(0) = 1$, the latter being an instance of the forced Van der Pol’s equation.

References

1. Aberth, O.: The failure in computable analysis of a classical existence theorem for differential equations. *Proceedings of the American Mathematical Society* 30(1), 151–156 (1971)
2. Abramsky, S., Jung, A.: Domain theory. In: Abramsky, S., Gabbay, D.M., Maibaum, T.S.E. (eds.) *Handbook of Logic in Computer Science*, vol. 3, pp. 1–168. Clarendon Press, Oxford (1994)
3. Achieser, N.I.: *Theory of Approximation*. Frederick Ungar Publishing Co. (1956)
4. Bernstein, S.N.: Sur les recherches récentes relatives à la meilleure approximation des fonctions continues par les polynômes. In: *Proc. of 5th Inter. Math. Congress*, vol. 1, pp. 256–266 (1912)
5. Bernstein, S.N.: Sur l'ordre de la meilleure approximation des fonctions continues par les polynômes de degré donné. *Mem. Cl. Sci. Acad. Roy. Belg.* 4, 1–103 (1912)
6. Bishop, E.: *Foundations of Constructive Analysis*. McGraw-Hill (1967)
7. Bournez, O., Graça, D.S., Pouly, A.: Solving Analytic Differential Equations in Polynomial Time over Unbounded Domains. In: Murlak, F., Sankowski, P. (eds.) *MFCS 2011. LNCS*, vol. 6907, pp. 170–181. Springer, Heidelberg (2011)
8. Edalat, A.: Dynamical systems, measures and fractals via domain theory. *Information and Computation* 120(1), 32–48 (1995)
9. Edalat, A., Lieutier, A.: Domain theory and differential calculus (functions of one variable). In: *Proceedings of 17th Annual IEEE Symposium on Logic in Computer Science (LICS 2002)*, Copenhagen, Denmark, pp. 277–286 (2002)
10. Edalat, A., Pattinson, D.: A domain-theoretic account of Picard's theorem. *LMS Journal of Computation and Mathematics* 10, 83–118 (2007)
11. Escardó, M.H.: PCF extended with real numbers: a domain theoretic approach to higher order exact real number computation. Ph.D. thesis, Imperial College (1997)
12. Farjudian, A., Konečný, M.: Time Complexity and Convergence Analysis of Domain Theoretic Picard Method. In: Hodges, W., de Queiroz, R. (eds.) *WoLLIC 2008. LNCS (LNAI)*, vol. 5110, pp. 149–163. Springer, Heidelberg (2008)
13. Gierz, G., Hofmann, K.H., Keimel, K., Lawson, J.D., Mislove, M.W., Scott, D.S.: *Continuous Lattices and Domains*. *Encycloedia of Mathematics and its Applications*, vol. 93. Cambridge University Press (2003)
14. Gutknecht, M.H., Trefethen, L.N.: Real polynomial Chebyshev approximation by the Carathéodory-Fejér method. *SIAM Journal on Numerical Analysis* 19(2), 358–371 (1982)
15. Jackson, D.: Über die Genauigkeit der Annäherung stetiger Funktionen durch ganze rationale Funktionen gegebenen Grades und trigonometrische Summen gegebener Ordnung. Ph.D. thesis, Göttingen (1911)
16. Kawamura, A.: Lipschitz Continuous Ordinary Differential Equations are Polynomial-Space Complete. In: *CCC 2009: 24th Annual IEEE Conference on Computational Complexity*, pp. 149–160 (2009)
17. Ko, K.I.: *Complexity Theory of Real Functions*. Birkhäuser, Boston (1991)
18. Konečný, M., Farjudian, A.: Compositional semantics of dataflow networks with query-driven communication of exact values. *Journal of Universal Computer Science* 16(18), 2629–2656 (2010)
19. Konečný, M., Farjudian, A.: Semantics of query-driven communication of exact values. *Journal of Universal Computer Science* 16(18), 2597–2628 (2010)
20. Lorentz, G.G.: *Approximation of Functions*. AMS Chelsea Publishing (1986)
21. Mhaskar, H.N., Pai, D.V.: *Fundamentals of Approximation Theory*. Narosa (2007)
22. Miller, R., Michel, A.: *Ordinary Differential Equations*. Academic Press (1982)

23. Müller, N., Moiske, B.: Solving initial value problems in polynomial time. In: Proc. 22nd JAIIO - PANEL 1993, Part 2, pp. 283–293 (1993)
24. Rivlin, T.J.: Chebyshev Polynomials: from Approximation Theory to Algebra and Number Theory, 2nd edn. Wiley, New York (1990)
25. Weihrauch, K.: Computable Analysis, An Introduction. Springer (2000)
26. Yoshizawa, T., Hayashi, K.: On the uniqueness of solutions of a system of ordinary differential equations. Memoirs of the College of Science, University of Kyoto. Ser. A, Mathematics 26(1), 19–29 (1950)

A Obtaining Lipschitz Oracles from Analytic Fields

Assume that $F : [0, a] \times [-b, b] \rightarrow \mathbb{R}$ is the field of our IVP and assume that F is continuous and Lipschitz (in the classical sense) in its second argument with Lipschitz constant L' . As F is continuous, there exists a sequence $\{P_n \mid n \in \mathbb{N}\}$ of polynomials in $\mathbb{Q}[X_1, X_2]$ such that $\forall n \in \mathbb{N} : F \in \langle P_n, 2^{-n} \rangle$ and $F = \lim_{n \rightarrow \infty} P_n$. Furthermore, if F is analytic, then we can choose the sequence of polynomials in such a way that $D_y(F) = \lim_{n \rightarrow \infty} D_y(P_n)$, where $D_y(F)$ is the derivative of F in the direction of the unit vector $(0, 1)$. In fact, we can choose $\{P_n \mid n \in \mathbb{N}\}$ in such a way that

$$\forall n \in \mathbb{N} : \|D_y(P_n)\| < L' + 1 \tag{10}$$

If one defines $L := L' + 1$, then there is a direct (yet very inefficient) way for the oracle \mathcal{F} to generate its output as follows: on each input $(n, \langle P, R \rangle)$, the oracle \mathcal{F} simply substitutes in $P_n(X_1, X_2)$ the identity polynomial X for X_1 , and the polynomial $P(X)$ for X_2 , to obtain the univariate polynomial $\tilde{P}_n(X)$. Then it returns the polynomial enclosure $\langle \tilde{P}_n, 2^{-n} + LR \rangle$. It should be clear that by the assumption of (10) and the fact that each P_n approximates F to within 2^{-n} accuracy, we will have $\langle \tilde{P}_n, 2^{-n} + LR \rangle \subseteq F(\langle P, R \rangle)$.

B Proof of Theorem 1 Regarding Geometric Convergence

We break down the proof of Theorem 1 into the following set of propositions:

Proposition 2. Assume that the field oracle \mathcal{F} is Lipschitz with constant L and let $y : [0, a] \rightarrow [-b, b]$ be a solution of the IVP. Then each $\langle y_n, r_n \rangle$ encloses y .

Proof. We prove the proposition by induction on n . Obviously $y \in \langle y_0, r_0 \rangle$. Now assume that $n > 0$ and $y \in \langle y_{n-1}, r_{n-1} \rangle$. Let $[\phi_n, \psi_n] = F(\langle y_{n-1}, r_{n-1} \rangle)$ and assume that $\langle \tilde{P}_n, \tilde{R}_n \rangle$ is the answer sent by the field oracle \mathcal{F} in response to the query $(n, \langle y_{n-1}, r_{n-1} \rangle)$. As \mathcal{F} is assumed to be Lipschitz, $\langle \tilde{P}_n, \tilde{R}_n \rangle \subseteq [\phi_n, \psi_n]$. By monotonicity of the integral operator and using (5) on page 237 we have $\forall t \in [0, a]$:

$$y_n(t) - r_n(t) \leq \int_0^t \phi_n(x) dx \leq \int_0^t \psi_n(x) dx \leq y_n(t) + r_n(t) \tag{11}$$

On the other hand, $y \in \langle y_{n-1}, r_{n-1} \rangle$ implies $\lambda x.F(x, y(x)) \in [\phi_n, \psi_n]$, and as y is a solution of the IVP:

$$\lambda t.y(t) = \lambda t. \int_0^t F(x, y(x)) dx \in [\lambda t. \int_0^t \phi_n(x) dx, \lambda t. \int_0^t \psi_n(x) dx] \tag{12}$$

Combining (11) and (12) completes the proof. □

For each $n \in \mathbb{N}$, we define the n -th degree polynomial

$$e_n(t) = t \sum_{k=0}^{n-1} \epsilon_k t^k + \epsilon_n t^n \tag{13}$$

by assigning $\epsilon_n = L^n b/n!$ and

$$\forall k \in \{0, \dots, n-1\} : \epsilon_k = \frac{L^k}{2^{n-k}(k+1)!}$$

Proposition 3. $\forall n \in \mathbb{N}, t \in [0, a] : 0 \leq r_n(t) \leq e_n(t)$

Proof. Straightforward induction using (5) on page 237 and using the fact that, as \mathcal{F} is Lipschitz (Definition 3 on page 237), $\forall n \in \mathbb{N}_+, t \in [0, a] : 0 \leq \tilde{R}_n(t) \leq 2^{-n} + Lr_{n-1}(t)$. □

Notice that in (13), we have split the coefficients of t^n into $\epsilon_{n-1} + \epsilon_n$. For instance, according to the format in (13) for r_2 we get:

$$0 \leq r_2(t) \leq \frac{1}{2^2 \times 1!} t + \frac{L}{2^1 \times 2!} t^2 + \frac{L^2 b}{2!} t^2$$

Proposition 4. *The maximum value of e_n over $[0, a]$ converges to zero geometrically, i. e. $\exists C > 0, n_0 \in \mathbb{N} : \forall n \geq n_0 : \|e_n\| \leq C/2^n$.*

Proof. First note that all of the coefficients of e_n are non-negative, hence e_n is non-decreasing and as a result it attains its maximum value at $t = a$, at which point according to (13)

$$e_n(a) = a \sum_{k=0}^{n-1} \epsilon_k a^k + \epsilon_n a^n = a \sum_{k=0}^{n-1} \frac{L^k a^k}{2^{n-k}(k+1)!} + \frac{L^n a^n}{n!}$$

Note that $(L^n a^n/n!)$ tends to 0 as $n \rightarrow \infty$ at a rate even faster than a geometric one. Moreover:

$$\begin{aligned} a \sum_{k=0}^{n-1} \frac{L^k a^k}{2^{n-k}(k+1)!} &= a \sum_{k=0}^{n-1} \frac{(2La)^k}{2^n(k+1)!} = \frac{1}{2L2^n} \sum_{k=1}^n \frac{(2La)^k}{k!} \\ &\leq \frac{1}{2L2^n} \sum_{k=1}^{\infty} \frac{(2La)^k}{k!} \leq \frac{e^{2La} - 1}{L2^{n+1}} \end{aligned} \tag{14}$$

Hence $\lim_{n \rightarrow \infty} e_n(a) = 0$ and the rate of convergence is geometric. □

Corollary 1. *The widths of $\langle y_n, r_n \rangle$ converge to zero geometrically.*

Theorem 1 follows from Proposition 2 and Corollary 1.

C Proof of Theorem 3 Regarding Polynomial-Time Complexity

Let us first clarify what we mean by a p-representable field oracle. For this we need to present a method for representing univariate rational polynomials of the form $P(x) = \sum_{i=0}^n a_i x^i$. Such a polynomial can be fed into a Turing machine as the following sequence of symbols:

$$\#\#a_0\#a_1\#\dots\#a_n\#\# \tag{15}$$

where # is used as a delimiter. If the size of each finite object x is denoted as $s(x)$, then according to (15) the size of the polynomial P would be $s(P) = (n + 4) s(\#) + \sum_{i=0}^n s(a_i)$.

A polynomial enclosure $\langle P, R \rangle$ can also be represented using the representations of P and R and an appropriate delimiter. For instance $\$\$P\$R\$\$,$ in which case $s(\langle P, R \rangle) = s(P) + s(R) + 5 s(\$)$.

Definition 5 (p-representable). We say that the field oracle \mathcal{F} is polynomial-space representable, p-representable for short, if there exists a polynomial $q \in \mathbb{N}[X]$ such that for each input query $(n, \langle P, R \rangle)$, the answer polynomial enclosure $\langle \tilde{P}, \tilde{R} \rangle$ satisfies $s(\langle \tilde{P}, \tilde{R} \rangle) \leq q(m)$, where $m = \max\{n, s(\langle P, R \rangle)\}$.

Now, assume that the field oracle \mathcal{F} is p-representable. To prove polynomial complexity, we need to analyse the following contributing factors:

Communication of Data: As we are capping the sizes of each y_n and r_n at stage n to at most $O(n^2)$, then the volume of data communicated between the solver on the one hand, and the field oracle and the outside world on the other hand remains within the polynomial bound. It is in fact very easy to see that this volume at each stage n is $O(n^{2 \deg(q)+1})$, where q is the bounding polynomial as in Definition 5 above.

Chebyshev Truncations: Note that the polynomial q puts a bound on the sizes of the answers of the field oracle \mathcal{F} , i. e. not just on the degree of the answers, but also on the sizes of the coefficients of the answer polynomials. Now assume that at stage n , the matrix S_n (as in (6) on page 238) is used to convert the representation of \tilde{P}_n from monomial to Chebyshev basis, and assume that S_n has got dimension $v_n \times v_n$. Then obviously we have $v_n \in O(n^{2 \deg(q)})$, and for each entry $s_{i,j}$ of S_n , we have $s(s_{i,j}) \in O(n^{2 \deg(q)})$. As S_n is upper triangular with all its diagonal entries non-zero, then at stage n the equation $S_n \times B = A$ can be solved with at most $O(v_n^2)$ arithmetic operations. The same argument holds for the Chebyshev truncation of \tilde{R}_n .

Dyadic Approximations: In Section 5 we discussed the approximation of each of the coefficients a_i (for $0 \leq i \leq n$) of \hat{y}_n by \hat{a}_i to obtain \tilde{y}_n . Let $i \leq n$ and for simplicity assume that $a_i \geq 0$, and let (the rational number) a_i be written as m_i/n_i . We want to find the biggest $p_i \in \mathbb{N}$ which is not bigger than $2^n m_i/n_i$. The easiest way to do this is by a simple integer division. As the sizes of the numerator and the denominator of $2^n m_i/n_i$ are at most $O(n^{2 \deg(q)+1})$, then the division can be carried out in at most $O(n^{4 \deg(q)+2})$ steps. This gives us p_i , and as a consequence, the required dyadic number $p_i/2^n$.

Therefore, as there are $n + 1$ coefficients, the approximation of \hat{y}_n by \tilde{y}_n can be carried out in $O(n^{4 \deg(q)+3})$. A similar argument holds for the approximation of \hat{r}_n by \tilde{r}_n .

This completes the proof of Theorem 3 □

Algorithmic Randomness and Ramsey Properties of Countable Homogeneous Structures^{*}

Willem L. Fouché

Department of Decision Sciences,
School of Economical Sciences,
University of South Africa, Pretoria, South Africa
fouchwl@gmail.com

Abstract. We study, in the context of algorithmic randomness, the closed amenable subgroups of the symmetric group S_∞ of a countable set. In this paper we investigate a link between the symmetries associated with Ramsey Fraïssé order classes and algorithmic randomness.

Mathematics Subject Classification (2000): 03C20, 03C57, 03D80, 05D10, 06D07, 20B27, 68Q30.

Keywords: Martin-Löf randomness, topological dynamics, amenable groups, Fraïssé limits, Ramsey theory.

1 Introduction

The focus of this work is, as for example in [4], [5], on the problem of understanding the symmetries that transform a recursively presented universal structure, which in this paper will be a Fraïssé limit of finite first order structures, to a copy of such a structure which is Martin-Löf random relative to a canonical S_∞ -invariant measure on the class of all universal structures of the given type. Here S_∞ is the symmetric group of a countable set, with the pointwise convergence topology. This investigation leads to a link between the symmetries associated with the so-called discernable flows in structural Ramsey theory and algorithmic randomness.

Glasner and Weiss [6] showed that there exists a unique measure on the set of linear orderings of the natural numbers (seen as a subset of Cantor space) that is invariant under the canonical action of the symmetric group of the natural numbers. The author [5] showed that this measure is computable and studied the associated Martin-Löf (ML) random points, which, due to the uniqueness of the Glasner-Weiss measure, may be regarded as random linear orders. The author showed that any ML-random linear order has the order type of the rationals. Moreover, it was shown that recent work by Kechris and Sokic [12] implies that no random linear order can be the extension of the universal poset (the Fraïssé limit

^{*} The research is based upon work supported by the National Research Foundation (NRF) of South Africa. Any opinion, findings and conclusions or recommendations expressed in this material are those of the author and therefore the NRF does not accept any liability in regard thereto.

of finite posets) to a linear order. In [5] a study was made of so-called “randomizers”. These are permutations of the natural numbers that transform a computable (Cantor) rational linear order into a random one. It was also proven in [5] that any such randomizer cannot be an automorphism of the universal poset.

The aim of this paper is to generalise these results to a broader class of Fraïssé limits \mathbb{F}_0 of Ramsey classes, the automorphism groups $\text{Aut}(\mathbb{F}_0)$ not being amenable. Again, as in [5], this paper relies heavily on the groundbreaking paper [10] by Kechris, Pestov and Todorcevic. The arguments in this paper require some understanding of the subtle interplay between structural Ramsey theory and topological dynamics as is beautifully explicated in the paper [10]. This paper has been written in such a way that it should be accessible to a non-specialist in Ramsey theory.

2 Preliminaries on Amenable Groups

Let G be a topological group and X a compact Hausdorff space. A dynamical system (X, G) (or a G -flow on X) is given by a jointly continuous action of G on X . If (Y, G) is a second dynamical system, then a G -morphism $\pi : (X, G) \rightarrow (Y, G)$ is a continuous mapping $\pi : X \rightarrow Y$ which intertwines the G -actions, i.e., the diagram

$$\begin{array}{ccc}
 G \times X & \xrightarrow{\alpha} & X \\
 1 \times \pi \downarrow & & \downarrow \pi \\
 G \times Y & \xrightarrow{\beta} & Y
 \end{array} ,$$

commutes with α, β being the group actions.

An isomorphism is a bijective homomorphism. A subflow of (X, G) is a G -flow on a compact subspace Y of X with the action of G on X restricted to the action on Y . A G -flow is minimal if it has no proper subflows. Every dynamical system has a minimal subflow (Zorn).

The following fact, first proven by Ellis (1949) [2], is central to the theory of dynamical systems:

Theorem 1. *Let G be a Hausdorff topological group. There exists, up to G -isomorphism, a unique minimal dynamical system, denoted by $(M(G), G)$, such that for every minimal dynamical system (X, G) there exists a G -epimorphism*

$$\pi : (M(G), G) \longrightarrow (X, G),$$

and any two such universal systems are isomorphic.

The flow $(M(G), G)$ is called the *universal minimal flow* of G .

We next introduce the notion of amenable groups.

Definition 1. *A topological group G is amenable if, whenever X is a non-empty compact Hausdorff space and π is a continuous action of G on X , then there is a G -invariant Borel probability measure on X .*

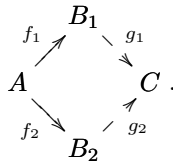
This means that, for every G -flow on a compact space X , there is a measure ν on the Borel algebra of X , such that, $\nu(X) = 1$ and, for every $g \in G$ and Borel subset U of X ,

$$\nu(gU) = \nu(U).$$

3 Fraïssé Limits and Their Recursive Representations

In the sequel, \mathcal{L} will stand for the signature of a relational structure. Moreover, \mathcal{L} will always be finite and the arities of the relational symbols will all be ≥ 1 . The definitions that follow were introduced by Fraïssé in 1954.

The *age* of an \mathcal{L} -structure X , written $\mathbf{Age}(X)$, is the class of all finite \mathcal{L} -structures (defined on finite ordinals) which can be embedded as \mathcal{L} -structures into X . The structure X is *homogeneous* (some authors say ultrahomogeneous) if, given any isomorphism $f : A \rightarrow B$ between finite substructures of X , there is an automorphism g of X whose restriction to A is f . A class \mathbf{K} of finite \mathcal{L} -structures has the *amalgamation property* if, for structures A, B_1, B_2 in \mathbf{K} and embeddings $f_i : A \rightarrow B_i$ ($i = 1, 2$) there is a structure C in \mathbf{K} and there are embeddings $g_i : B_i \rightarrow C$ ($i = 1, 2$), such that the following diagram commutes:



Suppose \mathbf{K} is a countable class of finite \mathcal{L} -structures, the domains of which are finite ordinals such that

1. if A is a finite \mathcal{L} -structure defined on some finite ordinal, if $B \in \mathbf{K}$ and if there is an embedding of A into B , then $A \in \mathbf{K}$;
2. the class \mathbf{K} has the amalgamation property.

Then, Fraïssé showed that there is a countable homogeneous structure X such that $\mathbf{Age}(X) = \mathbf{K}$. Moreover, X is unique up to isomorphism. The (essentially) unique X is called the *Fraïssé limit* \mathbb{K} of \mathbf{K} . Note that, conversely, the age \mathbf{K} of a countable homogeneous structure has properties (1) and (2). We shall frequently call a countable structure which is isomorphic to a Fraïssé limit a *universal structure*.

A recursive representation of a countably infinite \mathcal{L} -structure X is a bijection $\phi : X \rightarrow \mathbb{N}$ such that, for each $R \in \mathcal{L}$, if the arity of R is n , then the relation R^ϕ defined on \mathbb{N}^n by

$$R^\phi(x_1, x_2, \dots, x_n) \iff R(\phi^{-1}(x_1), \dots, \phi^{-1}(x_n)),$$

is recursive. If we identify the underlying set of X with \mathbb{N} via ϕ and each R with R^ϕ , we call the resulting structure a recursive \mathcal{L} -structure on \mathbb{N} and we say it has a recursive representation on \mathbb{N} .

If X is countable and homogeneous and if $\text{Age}(X)$ has an enumeration A_0, A_1, A_2, \dots , possibly with repetition, with the property that there is a recursive procedure that yields, for each $i \in \mathbb{N}$, and $R \in \mathcal{L}$, the underlying set $A(i)$ of A_i together with the interpretation of R in $A(i)$, then we call $(A_i : i \in \mathbb{N})$ a recursive enumeration of $\text{Age}(X)$. It follows from the construction of Fraïssé limits from their ages, that one can construct a recursive representation of X from a recursive enumeration of its age. (Conversely, it is trivial to derive a recursive enumeration of $\text{Age}(X)$ from a recursive representation of X .) It is therefore not difficult to find recursive representations for Fraïssé limits of classes \mathbf{K} from recursive enumerations of their ages.

Theorem 2. *Suppose \mathbb{C} and \mathbb{D} are countable and recursively represented \mathcal{L} -structures on \mathbb{N} with the same age. Suppose that they are both homogeneous. Then there is a recursive isomorphism from \mathbb{C} to \mathbb{D} .*

Proof. As mentioned in [5], the model-theoretic back-and-forth argument as discussed, for example, on pp 161-162 of Hodges [7] is constructive relative to the recursive representations of the homogeneous structures \mathbb{C} and \mathbb{D} .

4 Structural Ramsey Theory in a Model Theoretic Context

In this section we summarise the results from [10] which underly the formalisation and proof of the main theorem of this paper. Unless otherwise stated all the proofs of the statements made here can be found in [10].

Let \mathbf{K} be the age of some countable \mathcal{L} -structure. For $A, \pi \in \mathbf{K}$ we denote by A^π the set of all the (model-theoretic) structure-preserving embeddings of π in A . For a natural number $r \geq 1$ and for $\pi, A, B \in \mathbf{K}$ we introduce the predicate $B \rightsquigarrow (A)_r^\pi$ (Erdős-notation) to mean:

$$B \rightsquigarrow (A)_r^\pi \iff (\forall B^\pi \xrightarrow{\chi} B \exists A \xrightarrow{\alpha} B \begin{array}{ccc} A^\pi & \xrightarrow{\alpha_*} & B^\pi \\ & \searrow \oplus & \swarrow \chi \\ & r & \end{array}).$$

Here $\alpha_* : A^\pi \rightarrow B^\pi$ is the mapping that takes an embedding $\pi \xrightarrow{x} A$ to the induced embedding $\pi \xrightarrow{\alpha x} B$.

In other words, $B \rightsquigarrow (A)_r^\pi$ iff: for every r -colouring χ of the set B^π consisting of the embeddings of π in B (copies of π in B), there is an embedding α of A

into B such that χ_{α_*} is a constant. This means that χ assumes a constant value on all the embeddings of π into the image $A' \subset B$ of A under α .

We shall call an age \mathbf{K} a *Ramsey age* if, for all $\pi, A \in \mathbf{K}$ with $A^\pi \neq \emptyset$, and all natural numbers $r \geq 1$, there is some $B \in \mathbf{K}$ such that $B \rightsquigarrow (A)^\pi_r$.

Assume \mathcal{L} is a countable signature containing a distinguished binary relation symbol $<$. An *order structure* A for the signature \mathcal{L} with the distinguished symbol $<$, is a structure A for which the interpretation $<^A$ of the symbol $<$ in A is a total ordering.

An *order class* \mathbf{K} for \mathcal{L} is one for which all $A \in \mathbf{K}$ are order structures (relative to the distinguished $<$).

Let \mathcal{L}_0 be the signature obtained by removing the distinguished symbol $<$ from \mathcal{L} . For any \mathcal{L} -structure A , denote by A_0 the \mathcal{L}_0 -structure which is the reduct of A to \mathcal{L}_0 . This means that A_0 is the structure A where the distinguished order $<$ interpreted as a total order $<^A$ in A is being ignored.

Let \mathbf{K} be a Fraïssé order class. By this we mean an order class having a Fraïssé limit. Denote by \mathbf{K}_0 the class of all structures A_0 which are reducts of some $A \in \mathbf{K}$. We write \mathbb{F} for the Fraïssé limit of \mathbf{K} . We now discuss when \mathbf{K}_0 is also a Fraïssé class with limit \mathbb{F}_0 . Following Kechris, Pestov and Todorcevic [10], we say that the class \mathbf{K} is *reasonable* if for every $A_0, B_0 \in \mathbf{K}_0$, and linear ordering $<$ on A_0 such that $A := (A_0, <) \in \mathbf{K}$, and for an embedding $\pi : A_0 \rightarrow B_0$, there is a linear ordering $<_1$ on B_0 so that $B := (B_0, <_1) \in \mathbf{K}$ and $\pi : A \rightarrow B$ is also an embedding. (This means that $x < y \Leftrightarrow \pi(x) <_1 \pi(y)$.)

Then Kechris et al. (p 135) showed that \mathbf{K}_0 is a Fraïssé class with limit \mathbb{F}_0 , iff the Fraïssé order class \mathbf{K} is reasonable.

Note that, in this case, the underlying sets of \mathbb{F} and \mathbb{F}_0 are the same. Moreover, we can write

$$\mathbb{F} = (\mathbb{F}_0, <_0),$$

for some linear ordering $<_0$ on the underlying set of \mathbb{F}_0 .

We consider the continuous action of the automorphism group $\text{Aut}(\mathbb{F}_0)$ on the (topological) space of all linear orderings on the set F_0 , which is the underlying set of the structure \mathbb{F}_0 . Write $X_{\mathbf{K}} \subset \{0, 1\}^{F_0 \times F_0}$ for the orbit topological closure of the action of $\text{Aut}(\mathbb{F}_0)$ on the linear ordering $<_0$, i.e.,

$$X_{\mathbf{K}} = \overline{\text{Aut}(\mathbb{F}_0). <_0}.$$

This set is clearly a closed, hence compact, subset of the Baire space $\{0, 1\}^{F_0 \times F_0}$. Moreover, it is clearly also an $\text{Aut}(\mathbb{F}_0)$ -invariant subset of $\{0, 1\}^{F_0 \times F_0}$ under the natural action of $\text{Aut}(\mathbb{F}_0)$ on the latter space. We have thus obtained an $\text{Aut}(\mathbb{F}_0)$ -flow on $X_{\mathbf{K}}$.

This flow can be defined for any reasonable (in the technical sense as explained above) Fraïssé order class \mathbf{K} . I will call it the *discerning* flow associated with the reasonable Fraïssé order class \mathbf{K} .

Remark. The author uses the terminology *discerning* in acknowledgement of Ramseys pioneering work in developing his theorem in the context what now would be considered as a study of indiscernibles in model theory.

If, in addition to being a Fraïssé order class, the class \mathbf{K} is a Ramsey age, then every minimal subflow of the discernable flow is isomorphic to the universal minimal flow of $\text{Aut}(\mathbb{F}_0)$.

The Fraïssé order class \mathbf{K} is said to have the *ordering property* if for every $A_0 \in \mathbf{K}_0$, there is a $B_0 \in \mathbf{K}_0$ such for any linear ordering $<$ on A_0 and every linear ordering $<_1$ on B_0 , where both $<, <_1$ are restrictions of $<_0$, there is an embedding of $(A_0, <)$ into $(B_0, <_1)$.

The discerning flow associated with the Fraïssé order class \mathbf{K} is itself minimal iff \mathbf{K} has the ordering property.

We also extract the following remark from [10].

Proposition 1. *If \mathbf{K} is a Fraïssé order class which is Ramsey and has the ordering property, then a total order ξ belongs to the discerning flow $X_{\mathbf{K}}$ iff for any L_0 -structure A in the age of \mathbb{F}_0 allowing an $<^A$ interpretation of $<$, it is the case that $<^A$ is the restriction of ξ to A .*

We shall make substantial use of this remark in the sequel.

Example. It is known (see, for example [3] that the class \mathbf{P} (finite posets, linear extensions) is Ramsey and has the ordering property. As was noted in [10], this has the implication that the discerning $\text{Aut}(\mathbb{P}_0)$ -flow is thus a universal minimal flow. It acts on the space $X_{\mathbf{P}}$ consisting of the linear extensions of the universal poset \mathbb{P}_0 . Using these facts, Kechris and Sokič (2011) [12] recently showed that the automorphism group of \mathbb{P}_0 is not amenable. These results do imply that the set of linear extensions of the Fraïssé limit of finite posets are all, in a definite sense, nonrandom, at least from the point of view of algorithmic randomness as was shown in [5]. This result will be placed in a broader context in what follows.

5 Martin-Löf Random Countable Orders

Let S_∞ be the group of permutations of a countable set, which, without loss of generality, we may take to be \mathbb{N} . We place on S_∞ the pointwise convergence topology. Let $(\mathbb{N} \times \mathbb{N})_{\neq}$ denote the set of ordered pairs (i, j) of natural numbers with $i \neq j$. Write \mathcal{M} for the set of total orders on \mathbb{N} . We identify \mathcal{M} with a subset of $\{0, 1\}^{(\mathbb{N} \times \mathbb{N})_{\neq}}$ by identifying a total order $<$ on \mathbb{N} with the function $\xi : (\mathbb{N} \times \mathbb{N})_{\neq} \rightarrow \{0, 1\}$ given by

$$\xi(x, y) = 1 \Leftrightarrow x < y, \quad x, y \in \mathbb{N}.$$

The total order associated with ξ will be denoted by $<_\xi$. We topologise \mathcal{M} via the natural injection

$$\mathcal{M} \longrightarrow \{0, 1\}^{(\mathbb{N} \times \mathbb{N})_{\neq}},$$

where the (Baire) space $\{0, 1\}^{(\mathbb{N} \times \mathbb{N})_{\neq}}$ has the product topology. As such \mathcal{M} is a closed hence compact subspace of $\{0, 1\}^{(\mathbb{N} \times \mathbb{N})_{\neq}}$.

The group S_∞ acts continuously on \mathcal{M} if, for $\xi \in \mathcal{M}$ and $\sigma \in S_\infty$, we define the total order $\sigma\xi$ by:

$$x <_{\sigma\xi} y \iff \sigma^{-1}x <_\xi \sigma^{-1}y, \quad x, y \in \mathbb{N}.$$

Since S_∞ is an amenable group, there is an S_∞ -invariant measure on \mathcal{M} . In fact, Glasner and Weiss (2002) [6] showed that there is *exactly one* such measure (i.e., the flow on \mathcal{M} is uniquely ergodic). Their proof is based on an ergodic argument. Let us denote this measure by μ . I shall refer to this measure as in [5] as the *Glasner-Weiss measure*.

We write \mathcal{M}_f for the set of finite total orders on some subset of \mathbb{N} . For $\ell \in \mathcal{M}_f$, denote by Z_ℓ the set of $\xi \in \mathcal{M}$, such that ξ is an extension of ℓ . These sets are the cylinder subsets of \mathcal{M} . Write \mathcal{Z}_0 for the class of events of the form Z_ℓ for some $\ell \in \mathcal{M}_f$ and \mathcal{Z} for the algebra generated by \mathcal{Z}_0 . Note that the σ -algebra generated by \mathcal{Z} is exactly the Borel algebra on \mathcal{M} .

For $Z \in \mathcal{Z}_0$ we write Z^0 for the complement of Z and Z^1 for Z . Let $(T_i)_{i \in \mathbb{N}}$ be any enumeration of the algebra \mathcal{Z} generated by $(Z_\ell)_{\ell \in \mathcal{M}_f}$ in such a way that one can effectively retrieve from a given $i \in \mathbb{N}$, the corresponding T_i as a finite union of sets T of the form

$$T = Z_{\ell_1}^{\delta_1} \cap \dots \cap Z_{\ell_k}^{\delta_k}, \tag{1}$$

where each ℓ_i is in \mathcal{M}_f and $\delta_i \in \{0,1\}$ for $i = 1, \dots, k$. We call any such enumeration a *recursive representation* of \mathcal{Z} .

The Glasner-Weiss measure μ is computable in the following sense:

Theorem 3. *Denote by μ the Glasner-Weiss measure on the Borel-algebra of \mathcal{M} . Let $(T_i : i < \omega)$ be a recursive representation of the algebra \mathcal{Z} . There is an effective procedure that yields, for $i, k \in \mathbb{N}$, a binary rational β_k such that*

$$|\mu(T_i) - \beta_k| < 2^{-k}.$$

A proof of this result appears in [5].

Definition 2. *A set $A \subset \mathcal{M}$ is of constructive measure 0, if, for some recursive representation of $(T_i : i \in \mathbb{N})$ of \mathcal{Z} , there is a total recursive $\phi : \mathbb{N}^2 \rightarrow \mathbb{N}$ such that*

$$A \subset \bigcap_n \bigcup_m T_{\phi(n,m)}$$

and $\mu(\bigcup_m T_{\phi(n,m)}) < 2^{-n}$.

Definition 3. *A total order ξ is said to be μ -Martin-Löf random if ξ is in the complement of every subset B of \mathcal{M} of constructive measure 0.*

6 The Main Theorem

Write $ML_\mu \subset \mathcal{M}$ for the set of μ -Martin-Löf random total orders.

Theorem 4. *Let \mathbf{K} be a recursive Fraïssé order class which is Ramsey and has the ordering property. Write*

$$\mathbb{F} = (\mathbb{F}_0, <)$$

for its Fraïssé limit and $X_{\mathbf{K}}$ for the associated discerning flow. Fix some recursive representation of \mathbb{F} . Note that

$$X_{\mathbf{K}} \subset \mathcal{M}.$$

If some element of $X_{\mathbf{K}}$ is μ -Martin-Löf random, then the automorphism group $\text{Aut}(\mathbb{F}_0)$ is amenable. Equivalently, if $\text{Aut}(\mathbb{F}_0)$ is not amenable, then

$$ML_{\mu} \cap X_{\mathbf{K}} = \emptyset.$$

Corollary 1. *Consequently, if for some $\xi \in X_{\mathbf{K}}$ and some automorphism π of \mathbb{F}_0 it is the case that the linear order $\pi\xi$ is μ -Martin-Löf random, then $\text{Aut}(\mathbb{F}_0)$ is amenable.*

Proof of Theorem: Note that a topological group G is amenable iff its universal minimal flow $M(G)$ has a G -invariant probability measure. Indeed, let ν be an invariant measure on $M(G)$. Consider any G -flow on some compact Hausdorff space X . By Zorn’s lemma there is a minimal subflow Y and a G -embedding i of Y into X . Therefore, there are G -morphisms

$$M(G) \xrightarrow{\pi} Y \xrightarrow{i} X .$$

Let ρ be the pushout measure of ν under $i\pi$. In other words, for every Borel subset A of X , we set

$$\rho(A) = \nu(\pi^{-1}i^{-1}A).$$

Then ρ is an invariant measure on X . The converse is trivial, since $M(G)$ is a compact G -flow.

We introduce a number of (standard) recursion-theoretic concepts and terminology: A sequence (A_n) of sets in \mathcal{Z} is said to be *enumerable* if for each n , the set A_n is of the form $T_{\phi(n)}$ for some total recursive function $\phi : \omega \rightarrow \omega$ and some effective enumeration (T_i) of \mathcal{Z} . (Note that the sequence (A_n^c) , where A_n^c is the complement of A_n , is also an \mathcal{Z} -enumerable sequence.) In this case, we call the union $\bigcup_n A_n$ a Σ_1^0 class. A set is a Π_1^0 set if it is the complement of a Σ_1^0 set. It is of the form $\bigcap_n A_n$, for some \mathcal{Z} -semirecursive sequence (A_n) .

We shall also need the following observation. (In the language of algorithmic randomness, it states the well-known fact that the notion of Martin-Löf randomness is stronger than that of Kurtz randomness. A proof of this observation, in the present context, can be found in [5]. For more on Kurtz randomness, the reader is referred to the book [1] by Downey and Hirschfeldt.)

Lemma 1. *If A is a Σ_1^0 subset of \mathcal{M} and if $\mu(A) = 1$, then ML_{μ} is contained in A . In particular, if B is a Π_1^0 subset of \mathcal{M} that contains some element of ML_{μ} , then $\mu(B) > 0$.*

It follows from Proposition [1] that a total order ξ belongs to $X_{\mathbf{K}}$ iff for any A in the age of \mathbb{F}_0 it is the case that ξ^A is the restriction of ξ to A . Therefore, since \mathbf{K} is a recursive order class, the relation

$$\xi \in X_{\mathbf{K}}$$

is Π_1^0 definable over \mathcal{M} . It follows from Lemma [11](#) that, if

$$ML_\mu \cap X_{\mathbf{K}} \neq \emptyset,$$

then $\mu(X_{\mathbf{K}}) > 0$. This means that μ is a *nonzero* $\text{Aut}(\mathbb{F}_0)$ -invariant measure on the flow $X_{\mathbf{K}}$.

Since \mathbf{K} is a Ramsey order class, $X_{\mathbf{K}}$ is the *universal* minimal flow associated with the group $\text{Aut}(\mathbb{F}_0)$. By universality we can conclude that any $\text{Aut}(\mathbb{F}_0)$ -flow on a compact Hausdorff space will admit a nonzero $\text{Aut}(\mathbb{F}_0)$ -invariant measure. In particular, $\text{Aut}(\mathbb{F}_0)$ is an amenable topological group.

The second part now follows from the observation that if $\xi \in X_{\mathbf{K}}$ and π is an automorphism of \mathbb{F}_0 then $\pi\xi$ will also belong to $X_{\mathbf{K}}$.

Corollary 2. *Fix a recursive representation of the universal poset \mathbb{P}_0 on the natural numbers \mathbb{N} . Let $\mathcal{M}(\mathbb{P}_0)$ be the class of linear extensions of \mathbb{P}_0 . Write ML_μ for the set of total orders on \mathbb{N} that are Martin-Löf random relative to the Glasner-Weiss probability measure μ . Then*

$$ML_\mu \cap \mathcal{M}(\mathbb{P}_0) = \emptyset.$$

Proof: The result is a direct consequence of the fact that $\text{Aut}(\mathbb{P}_0)$ is not an amenable group. [\[12\]](#).

7 Open Problems

A total order on \mathbb{N} is a *Cantor rational order* if it is isomorphic to the standard ordering of the rational numbers. The following theorem is in [\[5\]](#).

Theorem 5. *Write \mathcal{Q} for the set of total orders on \mathbb{N} which are isomorphic to the Cantor rational order η . Then*

$$ML_\mu \subset \mathcal{Q}.$$

In particular,

$$\mu(\mathcal{Q}) = 1.$$

This observation has the following consequence.

Theorem 6. *For a total order η , set*

$$S_\mu(\eta) := \{\sigma \in S_\infty : \sigma\eta \in ML_\mu\}.$$

Then $S_\mu(\eta) \neq \emptyset$ iff η is a rational Cantor order.

Proof. By Theorem [5](#), if η were not rational, the corresponding set $S_\mu(\eta)$ must be the empty set. If η is rational, then the class \mathcal{Q} is exactly the orbit of η under the action of S_∞ . Since both \mathcal{Q} and ML_μ have μ -measure one, it follows that

$$\mu(\mathcal{Q} \cap ML_\mu) = 1,$$

and, therefore, that $S_\mu(\eta) \neq \emptyset$.

Following [5], note that, if $\pi \in S_\infty$, then

$$S_\mu(\eta)\pi^{-1} = S_\mu(\pi\eta). \tag{2}$$

Indeed, for $\alpha \in S_\mu(\eta)$, we have $\alpha\pi^{-1}(\pi\eta) = \alpha\eta \in ML_\mu$ and hence $\alpha\pi^{-1} \in S_\mu(\pi\eta)$. Conversely, if $\tau \in S_\mu(\pi\eta)$, then $\tau\pi\eta \in ML_\mu$, i.e., $\tau\pi \in S_\mu(\eta)$, and, so, $\tau \in S_\mu(\eta)\pi^{-1}$.

If $\eta_1, \eta_2 \in \mathcal{Q}$, there is some $\pi \in S_\infty$ such that $\eta_2 = \pi\eta_1$. Moreover, if η_1, η_2 were both recursive, the permutation π could also be chosen to be recursive. (See Theorem 2). Write S_r for the class of recursive permutations of \mathbb{N} . We let S_r act on the right on the class Σ of all sets of the form $S_\mu(\tau)$ with τ a recursive rational order on \mathbb{N} . The action is given by

$$\Sigma \times S_r \longrightarrow \Sigma,$$

$$(S_\mu(\tau), \pi) \mapsto S_\mu(\tau)\pi^{-1}, \pi \in S_r, \tau \in \mathcal{Q}_r,$$

where \mathcal{Q}_r denotes the class of all recursive rational orders on \mathbb{N} . It follows from the preceding arguments that this S_r -action will have a single orbit, i.e, the action is transitive. Set

$$\mathcal{S} = \bigcup_{\tau \in \mathcal{Q}_r} S_\mu(\tau).$$

If we choose any fixed $\eta \in \mathcal{Q}_r$, we also have

$$\mathcal{S} = \bigcup_{\pi \in S_r} S_\mu(\eta)\pi^{-1}.$$

We shall call the permutations in \mathcal{S} *Martin-Löf randomizers*. These are the permutations that transform some recursive rational order to one which is μ -Martin-Löf random.

These arguments show that an understanding of \mathcal{S} can be attained from any single $S_\mu(\tau)$ for a single recursive rational order τ modulo the recursive permutations in S_∞ .

Let \mathbf{K} be a recursive Fraïssé order class which is Ramsey and has the ordering property. Write again

$$\mathbb{F} = (\mathbb{F}_0, <)$$

for its Fraïssé limit and $X_{\mathbf{K}}$ for the associated discerning flow. The arguments of this paper show that, if $\tau \in \mathcal{Q}_r \cap X_{\mathbf{K}}$, then the presence of elements in $\text{Aut}(\mathbb{F}_0)$ which are Martin-Löf randomizers of τ is related to the amenability of the group $\text{Aut}(\mathbb{F}_0)$. Indeed, for some $\pi \in \text{Aut}(\mathbb{F}_0)$ to be a Martin-Löf randomizer of any τ as above is a *generic* property, in the sense that this very fact forces the group $\text{Aut}(\mathbb{F}_0)$ to be amenable! The problem still remains to identify the class of Martin-Löf randomizers.

Let \mathbf{L} be the Fraïssé order class consisting of all pairs $(L, <)$ where L is a lattice with underlying set a finite ordinal and with $<$ being a total order on the underlying set of L which is a linear extension of the partial order on L . As far as the author knows, it is unknown whether \mathbf{L} is Ramsey and whether it has

the ordering property. The author has discussed this problem with specialists in Ramsey theory and it would appear that this problem is wide open. Writing $\mathbb{L} = (\mathbb{L}_0, <)$ for the Fraïssé limit of \mathbf{L} , it is also an interesting open problem to relate $X_{\mathbf{L}}$ to ML_{μ} and thus perhaps gaining an understanding of the amenability or not of $\text{Aut}(\mathbb{L}_0)$. Note that if \mathbf{L} were Ramsey with the ordering property, then $\text{Aut}(\mathbb{L})$ would be an extremely amenable group. This would mean that its universal minimal flow is a singleton.

References

1. Downey, R.G., Hirschfeldt, D.R.: *Algorithmic Randomness and Complexity, Theory and Applications of Computability*. Springer (2010)
2. Ellis, R.: Universal minimal sets. *Proc. Amer. Math. Soc.* 11, 272–281 (1949)
3. Fouché, W.L.: Symmetry and the Ramsey degree of posets. *Discrete Math.* 167/168, 309–315 (1997)
4. Fouché, W.L., Potgieter, P.H.: Kolmogorov complexity and symmetrical relational structures. *J. Symbolic Logic* 63, 1083–1094 (1998)
5. Fouché, W.L.: Martin-Löf randomness, invariant measures and countable homogeneous structures. *Theory of Computing Systems* (to appear), arXiv:1205:0386v1
6. Glasner, E., Weiss, B.: Minimal actions of the group $S(\mathbb{Z})$ of permutations of the integers. *Geometric and Functional Analysis* 5, 964–988 (2002)
7. Hodges, W.: *A shorter model theory*. Cambridge University Press (1993)
8. Hrushovski, E.: Extending partial isomorphisms of graphs. *Combinatorica* 12, 411–416 (1992)
9. Kechris, A.S.: The dynamics of automorphism groups of homogeneous structures. Lecture at LMS Northern Regional Meeting (July 2011)
10. Kechris, A.S., Pestov, V.G., Todorćević, S.: Fraïssé limits, Ramsey theory, and topological dynamics of automorphism groups. *Geometric And Functional Analysis* 15, 106–189 (2005)
11. Kechris, A.S., Rosendal, C.: Turbulence, amalgamation and generic automorphisms of homogeneous structures. *Proc. London Math. Soc.* 94(3), 302–350 (2007)
12. Kechris, A.S., Sokić, M.: Dynamical properties of the automorphism groups of the random poset and random distributive lattice (2011), <http://www.math.caltech.edu/people/kechris.html>
13. Petrov, F., Vershik, A.: Uncountable graphs and invariant means on the set of universal countable graphs. *Random Structures and Algorithms* 126, 389–405 (2010)

Propositional Reasoning about Saturated Conditional Probabilistic Independence

Sebastian Link

Department of Computer Science, University of Auckland, New Zealand

Abstract. Conditional independence provides an essential framework to deal with knowledge and uncertainty in Artificial Intelligence, and is fundamental in probability and multivariate statistics. Its associated implication problem is paramount for building Bayesian networks. Unfortunately, the problem does not enjoy a finite ground axiomatization and is already coNP-complete to decide for restricted subclasses. Saturated conditional independencies form an important subclass of conditional independencies whose implication problem is decidable in almost linear time. Geiger and Pearl have established a finite ground axiomatization for this class. We establish a new completeness proof for this axiomatization, utilizing a new sound inference rule. The proof introduces special probability models where two values have probability one half. Special probability models allow us to establish a semantic proof for the equivalence between the implication of saturated conditional independencies and formulae in a Boolean propositional fragment. The equivalence extends the duality between the propositional fragment and multivalued dependencies in relational databases to a trinity involving saturated conditional independencies.

1 Introduction

The concept of conditional independence is important for capturing structural aspects of probability distributions, for dealing with knowledge and uncertainty in Artificial Intelligence, and for learning and reasoning in intelligent systems [15]. Application areas include natural language processing, speech processing, computer vision, robotics, computational biology, and error-control coding [7]. A conditional independence (CI) statement $I(Y, Z \mid X)$ represents the independence of two sets of attributes relative to a third: given three mutually disjoint subsets X , Y , and Z of a set S of attributes, if we have knowledge about the state of X , then knowledge about the state of Y does not provide additional evidence for the state of Z and vice versa. An important problem is the implication problem, which is to decide for an arbitrary set S , and an arbitrary set $\Sigma \cup \{\varphi\}$ of CI statements over S , whether every probability model that satisfies every CI statement in Σ also satisfies φ . The significance of this problem is due to its relevance for building Bayesian networks [15]. The implication problem for CI statements is not finitely axiomatizable by a set of Horn rules [17]. However, it is possible to express CI statements using polynomial likelihood formulae,

and reasoning about polynomial inequalities is axiomatizable [7]. Recently, the implication problem of stable CI statements has been shown to be finitely axiomatizable [14], and *coNP*-complete to decide [13]. Here, stability means that the validity of $I(Y, Z \mid X)$ over S implies the validity of every $I(Y, Z \mid X')$ where $X \subseteq X' \subseteq S - YZ$. An important subclass of CI statements are saturated conditional independence (SCI) statements. These are CI statements $I(Y, Z \mid X)$ over S that satisfy $XYZ = S$. Geiger and Pearl have established a finite axiomatization for the implication problem [6].

Example 1. Let $\text{BLUERAY} = \{\text{M(ovie), A(ctor), R(ole), C(rew), F(eature), L(anguage)}\}$ denote a set of attributes that model information about blue-rays of movies, including which actors played which role, what crew members were involved, and which features are available in which language. Let Σ consist of $I(\text{ARC}, \text{FL} \mid \text{M})$ and $I(\text{AR}, \text{CFL} \mid \text{M})$, let φ_1 be $I(\text{C}, \text{ARFL} \mid \text{M})$, and let φ_2 be $I(\text{A}, \text{CRFL} \mid \text{M})$. Then Σ implies φ_1 , but Σ does not imply φ_2 . Indeed, we can define two values that match on every attribute in MCFL , differ on A and on R ; and assign the probability one half to both values. This probability model satisfies both elements of Σ but violates φ_2 . \square

The notion of saturated conditional independence $I(Z, Y \mid X)$ over S is very closely related to that of a multivalued dependency (MVD) $X \twoheadrightarrow Y \mid Z$ over S , studied in the framework of relational databases [2,1,5,9,11,12,16]. Here, a set X of attributes is used to denote the X -value of a tuple over S , i.e., those tuple components that appear in the columns associated with X . Indeed, $X \twoheadrightarrow Y \mid Z$ expresses the fact that an X -value uniquely determines the set of associated Y -values independently of joint associations with Z -values where $Z = S - XY$. Thus, given a specific occurrence of an X -value within a tuple, so far not knowing the specific association with a Y -value and Z -value within this tuple, and then learning about the specific associated Y -value does not provide any information about the specific associated Z -value. Previous research has established an equivalence between the implication problem for SCI statements and that for MVDs [18]. The equivalence proof between the implication of MVDs and that of SCI statements was only syntactic in the sense that they both enjoy a common axiomatization. In addition it is known that the implication problem of MVDs can be decided in almost linear time [5], and that it is equivalent to that of formulae in a Boolean propositional fragment \mathcal{F} [16], even in nested databases with finite list, and record constructors [10]. Indeed, Sagiv et al. showed that it suffices to consider two-tuple relations in order to decide the implication problem of MVDs. This enabled them to define truth assignments from two-tuple relations, and vice versa, in such a way that the two-tuple relation satisfies an MVD if and only if the truth assignment is a model for the \mathcal{F} -formula that corresponds to the MVD. It follows from these results that the implication of SCI statements is equivalent to that of \mathcal{F} -formulae.

Example 2. Let $\mathcal{L} = \{M', A', R', C', F', L'\}$ denote a set of propositional variables. Let Σ' consist of the formulae $M' \rightarrow ((A' \wedge R' \wedge C') \vee (F' \wedge L'))$ and $M' \rightarrow ((A' \wedge R') \vee (C' \wedge F' \wedge L'))$, let φ'_1 be the formulae $M' \rightarrow (C' \vee (A' \wedge R' \wedge F' \wedge L'))$,

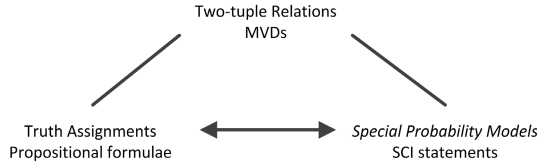


Fig. 1. Special Probability Models and their Correspondence to Truth Assignments

and let φ'_2 denote the formula $M' \rightarrow (A' \vee (C' \wedge R' \wedge F' \wedge L'))$. Indeed, Σ' logically implies φ'_1 , but Σ' does not logically imply φ'_2 . In fact, the truth assignment that assigns true to M', C', F', L' and false to A' and R' is a model for the formulae in Σ' but not a model for φ'_2 . \square

Contributions. The purpose of this paper is to establish a direct and explicit correspondence between probability models for SCI statements and models for \mathcal{F} -formulae. Based on a new sound inference rule for the implication of SCI statements we will develop a new completeness proof for Geiger and Pearls axiomatization of SCI statements. The proof introduces probability models that are special in the sense that two values have probability one half. The argument shows, in particular, that the implication problem of SCI statements over discrete probability models is the same as the implication problem of SCI statements over special binary probability models. Example 1 illustrates the definition of such a special probability model. Special probability models form the analogue of Sagiv et al.’s two-tuple relations. Indeed, we use special probability models to define truth assignments, and vice versa, in such a way that the special probability model satisfies an SCI statement φ if and only if the truth assignment is a model for the \mathcal{F} -formula φ' that corresponds to φ . Example 2 shows the definition of such a truth assignment. In fact, we simply assign true to variables that correspond to attributes with matching values in the special probability model. Figure 1 summarizes our contribution.

Organization. We give preliminary definitions in Section 2. The new completeness proof for Geiger and Pearl’s axiomatization is established in Section 3, introducing a new inference rule and special probability models. In Section 4 we establish the correspondence between special probability models and truth assignments. We conclude in Section 5.

2 Saturated Conditional Independence

We use the framework of Geiger and Pearl [6]. We denote by S a finite set of distinct symbols $\{s_1, \dots, s_n\}$, called *attributes*. A *domain mapping* is a mapping that associates a set, $dom(s_i)$, with each attribute s_i . This set is called the *domain* of s_i and each of its elements is a *value* for s_i . For $X = \{a_1, \dots, a_k\} \subseteq S$ we say that \mathbf{x} is a value of X , if $\mathbf{x} \in dom(a_1) \times \dots \times dom(a_k)$. For a value \mathbf{x} of X we write $\mathbf{x}(y)$ for the projection of \mathbf{x} onto $Y \subseteq X$.

Table 1. Axiomatization $\mathfrak{G} = \{\mathcal{T}, \mathcal{S}, \mathcal{C}, \mathcal{W}\}$ and Algebra Rule

$\frac{}{I(S - X, \emptyset X)}$ (saturated trivial independence, \mathcal{T})	$\frac{I(Y, Z X)}{I(Z, Y X)}$ (symmetry, \mathcal{S})
$\frac{I(ZW, Y X) \quad I(Z, W XY)}{I(Z, YW X)}$ (weak contraction, \mathcal{C})	$\frac{I(Y, ZW X)}{I(Y, Z XW)}$ (weak union, \mathcal{W})
$\frac{I(Y Y', Z Z' X) \quad I(Y Z, Y' Z' X)}{I(Y Y' Z, Z' X)}$ (algebra, \mathcal{A})	

A *probability model* over a finite set of attributes $S = \{s_1, \dots, s_n\}$ is a pair (dom, P) where dom is a domain mapping that maps each s_i to a finite domain $dom(s_i)$, and $P : dom(s_1) \times \dots \times dom(s_n) \rightarrow [0, 1]$ is a probability distribution having the Cartesian product of these domains as its sample space.

Definition 1. The expression $I(Y, Z | X)$ where X, Y and Z are disjoint subsets of S is called a *conditional independence (CI) statement over S* . If $XYZ = S$, we call $I(Y, Z | X)$ a *saturated CI (SCI) statement*. Let (dom, P) be a probability model over S . A CI statement $I(Y, Z | X)$ is said to hold for (dom, P) if for every value \mathbf{x} of X , for every value \mathbf{y} of Y , and for every value \mathbf{z} of Z ,

$$P(\mathbf{y}, \mathbf{z}, \mathbf{x}) \cdot P(\mathbf{x}) = P(\mathbf{y}, \mathbf{x}) \cdot P(\mathbf{z}, \mathbf{x}). \tag{1}$$

Equivalently, (dom, P) is said to satisfy $I(Y, Z | X)$. □

Let $\Sigma \cup \{\varphi\}$ be a set of SCI statements over S . We say that Σ *implies* φ , denoted by $\Sigma \models \varphi$, if every probability model over S that satisfies every SCI statement in Σ also satisfies the SCI statement φ . The *implication problem* is to decide the following problem.

PROBLEM:	Implication problem
INPUT:	Schema S , Set $\Sigma \cup \{\varphi\}$ of SCI statements over S
OUTPUT:	Yes, if $\Sigma \models \varphi$; No, otherwise

For Σ we let $\Sigma^* = \{\varphi \mid \Sigma \models \varphi\}$ be the *semantic closure* of Σ , i.e., the set of all SCI statements implied by Σ . In order to determine the implied SCI statements we use a syntactic approach by applying inference rules. These inference rules have the form

$$\frac{\text{premises}}{\text{conclusion}}$$

and inference rules without any premise are called axioms. An inference rule is called *sound*, if every probability model over S that satisfies every SCI statement in the premises of the rule also satisfies the SCI statement in the conclusion of the rule. We let $\Sigma \vdash_{\mathfrak{R}} \varphi$ denote the *inference* of φ from Σ by the set \mathfrak{R} of inference rules. That is, there is some sequence $\gamma = [\sigma_1, \dots, \sigma_n]$ of SCI statements such that $\sigma_n = \varphi$ and every σ_i is an element of Σ or results from an application of an inference rule in \mathfrak{R} to some elements in $\{\sigma_1, \dots, \sigma_{i-1}\}$. For Σ , let $\Sigma_{\mathfrak{R}}^+ = \{\varphi \mid \Sigma \vdash_{\mathfrak{R}} \varphi\}$ be its *syntactic closure* under inferences by \mathfrak{R} . A set \mathfrak{R} of inference rules is said to be *sound* (*complete*) for the implication of SCI statements, if for every S and for every set Σ of SCI statements over S we have $\Sigma_{\mathfrak{R}}^+ \subseteq \Sigma^*$ ($\Sigma^* \subseteq \Sigma_{\mathfrak{R}}^+$). The (finite) set \mathfrak{R} is said to be a (finite) *axiomatization* for the implication of SCI statements if \mathfrak{R} is both sound and complete.

Theorem 1 (Geiger and Pearl 1993). *The set $\mathfrak{G} = \{\mathcal{T}, \mathcal{S}, \mathcal{C}, \mathcal{W}\}$ forms a finite axiomatization for the implication of SCI statements.* \square

3 A New Completeness Proof

In this section we establish a new proof that \mathfrak{G} is complete for the implication of SCI statements. The new proof introduces a special probability model which consists of two values with probability one half. For the definition of this special probability model we introduce a new inference rule for SCI statements.

3.1 The Algebra Rule

Lemma 1. *The algebra rule \mathcal{A} can be derived from the symmetry \mathcal{S} , weak union \mathcal{W} , and weak contraction rule \mathcal{C} ; and is thus sound.*

Proof. We derive the algebra rule \mathcal{A} from the symmetry \mathcal{S} , weak union \mathcal{W} , and weak contraction rule \mathcal{C} .

$$\frac{\frac{I(YZ, Y'Z' \mid X)}{\mathcal{W} : I(YZ, Z' \mid XY')} \quad \frac{I(YY', ZZ' \mid X)}{\mathcal{S} : I(ZZ', YY' \mid X)}}{\mathcal{C} : I(Z', YY'Z \mid X)} \quad \frac{I(YZ, Y'Z' \mid X)}{\mathcal{S} : I(Z', YZ \mid XY')} \quad \frac{\mathcal{W} : I(ZZ', Z \mid XYY')}{\mathcal{W} : I(Z', Z \mid XYY')}}{\mathcal{S} : I(YY'Z, Z' \mid X)}$$

This concludes the proof. \square

Example 3. Recall the setting of Example [1](#). We can apply the symmetry rule to the SCI statement $I(ARC, FL \mid M)$ from Σ to infer the SCI statement $I(FL, ARC \mid M)$. We can then apply the algebra rule to the SCI statement $I(FL, ARC \mid M)$, and the SCI statement $I(AR, CFL \mid M)$ from Σ to infer the SCI statement $I(ARFL, C \mid M)$; where $Y = \emptyset$, $Y' = FL$, $Z = AR$ and $Z' = C$. A final application of the symmetry rule to $I(ARFL, C \mid M)$ results in φ_1 . Hence, Σ implies φ_1 due to the soundness of the symmetry and algebra rules. \square

3.2 The Independency Basis

For some S , and some set Σ of SCI statements over S , and some $X \subseteq S$ let $IDep_{\Sigma}(X) := \{Y \subseteq S - X \mid \Sigma \vdash_{\mathfrak{G}} I(Y, Z \mid X)\}$ denote the set of all $Y \subseteq S - X$ such that $I(Y, Z \mid X)$ can be inferred from Σ by \mathfrak{G} . The soundness of the algebra rule implies that $(IDep_{\Sigma}(X), \subseteq, \cup, \cap, (\cdot)^c, \emptyset, S - X)$ forms a finite Boolean algebra where $(\cdot)^c$ maps a set W to its complement $S - (XW)$. Recall that an element $a \in P$ of a poset $(P, \subseteq, 0)$ with least element 0 is called an *atom* of $(P, \subseteq, 0)$ precisely when $a \neq 0$ and every element $b \in P$ with $b \subseteq a$ satisfies $b = 0$ or $b = a$. Further, $(P, \subseteq, 0)$ is said to be *atomic* if for every element $b \in P - \{0\}$ there is an atom $a \in P$ with $a \subseteq b$. In particular, every finite Boolean algebra is atomic. Let $IDepB_{\Sigma}(X)$ denote the set of all atoms of $(IDep_{\Sigma}(X), \subseteq, \emptyset)$. We call $IDepB_{\Sigma}(X)$ the *independency basis* of X with respect to Σ . Its importance is manifested in the following result.

Theorem 2. *Let Σ be a set of SCI statements over S . Then $\Sigma \vdash_{\mathfrak{G}} I(Y, Z \mid X)$ if and only if $Y = \bigcup \mathcal{Y}$ for some $\mathcal{Y} \subseteq IDepB_{\Sigma}(X)$.*

Proof. Let $Y \in IDep_{\Sigma}(X)$. Since every element b of a Boolean algebra is the union over those atoms a with $a \subseteq b$ it follows that $Y = \bigcup \mathcal{Y}$ for $\mathcal{Y} = \{W \in IDepB_{\Sigma}(X) \mid W \subseteq Y\}$. Vice versa, let $Y = \bigcup \mathcal{Y}$ for some $\mathcal{Y} \subseteq IDepB_{\Sigma}(X)$. Since $I(W, W' \mid X) \in \Sigma_{\mathfrak{G}}^+$ holds for every $W \in \mathcal{Y}$ successive applications of the algebra rule result in $I(Y, Z \mid X) \in \Sigma_{\mathfrak{G}}^+$. \square

Example 4. Recall Example 1 where Σ consist of $I(ARC, FL \mid M)$ and $I(AR, CFL \mid M)$. It follows that $IDep_{\Sigma}(M) = \{M, AR, C, FL\}$. Indeed, for $I(C, ARFL \mid M)$ we have $C \in IDep_{\Sigma}(M)$, and, thus, $\Sigma \models \varphi_1$. Also, $A \notin IDep_{\Sigma}(M)$ and, thus, $\Sigma \models \varphi_2$ does not hold as $\varphi_2 = I(A, CRFL \mid M)$. \square

3.3 Completeness

The original completeness proof for multivalued dependencies constructs a counter-example relation with 2^k tuples [1], where k denotes the elements in the (in)dependency basis for the multivalued dependency $X \rightarrow Y \mid Z \notin \Sigma^+$. The original completeness proof for SCI statements constructs a probability model with $2^{|X|+1}$ values, where $I(Y, Z \mid X) \notin \Sigma_{\mathfrak{G}}^+$ [6]. Here, a new technique, only recently developed for multivalued dependencies [8], is generalized to define special probability models with two values of probability one half. Our technique therefore extends the existence of special probability models from the case of marginal SCI statements $I(Y, Z \mid \emptyset)$ [6] to general SCI statements.

Theorem 3. *The set \mathfrak{G} is complete for the implication of SCI statements.*

Proof. Let $\Sigma \cup \{I(Y, Z \mid X)\}$ be a set of SCIs over S , and suppose that $I(Y, Z \mid X)$ cannot be inferred from Σ using \mathfrak{G} . We will show that $I(Y, Z \mid X)$ is not implied by Σ . For this purpose, we will construct a probability model that satisfies all SCI statements of Σ , but violates $I(Y, Z \mid X)$.

Let $IDepB_{\Sigma}(X)$ be the disjoint union of $\{\{a\} \mid a \in X\}$ and $\{W_1, \dots, W_k\}$, in particular $S = XW_1 \cdots W_k$. Since $I(Y, Z \mid X) \notin \Sigma_{\mathfrak{G}}^+$ we conclude by Theorem 2 that Y is not the union of some elements of $IDepB_{\Sigma}(X)$. Consequently, there is some $i \in \{1, \dots, k\}$ such that $Y \cap W_i \neq \emptyset$ and $Y - W_i \neq \emptyset$ hold. For every $a \in S$ we define $dom(a) = \{\mathbf{0}, \mathbf{1}\}$. We define the following two values \mathbf{v}_1 and \mathbf{v}_2 of S . We define $\mathbf{v}_1(a) = \mathbf{0}$ for all $a \in S$. We further define $\mathbf{v}_2(a) = \mathbf{1}$ iff $a \in W_i$, and $\mathbf{v}_2(a) = \mathbf{0}$, otherwise. As probability measure we define $P(\mathbf{v}_1) = P(\mathbf{v}_2) = 0.5$. It follows from the construction that (dom, P) does not satisfy $I(Y, Z \mid X)$.

It remains to show that (dom, P) satisfies every SCI statement $I(V, W \mid U)$ in Σ . Suppose that for some value \mathbf{u} of U , $P(\mathbf{u}) = 0$. Then equation (1) will always be satisfied. If $P(\mathbf{u}, \mathbf{v}) = 0$ or $P(\mathbf{u}, \mathbf{w}) = 0$ for some value \mathbf{u} of U , and for some value \mathbf{v} of V or for some value \mathbf{w} of W , then $P(\mathbf{u}, \mathbf{v}, \mathbf{w}) = 0$. Then equation (1) is also satisfied. Suppose that for some value \mathbf{u} of U , $P(\mathbf{u}) = 0.5$. If for some value \mathbf{v} of V and for some value \mathbf{w} of W , $P(\mathbf{u}, \mathbf{v}) = P(\mathbf{u}, \mathbf{w}) = 0.5$, then $P(\mathbf{u}, \mathbf{v}, \mathbf{w}) = 0.5$, too. It remains to consider the case where \mathbf{u} is a value of U such that $P(\mathbf{u}) = 1$. In this case, the construction of the probability model tells us that $U \subseteq XT$ where $T = W_1 \cdots W_{i-1}W_{i+1} \cdots W_k$. Consequently, we can apply the weak union and symmetry rules to $I(V, W \mid U) \in \Sigma$ to infer $I(V - XT, W - XT \mid XT) \in \Sigma_{\mathfrak{G}}^+$. Theorem 2 also shows that $I(W_i, T \mid X) \in \Sigma_{\mathfrak{G}}^+$. However, $W_i = (V - XT)(W - XT)$. An application of the weak contraction rule to $I((V - XT)(W - XT), T \mid X)$ and $I(V - XT, W - XT \mid XT)$ results in $I(V - XT, T(W - XT) \mid X)$. It follows from Theorem 2 that $V - XT$ is the union of elements from $IDepB_{\Sigma}(X)$. Suppose first that $V - XT = W_i$. Then, we are either in the previous case where $P(\mathbf{u}, \mathbf{v}) = 0$ or $P(\mathbf{u}, \mathbf{w}) = 0$; or $P(\mathbf{u}, \mathbf{v}) = 0.5$, $P(\mathbf{u}, \mathbf{w}) = 1$ and $P(\mathbf{u}, \mathbf{v}, \mathbf{w}) = 0.5$. Otherwise, $V - XT = \emptyset$. Then, we are either in the previous case where $P(\mathbf{u}, \mathbf{v}) = 0$ or $P(\mathbf{u}, \mathbf{w}) = 0$, or $P(\mathbf{u}, \mathbf{v}) = 1$, $P(\mathbf{u}, \mathbf{w}) = 0.5$ and $P(\mathbf{u}, \mathbf{v}, \mathbf{w}) = 0.5$. This concludes the proof. \square

We call a probability model (dom, P) over S *special*, if for every $a \in S$, $dom(a)$ consists of two elements, and there are two values v_1, v_2 over S such that $P(v_1) = 0.5 = P(v_2)$. We say that Σ *implies* φ in the world of special probability models, denoted by $\Sigma \models_2 \varphi$, if every special probability model over S that satisfies every SCI statement in Σ also satisfies the SCI statement φ . The following variant of the implication problem for SCI statements emerges.

PROBLEM: Implication problem in the world of special probability models	
INPUT:	Schema S , Set $\Sigma \cup \{\varphi\}$ of SCI statements over S
OUTPUT:	Yes, if $\Sigma \models_2 \varphi$; No, otherwise

The proof of Theorem 3 implies the following result.

Corollary 1. *The implication problem for SCI statements coincides with the implication problem for SCI statements in the world of special probability models.*

Proof. Let $\Sigma \cup \{\varphi\}$ be a set of SCI statements over S . We need to show that $\Sigma \models \varphi$ if and only if $\Sigma \models_2 \varphi$. If it does not hold that $\Sigma \models_2 \varphi$, then it also does not hold that $\Sigma \models \varphi$ since every special probability model is a probability model.

Vice versa, if it does not hold that $\Sigma \models \varphi$, then it does not hold that $\Sigma \vdash_{\mathfrak{G}} \varphi$ since \mathfrak{G} is sound for the implication of SCI statements. However, the proof of Theorem 3 shows how to construct a special probability model that satisfies every SCI statement in Σ but does not satisfy φ . Hence, it does not hold that $\Sigma \models_2 \varphi$. \square

Corollary 1 shows that to decide the implication problem for SCI statements over S it suffices to check special probability models over S .

Example 5. Recall from before that in the setting of Example 1, Σ does not imply $\varphi_2 = I(A, CRFL \mid M)$, where $IDep_{\Sigma}(M) = \{M, AR, C, FL\}$. One may define a special probability model based on the values

(The Seven Samurai, T. Mifune, Kikuchiyo, A. Kurosawa, Subtitle, English) & (The Seven Samurai, T. Shimura, K. Shimada, A. Kurosawa, Subtitle, English).

The special probability model satisfies the SCI statements in Σ but violates φ_2 . Hence, Σ does not imply φ_2 in the world of special probability models. \square

4 Characterization by a Propositional Fragment

In this section we will define the mapping between SCI statements and formulae in a Boolean propositional fragment. We then present a semantic proof for the equivalence of the implication problem for SCI statements and the implication problem for the Boolean propositional fragment. We assume familiarity with basic notions from Boolean propositional logic [3]. We are interested in propositional formulae of the form $(a'_1 \wedge \dots \wedge a'_n) \rightarrow ((b'_1 \wedge \dots \wedge b'_m) \vee (c'_1 \wedge \dots \wedge c'_k))$ where a'_i, b'_j, c'_l denote propositional variables. We assume that conjunction \wedge binds stronger than disjunction \vee , and disjunction \vee binds stronger than material implication \rightarrow . Hence, the formula above becomes: $a'_1 \wedge \dots \wedge a'_n \rightarrow b'_1 \wedge \dots \wedge b'_m \vee c'_1 \wedge \dots \wedge c'_k$. Formulae in this fragment are denoted by either σ' or φ' , and sets of such formulae by Σ' . The satisfaction of a formula φ' by a truth assignment ω' is denoted by $\models_{\omega'} \varphi'$. We also say that ω' is a model of φ' . We write $\Sigma' \models \varphi'$ to say that Σ' logically implies φ' , i.e., every truth assignment that satisfies every formula in Σ' also satisfies φ' .

4.1 The Propositional Fragment \mathcal{F}

As a first step, we define the Boolean fragment \mathcal{F} that corresponds to SCI statements. Let $\phi : S \rightarrow \mathcal{L}$ denote a bijection between a set S of attributes and the set $\mathcal{L} = \{a' \mid a \in S\}$ of propositional variables. We extend ϕ to a mapping Φ from the set of SCI statements over S to the set \mathcal{L} . For an SCI statement $I(Y, Z \mid X)$ over S , let $\Phi(I(Y, Z \mid X))$ denote the formula $\bigwedge_{a \in X} a' \rightarrow \bigwedge_{b \in Y} b' \vee \bigwedge_{c \in Z} c'$. Disjunctions over zero disjuncts are interpreted as \mathbb{F} and conjunctions over zero conjuncts are interpreted as \mathbb{T} . We will simply denote $\Phi(\varphi) = \varphi'$ and $\Phi(\Sigma) = \{\Phi(\sigma) \mid \sigma \in \Sigma\} = \Sigma'$. Example 2 shows the \mathcal{F} -formula that correspond to the SCI statements from Example 1.

4.2 Special Truth Assignments

We will now show that for any set $\Sigma \cup \{\varphi\}$ of SCI statements over S there is a probability model $\pi = (dom, P)$ over S that satisfies Σ and violates φ if and only if there is a truth assignment ω'_π that is a model of Σ' but not a model of φ' . For arbitrary probability models π it is not obvious how to define the interpretation ω'_π . However, the key to showing the correspondence between counterexample probability models and counterexample truth assignments is Corollary [□](#). Corollary [□](#) tells us that for deciding the implication problem $\Sigma \models \varphi$ it suffices to examine special probability models (instead of arbitrary probability models). For a special probability model $\pi = (dom, \{\mathbf{v}_1, \mathbf{v}_2\})$, however, we can define its corresponding special truth assignment ω'_π of \mathcal{L} as follows:

$$\omega'_{(dom, \{\mathbf{v}_1, \mathbf{v}_2\})}(a') = \begin{cases} \mathbb{T} & , \text{ if } \mathbf{v}_1(a) = \mathbf{v}_2(a) \\ \mathbb{F} & , \text{ otherwise} \end{cases} .$$

Next we justify the definition of the special truth assignment and that of the Boolean fragment \mathcal{F} in terms of the special probability models.

Lemma 2. *Let $\pi = (dom, \{\mathbf{v}_1, \mathbf{v}_2\})$ be a special probability model over S , and let φ denote an SCI statement over S . Then π satisfies φ if and only if ω'_π is a model of φ' .*

Proof. Let $\varphi = I(Y, Z \mid X)$ and $\varphi' = \bigwedge_{a \in X} a' \rightarrow \bigwedge_{b \in Y} b' \vee \bigwedge_{c \in Z} c'$. Suppose first that π satisfies φ . We need to show that ω'_π is a model of φ' . Assume that $\omega'_\pi(a') = \mathbb{T}$ for all $a \in X$. According to the special truth assignment we must have $\mathbf{v}_1(a) = \mathbf{v}_2(a)$ for all $a \in X$. That means $P(\mathbf{v}_1(X)) = 1$. Suppose that for some $b \in Y$ we have $\omega'_\pi(b') = \mathbb{F}$. Then $\mathbf{v}_1(b) \neq \mathbf{v}_2(b)$ according to the special truth assignment. Then $P(\mathbf{v}_1(XY)) = P(\mathbf{v}_1) = 0.5$. However, since π satisfies φ we must have $P(\mathbf{v}_1(XZ)) = 1$. Hence, for every $c \in Z$, we cannot have $\mathbf{v}_1(c) \neq \mathbf{v}_2(c)$. This means that for all $c \in Z$ we have $\omega'_\pi(c') \neq \mathbb{F}$. This shows that ω'_π is a model of φ' .

Suppose ω'_π is a model of φ' . We need to show that π satisfies φ . That is, for every value \mathbf{x} of X , and every value \mathbf{y} of Y , and every value \mathbf{z} of Z , that $P(\mathbf{x}) \cdot P(\mathbf{x}, \mathbf{y}, \mathbf{z}) = P(\mathbf{x}, \mathbf{y}) \cdot P(\mathbf{x}, \mathbf{z})$ holds. We distinguish between a few cases.

Case 1. Certainly, if $P(\mathbf{x}, \mathbf{y}) = 0$ or $P(\mathbf{x}, \mathbf{z}) = 0$, then $P(\mathbf{x}, \mathbf{y}, \mathbf{z}) = 0$, too. For the remaining cases we can therefore assume that $P(\mathbf{x}, \mathbf{y}) > 0$ and $P(\mathbf{x}, \mathbf{z}) > 0$. In particular, $P(\mathbf{x}) > 0$.

Case 2. Suppose $P(\mathbf{x}) = 1$. It follows that $\mathbf{v}_1(a) = \mathbf{v}_2(a)$ for all $a \in X$. Consequently, $\omega'_\pi(a') = \mathbb{T}$ for all $a \in X$. Since ω'_π is a model of φ' we conclude that $\omega'_\pi(b') = \mathbb{T}$ for all $b \in Y$, or $\omega'_\pi(c') = \mathbb{T}$ for all $c \in Z$. This, however, would mean that $P(\mathbf{x}, \mathbf{y}) = 1$ or $P(\mathbf{x}, \mathbf{z}) = 1$. Since φ is saturated, it follows that exactly one of $P(\mathbf{x}, \mathbf{y})$ and $P(\mathbf{x}, \mathbf{z})$ is 1, and the other 0.5. Consequently, $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ equals \mathbf{v}_1 or \mathbf{v}_2 . Hence, $P(\mathbf{x}, \mathbf{y}, \mathbf{z}) = 0.5$. It follows that π satisfies φ .

Case 3. Suppose that $P(\mathbf{x}) = 0.5$. Then $P(\mathbf{x}, \mathbf{y}) = 0.5 = P(\mathbf{x}, \mathbf{z})$. Then $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ equals \mathbf{v}_1 or \mathbf{v}_2 , as $P(\mathbf{x})$ would have to be 1 otherwise. Hence, $P(\mathbf{x}, \mathbf{y}, \mathbf{z}) = 0.5$. □

The equivalence between special probability models and special truth assignments extend the existing equivalence between two-tuple relations for multivalued dependencies and special truth assignments [4,8,16].

4.3 The Equivalence

Corollary 1 and Lemma 2 allow us to establish the anticipated equivalence between probability models and propositional truth assignments.

Theorem 4. *Let $\Sigma \cup \{\varphi\}$ be a set of SCI statements over some set S of attributes, and let $\Sigma' \cup \{\varphi'\}$ denote the set of its corresponding Boolean formulae over \mathcal{L} . Then $\Sigma \models \varphi$ if and only if $\Sigma' \models \varphi'$.*

Proof. Based on Corollary 1 it remains to establish the equivalence between $\Sigma \models_2 \varphi$ and $\Sigma' \models \varphi'$. Suppose first that $\Sigma \models_2 \varphi$ does not hold. Then there is some special probability model π over S that satisfies every SCI statement σ in Σ but violates φ . Let ω'_π denote the special truth assignment associated with π . By Lemma 2 it follows that ω'_π satisfies every formula σ' in Σ' but violates φ' . Consequently, $\Sigma' \models \varphi'$ does not hold. Suppose now that $\Sigma' \models \varphi'$ does not hold. Then there is some truth assignment ω' over \mathcal{L} that is a model for every formula σ' in Σ' , but not a model for the formula φ' . Define the following special probability model $\pi = (dom, P)$ over S : for all $a \in S$, let $v_1(a) = v_2(a)$ if and only if $\omega'(a) = \mathbb{T}$. It follows that $\omega'_\pi = \omega'$. By Lemma 2 it follows that π satisfies every SCI statement σ in Σ but violates φ . Hence, $\Sigma \models_2 \varphi$ does not hold. \square

Example 2 illustrates the equivalences of the special truth assignments and probability models.

5 Conclusion

Saturated conditional independence statements form an important subclass of conditional independence statements since their associated implication problem is finitely axiomatizable and decidable in almost linear time. We have established a new completeness proof for Geiger and Pearl's axiomatization of SCI statements. The proof shows that, in order to decide the implication problem of SCI statements over discrete probability models, it suffices to consider special binary probability models where two values have probability one half. Special probability models allow us to establish a semantic proof for the equivalence between the implication problem of i) SCI statements, ii) a fragment of Boolean propositional logic, and iii) multivalued dependencies in relational and nested relational databases.

Acknowledgement. This research is supported by the Marsden Fund Council from Government funding, administered by the Royal Society of New Zealand. I would like to thank the anonymous reviewers for helping to improve the presentation, and pointing out that CI statements can be expressed by polynomial likelihood formulae.

References

1. Beeri, C., Fagin, R., Howard, J.H.: A complete axiomatization for functional and multivalued dependencies in database relations. In: Smith, D. (ed.) SIGMOD Conference, pp. 47–61. ACM (1977)
2. Biskup, J., Link, S.: Appropriate inferences of data dependencies in relational databases. *Ann. Math. Artif. Intell.* 63(3-4), 213–255 (2012)
3. Enderton, H.: *A Mathematical Introduction to Logic*. Academic Press (2001)
4. Fagin, R.: Functional dependencies in a relational data base and propositional logic. *IBM Journal of Research and Development* 21(6), 543–544 (1977)
5. Galil, Z.: An almost linear-time algorithm for computing a dependency basis in a relational database. *J. ACM* 29(1), 96–102 (1982)
6. Geiger, D., Pearl, J.: Logical and algorithmic properties of conditional independence and graphical models. *The Annals of Statistics* 21(4), 2001–2021 (1993)
7. Halpern, J.: *Reasoning about uncertainty*. MIT Press (2005)
8. Hartmann, S., Link, S.: The implication problem of data dependencies over SQL table definitions: Axiomatic, algorithmic and logic characterizations. *ACM Trans. Datab. Syst.* 13, Article 13 (May 2012)
9. Hartmann, S., Link, S.: On a problem of Fagin concerning multivalued dependencies in relational databases. *Theor. Comput. Sci.* 353(1-3), 53–62 (2006)
10. Hartmann, S., Link, S.: Characterising nested database dependencies by fragments of propositional logic. *Ann. Pure Appl. Logic* 152(1-3), 84–106 (2008)
11. Link, S.: Charting the completeness frontier of inference systems for multivalued dependencies. *Acta Inf.* 45(7-8), 565–591 (2008)
12. Link, S.: Characterisations of multivalued dependency implication over undetermined universes. *J. Comput. Syst. Sci.* 78(4), 1026–1044 (2012)
13. Niepert, M., Van Gucht, D., Gyssens, M.: Logical and algorithmic properties of stable conditional independence. *Int. J. Approx. Reasoning* 51(5), 531–543 (2010)
14. Niepert, M., Van Gucht, D., Gyssens, M.: On the conditional independence implication problem: A lattice-theoretic approach. In: McAllester, D., Myllymäki, P. (eds.) *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence - UAI*, pp. 435–443 (2008)
15. Pearl, J.: *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann (1988)
16. Sagiv, Y., Delobel, C., Parker Jr., D., Fagin, R.: An equivalence between relational database dependencies and a fragment of propositional logic. *J. ACM* 28(3), 435–453 (1981)
17. Studený, M.: Conditional independence relations have no finite complete characterization. In: Kubik, S., Visek, J. (eds.) *Transactions of the 11th Prague Conference on Information Theory*, pp. 377–396. Kluwer (1992)
18. Wong, S., Butz, C., Wu, D.: On the implication problem for probabilistic conditional independency. *IEEE Transactions on Systems, Man, and Cybernetics, Part A* 30(6), 785–805 (2000)

Contracting Logics

Márcio M. Ribeiro and Marcelo E. Coniglio

Centre for Logic, Epistemology and the History of Science
State University of Campinas, Brazil
{marcio,coniglio}@cle.unicamp.br

Abstract. In this paper, inspired in the field of belief revision, it is presented a novel operation for defining a new logic given a known logic. The operation consists in removing some (maybe undesirable) derived rule from a logic. Besides removing the ‘undesirable’ rule, this operation (called contraction) should change the logic in a minimal way. This paper presents formal definitions for contraction operations over logics, both as sets of rationality postulates and by means of concrete constructions. This allowed us to generalize several notions of maximality of logics presented in the literature. Furthermore, the proposed constructions are applied to the study of some paraconsistent and intermediate logics.

1 Introduction

Belief revision is a subfield of knowledge representation that studies the dynamics of propositional theories [AGM85]. The dynamics of the theories is given by a set of operations (contraction, revision, expansion etc.) which are defined via sets of rationality postulates.

The first papers in the field restricted themselves to the study of the dynamics of theories within supra-classical logics, but recently it was showed how this can be generalized to several other non-classical logics [Rib12]. In this paper it is presented a way to generalize belief revision techniques even more.

Instead of considering operations for changing a theory within a given logic, it is presented operations that change the *logic* itself. This paper focuses on contraction operations i.e. operations that given a logic returns another logic where certain rule doesn’t hold.

Firstly a set of rationality postulates that the operation should satisfy is presented. This includes some kind of minimality criterion concerning the change needed to perform the operation. The techniques involved lead us naturally to the notion of maximality of logics. The constructions proposed here generalize several notions of maximality considered in the literature. After this, representation theorems relating the constructions defined here with sets of postulates are obtained. Finally, several examples involving paraconsistent, paracomplete and super-intuitionistic logics are shown.

2 Preliminaries

In this section we recall the basic notions to be used along the paper.

Definition 1. A (Tarskian) consequence system is a pair $\langle For, Cn \rangle$ such that For is a set (whose elements are called formulas) and $Cn : \wp For \rightarrow \wp For$ is a map satisfying, for every $\Gamma, \Delta \subseteq For$:

- $\Gamma \subseteq Cn(\Gamma)$ (extensiveness)
- if $\Gamma \subseteq \Delta$ then $Cn(\Gamma) \subseteq Cn(\Delta)$ (monotonicity)
- $Cn(Cn(\Gamma)) \subseteq Cn(\Gamma)$ (idempotence)

The map $Cn : \wp For \rightarrow \wp For$ is called a *consequence operator*. We say that $\langle For, Cn \rangle$ is *compact* if $Cn(\Gamma) = \bigcup \{Cn(\Gamma_0) : \Gamma_0 \text{ is a finite subset of } \Gamma\}$.

The usual consequence systems are defined over formal languages generated by signatures.

Definition 2. A (propositional) signature is a family of sets $C = \{C_n\}_{n \in \mathbb{N}}$ such that $C_i \cap C_j = \emptyset$ if $i \neq j$, and $C_0 \neq \emptyset$.

The elements of C_n are called *n-ary connectives*. The elements of C_0 , in particular, are called *propositional variables*. Let Ξ be a given set of symbols called *schema variables*.

Definition 3. Let $C = \{C_n\}_{n \in \mathbb{N}}$ be a signature. The propositional schema language generated by C is the algebra $L(C, \Xi)$ of type C freely generated by $\Xi \cup C_0$. The propositional language generated by C is the algebra $L(C)$ of type C freely generated by C_0 .

The elements of $L(C, \Xi)$ (of $L(C)$, resp.) are called *schema formulas* (formulas, resp.) over C . Note that $C_0 \subseteq L(C) \subseteq L(C, \Xi)$.

Definition 4. A Hilbert calculus is a pair $H = \langle C, R \rangle$ such that C is a signature, and R is a set of inference rules, that is, a set of pairs $\langle \Delta, \varphi \rangle$ where $\Delta \cup \{\varphi\}$ is a finite subset of $L(C, \Xi)$.

When $\Delta = \emptyset$ we say that the inference rule is an *axiom*. The notion of derivation in a Hilbert calculus is the usual one. Before that it is necessary to introduce the notion of substitution.

A *substitution over C* is a map $\sigma : \Xi \rightarrow L(C, \Xi)$. A substitution can be extended to a unique C -homomorphism $\hat{\sigma} : L(C, \Xi) \rightarrow L(C, \Xi)$ as usual. We denote by $\hat{\sigma}(\Delta)$ the set of schema formulas $\{\hat{\sigma}(\psi) : \psi \in \Delta\}$, for $\Delta \subseteq L(C, \Xi)$. Finally, let $Subs(C) = \{\sigma : \sigma \text{ is a substitution over } C\}$.

Definition 5. A derivation of $\varphi \in L(C, \Xi)$ in H from a set $\Gamma \subseteq L(C, \Xi)$ is a sequence $\varphi_1 \dots \varphi_n$ such that $\varphi_n = \varphi$ and, for $i = 1, \dots, n$, each φ_i is either an element of Γ or there is a substitution σ and an inference rule $\langle \Delta, \varphi \rangle$ in H such that $\hat{\sigma}(\varphi)$ is φ_i and, for every $\psi \in \Delta$, $\hat{\sigma}(\psi)$ is φ_j for some $j < i$.

We say that φ is derived from Γ in H , denoted by $\Gamma \vdash_H \varphi$, if there is a derivation of it in H from Γ .

If H is a Hilbert calculus then its set of rules will be frequently denoted by R_H . Given two Hilbert calculi $H_i = \langle C, R_i \rangle$ over C (for $i = 1, 2$) we say that H_1 is a subcalculus of H_2 , denoted by $H_1 \subseteq H_2$, if $R_1 \subseteq R_2$. Given $H = \langle C, R \rangle$ over C then the *closure* of the set of rules R is the set of rules $Cl(R) = \{ \langle \Delta, \varphi \rangle : \Delta \vdash_H \varphi \}$. The *closure of H* is the Hilbert calculus $Cl(H) = \langle C, Cl(R) \rangle$. If $r \in Cl(R)$ then r is a *derived rule* of H . Let $Cn_H : \wp(L(C, \Xi)) \rightarrow \wp(L(C, \Xi))$ be the consequence operator generated by H as expected: $Cn_H(\Gamma) = \{ \varphi : \Gamma \vdash_H \varphi \}$. This consequence operator is Tarskian and *structural*, that is: for every substitution σ and every set of schema formulas Γ , $\hat{\sigma}(Cn_H(\Gamma)) \subseteq Cn_H(\hat{\sigma}(\Gamma))$. We say that $\langle L(C, \Xi), Cn_H \rangle$ is the consequence system, associated to the Hilbert calculus H . If $R(C) =_{def} \wp_f(L(C, \Xi)) \times L(C, \Xi)$ denotes the set of all rules over C , it is easy to see that $\langle R(C), Cl \rangle$ is a compact and structural consequence system, where $\wp_f(X)$ denotes the set of finite subsets of a set X , and where $\hat{\sigma}(r)$ is defined in the obvious way for every $r \in R(C)$ and every $\sigma \in Subs(C)$.

3 Rationality Postulates

Given a Hilbert calculus H over C we want to define an operation $-$ that for each rule $r = \langle \Delta, \varphi \rangle$ over C returns a Hilbert calculus $H - r$ over C that satisfies certain properties:

(inclusion) $H - r \subseteq H$

Inclusion states that the result of a contraction in H is a subcalculus of H .

(success) If $\varphi \notin \Delta$ (i.e. $r \notin Cl(\emptyset)$) then $\Delta \not\vdash_{H-r} \varphi$.

Success states that the resulting calculus should not derive the rule r , unless this is impossible. The only case when it is impossible to remove r from H is when φ is a substitution of some element of Δ i.e. if r is a derived rule in *any* Hilbert calculus, by extensiveness.

(failure) If $\varphi \in \Delta$ (i.e. $r \in Cl(\emptyset)$) then $H \subseteq H - r$.

Failure states that if r is a derived rule in *any* Hilbert calculus, then the contraction $-$ over H should not remove any element from H . Together with inclusion, failure postulate states that $H - r = H$ if $r \in Cl(\emptyset)$. In other words, in this case the contraction should fail.

Besides these, we need some postulate that guaranties the minimality of change i.e. that guarantees that only derivations relevant to prove r should be removed. Next section investigate possible minimality postulates.

So far postulates for contraction over an arbitrary Hilbert calculus were presented. A criticism one can make to this approach is that it compromises itself with one specific axiomatization for a logic.

One way to abstract away the specificity of certain choice of a set of rules R is to consider the closure of $H = \langle C, R \rangle$, that is, the calculus $Cl(H) = \langle C, Cl(R) \rangle$. Observe that $\langle L(C, \Xi), Cl(R) \rangle$ is a Tarskian, compact and structural consequence system which represents the logic generated by H . For this reason it

is more robust to consider the contraction over the closure of H instead of H itself.

Let us then define a contraction $-$ over a closed Hilbert calculus H (i.e. where $H = Cl(H)$). In this case we want the result of a contraction $-$ over H to be also a closed Hilbert calculus:

$$\text{(closure)} \quad H - r = Cl(H - r)$$

The other postulates for contraction over closed Hilbert calculi are the same we already stated.

3.1 Minimality Criteria

In belief revision literature we can find several minimality postulates: recovery, relevance, core-retainment, fullness etc. (see [Han99]). We will focus here in two of these postulates, namely, relevance and fullness.

$$\text{(fullness)} \quad \text{If } r' \in R_H \text{ and } r' \notin R_{H-r} \text{ then } r \in Cl(R_{H-r} \cup \{r'\}).$$

Fullness states that if some rule r' was removed from H after contraction then re-inserting r' should recover the rule r .

It will be sometimes useful to weaken this postulate to guarantee that only axioms $r' = \langle \emptyset, \varphi' \rangle$ may be removed from H .

$$\text{(weak fullness)} \quad \text{If } r' \in R_H \text{ is an axiom and } r' \notin R_{H-r} \text{ then } r \in Cl(R_{H-r} \cup \{r'\}).$$

However, it is pointed out in the literature that these postulates may be too strong for certain purposes. Relevance is a weaker version of this postulate (see [Han91]):

$$\text{(relevance)} \quad \text{If } r' \in R_H \text{ and } r' \notin R_{H-r} \text{ then there is } H' \text{ such that } H - r \subseteq H' \subseteq H, r \notin Cl(H') \text{ and } r \in Cl(R_{H'} \cup \{r'\}).$$

Relevance states that if r' is removed then there is some intermediary calculus H' such that r is not a derived rule in H' but r is a derived rule if r' is added. Again we can define a weak version of the postulate which is concerned only with derived axioms, not derived rules in general:

$$\text{(weak relevance)} \quad \text{If } r' \in R_H \text{ is an axiom and } r' \notin R_{H-r} \text{ then there is } H' \text{ such that } H - r \subseteq H' \subseteq H, r \notin Cl(H') \text{ and } r \in Cl(R_{H'} \cup \{r'\}).$$

Notice that relevance implies failure:

Lemma 6. *Let H be a Hilbert calculus over C . If $-$ over H satisfies relevance then H satisfies failure.*

As will be shown in section 4, these minimality postulates are related with a construction called *remainder set*.

4 Maximality

In previous sections we presented a set of postulates for a contraction operation over a Hilbert calculus which may or may not be closed. We argued that minimality is a desirable property of the operation, i.e. contraction should change the logic as little as possible. This desiderata is closely related to the notion of maximal logics. In this section we will investigate different definitions for maximal logic presented in the literature and we will present a definition which generalizes them.

Usually, a logic \mathbf{L}_1 is said to be *maximal* w.r.t. another logic \mathbf{L}_2 when both are defined over the same language, the consequence relation $\vdash_{\mathbf{L}_1}$ of \mathbf{L}_1 is contained in $\vdash_{\mathbf{L}_2}$ and, if φ is a schema formula such that $\vdash_{\mathbf{L}_2} \varphi$ but $\not\vdash_{\mathbf{L}_1} \varphi$ then the extension of \mathbf{L}_1 obtained by adding φ as a valid schema coincides with \mathbf{L}_2 . In our framework if \mathbf{L}_1 is given by the closure of a Hilbert calculus H_1 then $Cl(R_{\mathbf{L}_1} \cup \{\langle \emptyset, \varphi \rangle\}) = R_{\mathbf{L}_2}$ whenever $\langle \emptyset, \varphi \rangle \in R_{\mathbf{L}_2} \setminus R_{\mathbf{L}_1}$.

For example, a logic over a signature C is *Post complete* if it is maximal w.r.t. the trivial logic $\mathbf{Triv}_C = \langle C, R(C) \rangle$ over C .

Another definition of maximality found in the literature comes from [AAZ10]. In this article the authors defined a logic \mathbf{L} as being maximal w.r.t. a rule (in their case the principle of explosion $\langle \{-\xi_1, \xi_1\}, \xi_2 \rangle$). They define two types of maximality: strong and weak. The logic \mathbf{L}_1 seeing as a closed Hilbert calculus is strongly maximal w.r.t. a rule $r \notin R_{\mathbf{L}_1}$ if for every logic \mathbf{L}_2 such that $R_{\mathbf{L}_1} \subset R_{\mathbf{L}_2}$ we have that $r \in Cl(\mathbf{L}_2)$. Using this definition the authors proved that several three-valued logics, such as \mathbf{P}^1 and \mathbf{J}_3 , are maximal w.r.t. the principle of explosion.

We will borrow a concept from belief revision literature to generalize the notions of maximality.

Definition 7 (Remainder set). *Let H be a Hilbert calculus over C and R be a set of rules over C . A remainder set $H \perp R$ is a set such that $X \in H \perp R$ iff:*

- 1 $X \subseteq H$ (X is a subcalculus of H).
- 2 $R \not\subseteq Cl(R_X)$ (there is some $r \in R$ that is not a derived rule in X).
- 3 If $X \subset X' \subseteq H$ then $R \subseteq Cl(R_{X'})$ (X is maximal).

The remainder set $H \perp R$ is the set of all maximal subcalculus X of H such that some rule in R is not a derived rule in X . We can also define a weak version of remainder set, denoted by $H \perp_w R$, by changing item 3 in the above definition to:

- 3' For any axiom $r \in R_H \setminus R_X$ we have that $R \subseteq Cl(R_X \cup \{r\})$.

Example 8. *Consider the following Hilbert calculus $H_{CPL} = \langle C, R \rangle$ for Classical Propositional Logic (CPL). Let $C = \{\mathbb{P}, \{\neg\}, \{\wedge, \vee, \rightarrow\}\}$ where \mathbb{P} denote the set of propositional variables, and let R be the following set of rules:*

¹ In this example we will identify an axiom $\langle \emptyset, \varphi \rangle$ with φ .

$$\begin{aligned}
 (Ax_1) \quad & \xi_1 \rightarrow (\xi_2 \rightarrow \xi_1) & (Ax_2) \quad & \xi_1 \rightarrow (\xi_2 \rightarrow (\xi_1 \wedge \xi_2)) \\
 (Ax_3) \quad & (\xi_1 \wedge \xi_2) \rightarrow \xi_1 & (Ax_4) \quad & (\xi_1 \wedge \xi_2) \rightarrow \xi_2 \\
 (Ax_5) \quad & \xi_1 \rightarrow (\xi_1 \vee \xi_2) & (Ax_6) \quad & \xi_2 \rightarrow (\xi_1 \vee \xi_2) \\
 (Ax_7) \quad & \xi_1 \rightarrow (\neg \xi_1 \rightarrow \xi_2) & (Ax_8) \quad & \xi_1 \vee \neg \xi_1 \\
 (Ax_9) \quad & (\xi_1 \rightarrow (\xi_2 \rightarrow \xi_3)) \rightarrow ((\xi_1 \rightarrow \xi_2) \rightarrow (\xi_1 \rightarrow \xi_3)) \\
 (Ax_{10}) \quad & (\xi_1 \rightarrow \xi_2) \rightarrow ((\xi_3 \rightarrow \xi_2) \rightarrow (\xi_1 \vee \xi_3 \rightarrow \xi_2)) \\
 (Ax_{11}) \quad & (\xi_1 \rightarrow \xi_2) \rightarrow ((\xi_1 \rightarrow \neg \xi_2) \rightarrow \neg \xi_1) \\
 (MP) \quad & \langle \{\xi_1, \xi_1 \rightarrow \xi_2\}, \xi_2 \rangle
 \end{aligned}$$

Let $r = \langle \{\neg \neg \xi\}, \xi \rangle$. It is well known that $r \in Cl(H_{\mathbf{CPL}})$. Consider now the Hilbert calculus $H_{\mathbf{Int}} = \langle C, R \setminus \{Ax_8\} \rangle$ for Intuitionistic logic. It is also well known that $r \notin Cl(H_{\mathbf{Int}})$. Hence, it is trivial to show that $H_{\mathbf{Int}} \in H_{\mathbf{CPL}} \perp \{r\}$.

Notice that this is strongly dependent on the choice of the rules. This is why it is important to consider the contraction operation not over arbitrary Hilbert calculi, but over closed Hilbert calculi².

The above example shows the limitation of using an arbitrary Hilbert calculus H in the definition of remainder set. We will be more interested in applications where H is closed. In this case, we can prove that the elements of $H \perp R$ are also closed:

Lemma 9. *If $H = Cl(H)$ and $X \in H \perp R$ then $X = Cl(X)$.*

As a first application of our framework, let us show that the notions of maximality presented above can be represented using remainder sets:

- A logic \mathbf{L}_1 seeing as a closed Hilbert calculus is maximal w.r.t. a logic \mathbf{L}_2 iff $\mathbf{L}_1 \in \mathbf{L}_2 \perp_w R_{\mathbf{L}_2}$.
- A logic \mathbf{L} over a signature C is strongly maximal w.r.t. a rule r iff $\mathbf{L} \in \mathbf{Triv}_C \perp \{r\}$.
- A logic \mathbf{L} over C is weakly maximal w.r.t. a rule r iff $\mathbf{L} \in \mathbf{Triv}_C \perp_w \{r\}$.
- A logic \mathbf{L} over C is Post complete iff $\mathbf{L} \in \mathbf{Triv}_C \perp_w R_{\mathbf{Triv}_C}$.

Now let us show some useful propositions about remainder sets.

Proposition 10. *Let H be an arbitrary Hilbert calculus and let R_1 and R_2 be sets of rules. If $X \in H \perp R_1$, $R_2 \subseteq R_1 \subseteq R_H$ and $R_2 \not\subseteq Cl(R_X)$ then $X \in H \perp R_2$.*

This proposition states that if we know that $X \in H \perp R_1$ and we know that a subset R_2 of R_1 also contains some rule that is not derived from X then $X \in H \perp R_2$.

Recall that \mathbf{P}^1 was introduced in [Set73] as a three valued paraconsistent logic axiomatized by a Hilbert calculus over a signature just containing \neg and \rightarrow . We know from [Set73] that \mathbf{P}^1 is maximal w.r.t. \mathbf{CPL} , that is $\mathbf{P}^1 \in \mathbf{CPL} \perp_w R_{\mathbf{CPL}}$, when both logics are considered as closed Hilbert calculi. Furthermore, we know that $r = \langle \{\xi_1, \neg \xi_1\}, \xi_2 \rangle \notin R_{\mathbf{P}^1}$ and that $r \in R_{\mathbf{CPL}}$. Hence, we have the following corollary of Proposition 10:

² Notice that similar problems arise when using belief base contraction instead of belief set contraction.

Corollary 11. $\mathbf{P}^1 \in \mathbf{CPL}\perp_w\{\{\xi_1, \neg\xi_1\}, \xi_2\}$.

The logic \mathbf{I}^1 was introduced in [SC95] as a three valued paracomplete logic (dual to \mathbf{P}^1) axiomatized by a Hilbert calculus over a signature just containing \neg and \rightarrow . From [SC95] we know that \mathbf{I}^1 is maximal w.r.t. \mathbf{CPL} , i.e. $\mathbf{I}^1 \in \mathbf{CPL}\perp_w R_{\mathbf{CPL}}$. Furthermore, the rule of double negation $r = \langle\{\neg\neg\xi_1\}, \xi_1\rangle$ doesn't hold in \mathbf{I}^1 .

The Smetanich's logic \mathbf{Sm} is another example of logic where the rule of double negation fails. \mathbf{Sm} is the greatest super-intuitionistic logic (cf. [CZ97]), i.e. $\mathbf{Int} \subset \mathbf{Sm} \in \mathbf{CPL}\perp_w R_{\mathbf{CPL}}$. From this, clearly \mathbf{I}^1 is not super-intuitionistic.

Since $\langle\{\neg\neg\xi_1\}, \xi_1\rangle \in R_{\mathbf{CPL}}$, we have the following corollaries:

Corollary 12. $\mathbf{I}^1 \in \mathbf{CPL}\perp_w\{\{\neg\neg\xi_1\}, \xi_1\}$.

Corollary 13. $\mathbf{Sm} \in \mathbf{CPL}\perp_w\{\{\neg\neg\xi_1\}, \xi_1\}$.

Proposition 10 showed a relation that holds when we fix the Hilbert calculus H and change the set of rules. The following proposition states a relation that holds when we fix the set of rules R and change the Hilbert calculus.

Proposition 14. *Let H_1 and H_2 be arbitrary Hilbert calculi and let R be a set of rules. If $X \in H_1 \perp R$ and $X \subseteq H_2 \subseteq H_1$ then $X \in H_2 \perp R$.*

The logic \mathbf{J}_3 , introduced in [DdC70], is a paraconsistent three valued logic that can be defined over the signature $C = \langle\mathbb{P} \cup \{\perp\}, \{\neg\}, \{\wedge, \vee, \rightarrow\}\rangle$. From [AAZ10] we know that \mathbf{J}_3 is strongly maximal w.r.t. the principle of explosion, that is $\mathbf{J}_3 \in \mathbf{Triv}_C \perp \{\{\neg\xi_1, \xi_1\}, \xi_2\}$. The following corollary is a consequence of the fact that $\mathbf{J}_3 \subseteq \mathbf{CPL} \subseteq \mathbf{Triv}_C$:

Corollary 15. $\mathbf{J}_3 \in \mathbf{CPL}\perp\{\{\neg\xi_1, \xi_1\}, \xi_2\}$.

The following results analyze the connection between the notions of remainder sets and weak remainder sets just introduced. The main purpose is to obtain sufficient conditions in order to guarantee the equivalence between both notions.

Lemma 16. *Let $H = \langle C, R \rangle$ be a Hilbert calculus over C , $r = \langle\{\gamma_1, \dots, \gamma_n\}, \varphi\rangle$ a rule over C and $H^r = \langle C, R \cup \{r\}\rangle$. Assume that H has (possibly derived) connectives \rightarrow (binary) and \sim (unary) such that the following holds:*

1. $\xi_1, (\xi_1 \rightarrow \xi_2) \vdash_H \xi_2$;
2. $\vdash_H \xi_1 \rightarrow (\xi_2 \rightarrow \xi_1)$;
3. $\sim\xi_1 \vdash_H (\xi_1 \rightarrow \xi_2)$;
4. *If $\Gamma, \xi_1 \vdash_{H^r} \xi_2$ and $\Gamma, \sim\xi_1 \vdash_{H^r} \xi_2$ then $\Gamma \vdash_{H^r} \xi_2$, for every Γ .*

Let ξ be a schema variable not occurring in r . Then

$$(\xi \rightarrow \gamma_1), \dots, (\xi \rightarrow \gamma_n) \vdash_{H^r} (\xi \rightarrow \varphi).$$

Corollary 17. *Let H and H^r as in Lemma 16. Assume that H satisfies the Deduction Meta-Theorem (MTD) with respect to \rightarrow : $\Gamma, \alpha \vdash_H \beta$ implies $\Gamma \vdash_H (\alpha \rightarrow \beta)$, for every α, β . Then H^r also satisfies MTD with respect to \rightarrow .*

Corollary 18. *Let H and H^r as in Corollary 17 for $r = \langle \{\gamma_1, \dots, \gamma_n\}, \varphi \rangle$. Let $H_r = \langle C, R \cup \{\langle \emptyset, \gamma_1 \rightarrow (\gamma_2 \rightarrow (\dots \rightarrow (\gamma_n \rightarrow \varphi) \dots)) \rangle\} \rangle$. Then the consequence systems generated by H^r and H_r coincide, that is: $Cl(H^r) = Cl(H_r)$.*

Theorem 19. *Let H and H' be closed Hilbert calculi over C such that $H \subseteq H'$. Assume that H satisfies the conditions of Corollary 17 for every rule r over C , and that H' satisfies MTD with respect to \rightarrow . Then, $H \in H' \perp_w R$ iff $H \in H' \perp R$ for every $R \subseteq R(C)$.*

With Theorem 19 we can now prove that both \mathbf{I}^1 and \mathbf{P}^1 are not only weakly maximal, but also strongly maximal w.r.t. **CPL**:

Proposition 20. *Let **CPL** be the Classical Propositional Logic in the signature containing just \rightarrow and \neg , seeing as closed Hilbert set. Then we have the following:*

1. $\mathbf{I}^1 \in \mathbf{CPL} \perp R_{\mathbf{CPL}}$ and
2. $\mathbf{I}^1 \in \mathbf{CPL} \perp \{\langle \neg \xi_1, \xi_1 \rangle\}$.

Proposition 21. *Let **CPL** be the Classical Propositional Logic as in Proposition 20. Then the following holds:*

1. $\mathbf{P}^1 \in \mathbf{CPL} \perp R_{\mathbf{CPL}}$ and
2. $\mathbf{P}^1 \in \mathbf{CPL} \perp \{\langle \neg \xi_1, \xi_1 \rangle, \xi_2 \rangle\}$.

5 Representation Theorems

In Section 3 we enumerated a set of rationality postulates for a contraction operation. In Section 4 we claimed that this operation is related with a notion of maximality which we explored. This section presents constructions for contraction. Each of them is proved to be characterized by a specific set of rationality postulates.

The following is an important lemma related to remainder sets called *upper-bound lemma*. This result was adapted from a similar one in belief revision field found in [AM81].

Lemma 22 (upper-bound). *Let H and X be Hilbert Calculi such that $X \subseteq H$ and let R be a finite set of rules, all over a signature C . If $R \not\subseteq Cl(X)$ then*

1. *there is some H' such that $X \subseteq H' \in H \perp R$ and*
2. *there is some H'' such that $X \subseteq H'' \in H \perp_w R$.*

Consider now a Hilbert calculus H and a rule r both over the same signature. A (strong) *subset selection function* is any function Υ that satisfies the following properties:

1. $\Upsilon(H, r) \neq \emptyset$;
2. $\Upsilon(H, r) \subseteq H \perp \{r\}$ if $H \perp \{r\} \neq \emptyset$;
3. $\Upsilon(H, r) = \{H\}$ if $H \perp \{r\} = \emptyset$.

A *weak subset selection function* is defined analogously using weak remainder set instead of remainder set. Now consider the following construction for some selection function Υ :

$$H -_{\Upsilon} r = \bigcap \Upsilon(H, r)$$

In the belief revision field this construction is called *partial meet contraction* (cf. [AGM85]). Any partial meet contraction satisfies *success*, *inclusion* and *failure*. It also satisfies *relevance* or *weak relevance* depending if Υ is a weak or a strong selection function. Furthermore, if H is closed, by Lemma 9 and the fact that the intersection of closed Hilbert calculi is closed, partial meet contraction satisfies *closure*.

Theorem 23. *For any weak subset selection function Υ , the contraction $-_{\Upsilon}$ over H defined as $H -_{\Upsilon} r = \bigcap \Upsilon(H, r)$ satisfies success, inclusion, failure and weak relevance. Furthermore:*

- If Υ is a (strong) subset selection function then $-_{\Upsilon}$ also satisfies relevance.
- If H is closed then $-_{\Upsilon}$ satisfies closure

Besides satisfying the postulates, the following theorem shows that, in fact, the postulates fully characterize the construction.

Theorem 24. *Let H be a Hilbert calculus. If a contraction $-$ over H satisfies success, inclusion, failure and weak relevance then there is some weak subset selection function Υ such that:*

$$H - r = H -_{\Upsilon} r = \bigcap \Upsilon(H, r)$$

If $-$ satisfies relevance then the above equation holds for some (not weak) subset selection function. In this case, failure property become redundant by Lemma 6.

Now let us define an *element selection function* as any function Ψ such that:

1. $\Psi(H, r) \in H \perp \{r\}$ if $H \perp \{r\} \neq \emptyset$.
2. $\Psi(H, r) = H$ if otherwise.

Weak element selection function is defined analogously using a weak remainder set. A *maxi-choice contraction* is defined as follows:

$$H -_{\Psi} r = \Psi(H, r)$$

Weak maxi-choice contraction is defined analogously.

As for partial meet contraction, maxi-choice contraction is fully characterized by a set of postulates, namely: *success*, *inclusion*, *failure* and *fullness*. Furthermore, weak maxi-choice contraction is characterized by the same postulates with fullness exchanged by *weak fullness*.

Theorem 25. *Let H be a Hilbert calculus. An operation $-$ over H is a maxi-choice contraction iff it satisfies success, inclusion, failure and fullness. $-$ is a weak maxi-choice contraction iff instead of fullness it satisfies weak fullness.*

Example 26. *Consider \mathbf{P}^1 and \mathbf{CPL} defined in the same signature of \mathbf{J}_3 and let $r = \langle \{\neg\xi_1, \xi_1\}, \xi_2 \rangle$. For certain choice of element selection functions we have that $\mathbf{P}^1 = \mathbf{CPL} -_{\psi_1} r$ and $\mathbf{J}_3 = \mathbf{CPL} -_{\psi_2} r$. Furthermore, for certain choice of subset selection function Υ , we have $\mathbf{P}^1 \cap \mathbf{J}_3 = \mathbf{CPL} -_{\Upsilon} r$. An analogous result can be obtained by considering weak selection functions, \mathbf{I}^1 , \mathbf{Sm} and \mathbf{CPL} (over the same signature) and $r = \langle \{\neg\neg\xi_1\}, \xi_1 \rangle$. Note that \mathbf{I}^1 can be choose even for non weak selection functions.*

6 Conclusion and Future Work

In this paper it was presented a formal framework for defining new logics by contracting a derived rule from a known logic. The framework consists in defining operations $-$ over possibly closed Hilbert Calculi. These operation are related with contraction operation in belief revision field. Constructions for the operations as well as postulates that characterize them were presented.

This framework is related with the study of maximal logics w.r.t. another logic (cf. [Set73, SC95]) and the study of maximal logics w.r.t. a principle (cf. [AAZ10]). Furthermore, the relation between the notion of strong remainder set and weak remainder set was analyzed. As an application it was proved that the logics \mathbf{P}^1 and \mathbf{I}^1 are strongly maximal w.r.t. Classical Propositional Logic.

This paper focused in the contraction of *one* rule from a logic. It would be interesting to analyze the result of contracting a set of rules. Once again the field of belief revision may help us in this task.

As future work we intend to extend this framework to sequent calculi and also to study the analogous of a revision operation over logics.

Acknowledgements. This research was financed by FAPESP (Brazil), Thematic Project LogCons 2010/51038-0. The first author was also supported by a Post-doctoral grant by FAPESP, process 2011/01384-1. The second author was also financed by an individual research grant from The National Council for Scientific and Technological Development (CNPq), Brazil, process 305237/2011-0.

References

- [AAZ10] Arieli, O., Avron, A., Zamansky, A.: Maximally paraconsistent three-valued logics. In: Proceedings of the 12th International Conference on the Principles of Knowledge Representation and Reasoning (KR 2010), Toronto, Canada (2010)
- [AGM85] Alchourrón, C., Gärdenfors, P., Makinson, D.: On the logic of theory change. *Journal of Symbolic Logic* 50(2), 510–530 (1985)
- [AM81] Alchourrón, C., Makinson, D.: Hierarchies of regulation and their logic. In: Hilpinen (ed.) *New Studies in Deontic Logic*, pp. 125–148. D. Reidel Publishing Company (1981)

- [CZ97] Chagrov, A., Zakharyashev, M.: Modal Logic. Oxford Logic Guides. Clarendon Press (1997)
- [DdC70] D'Ottaviano, I.M.L., da Costa, N.C.A.: Sur un problème de Jaśkowski. Comptes Rendus de l'Académie de Sciences de Paris 270, 1349–1353 (1970)
- [Han91] Hansson, S.O.: Belief contraction without recovery. *Studia Logica* 50(2), 251–260 (1991)
- [Han99] Hansson, S.O.: A Textbook of Belief Dynamics. Kluwer Academic (1999)
- [Rib12] Ribeiro, M.M.: Belief Revision in Non-Classical Logics. Springer Briefs in Computer Science, vol. XII. Springer (2012)
- [SC95] Sette, A.M., Carnielli, W.A.: Maximal weakly-intuitionistic logics. *Studia Logica* 55(1), 181–203 (1995)
- [Set73] Sette, A.M.: On the propositional calculus P^1 . *Mathematica Japonicae* 18, 173–180 (1973)
- [Wój88] Wójcicki, R.: Theory of logical calculi: basic theory of consequence operations. Synthese library. Kluwer Academic Publishers (1988)

Appendix A: Proofs of the Main Results

Lemma 6: **Proof.** Let r be a rule over C such that $r \in Cl(\emptyset)$. If $H \not\subseteq H - r$ then there is $r' \in R_H \setminus R_{H-r}$. By relevance, there is H' such that $r \notin Cl(H')$. But this is an absurd, by monotonicity of Cl . ■

Lemma 9: **Proof.** Let $H = Cl(H)$, and $X \in H \perp R$. By extensiveness it follows that $X \subseteq Cl(X)$. Now, suppose that $X \subset Cl(X)$. By monotonicity $Cl(X) \subseteq Cl(H) = H$, and so $X \subset Cl(X) \subseteq H$. Since $X \in H \perp R$ we have, by idempotence, that $R \subseteq Cl(R_X)$, which is a contradiction. Hence, there is no $r \in Cl(X) \setminus X$ i.e. $X = Cl(X)$. ■

Proposition 10: **Proof.** To prove that $X \in H \perp R_2$ we need to show 1) that $X \subseteq H$, 2) $R_2 \not\subseteq Cl(R_X)$ and 3) if $X \subset X' \subseteq H$ then $R_2 \subseteq Cl(R_{X'})$. 1) follows directly from the fact that $X \in H \perp R_1$ and 2) follows by hypothesis. To prove 3) notice that if $X \subset X' \subseteq H$ then $R_1 \subseteq Cl(R_{X'})$ and, since $R_2 \subseteq R_1$, $R_2 \subseteq Cl(X')$. ■

Proposition 14: **Proof.** We need to prove 1) that $X \subseteq H_2$, 2) $R \not\subseteq Cl(R_X)$ and 3) if $X \subset X' \subseteq H_2$ then $R \subseteq Cl(R_{X'})$. 1) and 2) follow directly from hypothesis. Now let consider X' such that $X \subset X' \subseteq H_2$. We have by hypothesis that $H_2 \subseteq H_1$ and, hence, $X' \subseteq H_1$. It follows that $R \subseteq Cl(R_{X'})$, because $X \in H_1 \perp R$. ■

Lemma 16: **Proof.** For $i = 1, \dots, n$ it holds that $(\xi \rightarrow \gamma_i), \xi \vdash_{Hr} \gamma_i$, by (1), and so $(\xi \rightarrow \gamma_1), \dots, (\xi \rightarrow \gamma_n), \xi \vdash_{Hr} \gamma_i$, for every i . Since $\gamma_1, \dots, \gamma_n \vdash_{Hr} \varphi$ then $(\xi \rightarrow \gamma_1), \dots, (\xi \rightarrow \gamma_n), \xi \vdash_{Hr} \varphi$. But then $(\xi \rightarrow \gamma_1), \dots, (\xi \rightarrow \gamma_n), \xi \vdash_{Hr} (\xi \rightarrow \varphi)$, since $\vdash_{Hr} \varphi \rightarrow (\xi \rightarrow \varphi)$ and by (1). On the other hand $\sim \xi \vdash_{Hr} (\xi \rightarrow \varphi)$, by (3), and so $(\xi \rightarrow \gamma_1), \dots, (\xi \rightarrow \gamma_n), \sim \xi \vdash_{Hr} (\xi \rightarrow \varphi)$. By (4) it follows that $(\xi \rightarrow \gamma_1), \dots, (\xi \rightarrow \gamma_n) \vdash_{Hr} (\xi \rightarrow \varphi)$ as required. ■

³ Lemmas 6 and 9 were adapted from [Han99].

Corollary 17: Proof. Assume that $\Gamma, \alpha \vdash_{H^r} \beta$. By induction on the length of a derivation $\varphi_1 \dots \varphi_k$ of β from $\Gamma \cup \{\alpha\}$ in H^r it will be shown that $\Gamma \vdash_{H^r} (\alpha \rightarrow \beta)$. Since H satisfies MTD with respect to \rightarrow , the only case to be analyzed is when β is obtained by the use of the rule $r = \langle \{\gamma_1, \dots, \gamma_n\}, \varphi \rangle$. Thus, there is some substitution σ such that $\beta = \hat{\sigma}(\varphi)$ and $\{\hat{\sigma}(\gamma_1), \dots, \hat{\sigma}(\gamma_n)\} \subseteq \{\varphi_1, \dots, \varphi_{k-1}\}$. By induction hypothesis, $\Gamma \vdash_{H^r} (\alpha \rightarrow \hat{\sigma}(\gamma_i))$ for every i . By Lemma 16 $(\alpha \rightarrow \hat{\sigma}(\gamma_1)), \dots, (\alpha \rightarrow \hat{\sigma}(\gamma_n)) \vdash_{H^r} (\alpha \rightarrow \hat{\sigma}(\varphi))$ (by taking $\sigma(\xi) = \alpha$). Therefore $\Gamma \vdash_{H^r} (\alpha \rightarrow \hat{\sigma}(\varphi))$, that is, $\Gamma \vdash_{H^r} (\alpha \rightarrow \beta)$. ■

Corollary 18: Proof. Clearly $\gamma_1, \dots, \gamma_n \vdash_{H^r} \varphi$, and then $\vdash_{H^r} \gamma_1 \rightarrow (\gamma_2 \rightarrow (\dots \rightarrow (\gamma_n \rightarrow \varphi) \dots))$, by MTD. Thus $Cl(H_r) \subseteq Cl(H^r)$.

On the other hand, by (i) of Lemma 16 it holds that $\gamma_1, \dots, \gamma_n \vdash_{H^r} \varphi$ and so $Cl(H^r) \subseteq Cl(H_r)$. This completes the proof. ■

Theorem 19: Proof. Assume that H and H' satisfy the hypothesis of the theorem. The ‘if’ part is obviously true. For the ‘only if’ part, assume that $H \in H' \perp_w R$ and let $r = \langle \{\gamma_1, \dots, \gamma_n\}, \varphi \rangle$ be a rule such that $r \in R_{H'} \setminus R_H$. Since H' satisfies MTD with respect to \rightarrow it follows that $\vdash_{H'} \gamma_1 \rightarrow (\gamma_2 \rightarrow (\dots \rightarrow (\gamma_n \rightarrow \varphi) \dots))$. Since H satisfies (1) of Lemma 16 it follows that $\not\vdash_H \gamma_1 \rightarrow (\gamma_2 \rightarrow (\dots \rightarrow (\gamma_n \rightarrow \varphi) \dots))$. Given that $H \in H' \perp_w R$ it follows that $R \not\subseteq Cl(R_H)$ but $R \subseteq Cl(H_r)$, where $H_r = \langle C, R_H \cup \{\langle \emptyset, \gamma_1 \rightarrow (\gamma_2 \rightarrow (\dots \rightarrow (\gamma_n \rightarrow \varphi) \dots)) \rangle\} \rangle$. But $Cl(H_r) = Cl(H^r)$, by Corollary 18, where $H^r = \langle C, R_H \cup \{r\} \rangle$. Then $R \subseteq Cl(H^r)$ and so $H \in H' \perp R$. ■

Proposition 20: Proof. 1. It is known that the logic \mathbf{I}^1 is maximal with respect to \mathbf{CPL} in the signature just containing \rightarrow and \neg (cf. [SC95]). That is, $\mathbf{I}^1 \in \mathbf{CPL} \perp_w R_{\mathbf{CPL}}$. Let $\sim \alpha =_{def} (\alpha \rightarrow \neg \alpha)$ and $\alpha \vee \beta =_{def} (\neg(\beta \rightarrow \alpha) \rightarrow \beta) \rightarrow ((\alpha \rightarrow \alpha) \rightarrow \alpha)$. Then $\vdash_{\mathbf{I}^1} (\sim \alpha \vee \alpha)$ for every α . On the other hand, it is easy to show the following: $\Gamma, \alpha \vdash_{(\mathbf{I}^1)^r} \gamma$ and $\Gamma, \beta \vdash_{(\mathbf{I}^1)^r} \gamma$ implies that $\Gamma, (\alpha \vee \beta) \vdash_{(\mathbf{I}^1)^r} \gamma$, for every $\Gamma, \alpha, \beta, \gamma$ and for every rule r , where $(\mathbf{I}^1)^r$ is as in Lemma 16. Thus, \mathbf{I}^1 satisfies the conditions (1)-(4) of Lemma 16, for every rule r . On the other hand, both \mathbf{CPL} and \mathbf{I}^1 satisfy MTD with respect to \rightarrow . Then, by Theorem 19 it follows that \mathbf{I}^1 is strongly maximal with respect to \mathbf{CPL} , that is, $\mathbf{I}^1 \in \mathbf{CPL} \perp R_{\mathbf{CPL}}$.

2. We know from Corollary 12 that $\mathbf{I}^1 \in \mathbf{CPL} \perp_w \{\langle \{\neg \xi_1\}, \xi_1 \rangle\}$. The proof that $\mathbf{I}^1 \in \mathbf{CPL} \perp \{\langle \{\neg \xi_1\}, \xi_1 \rangle\}$ is analogous to that of item 1. ■

Proposition 21: Proof. The proof is similar to that of Proposition 20.

1. We begin by observing that the logic \mathbf{P}^1 is maximal with respect to \mathbf{CPL} in the signature just containing \rightarrow and \neg (cf. [Set73]). That is, $\mathbf{P}^1 \in \mathbf{CPL} \perp_w R_{\mathbf{CPL}}$. It is enough to define in \mathbf{P}^1 a classical negation \sim and a disjunction \vee which guarantee, as in the case of \mathbf{I}^1 , the satisfaction of conditions (1)-(4) of Lemma 16, for every rule r . The derived connectives $\sim \alpha =_{def} \neg(\alpha \rightarrow \alpha)$ and $\alpha \vee \beta =_{def} (\sim \alpha \rightarrow \beta)$ ($\sim \alpha \rightarrow \beta$) satisfy the required properties. Since both \mathbf{P}^1 and \mathbf{CPL} satisfy MTD

with respect to \rightarrow then, by Theorem 19, it follows that \mathbf{P}^1 is strongly maximal with respect to \mathbf{CPL} . That is, $\mathbf{P}^1 \in \mathbf{CPL} \perp R_{\mathbf{CPL}}$.

2. We know from Corollary 11 that $\mathbf{P}^1 \in \mathbf{CPL} \perp_w \{\langle \{\xi_1, \neg \xi_1\}, \xi_2 \rangle\}$. The proof that $\mathbf{P}^1 \in \mathbf{CPL} \perp \{\langle \{\xi_1, \neg \xi_1\}, \xi_2 \rangle\}$ is analogous to that of item 1. ■

Lemma 22. Proof. Let C be the signature of H and enumerate the rules in $R_H: \{r_1, r_2, \dots\}$. Let $R_0 = R_X$ and for each $i \geq 1$ let:

$$R_i = \begin{cases} R_{i-1} \cup \{r_i\} & \text{if } R \not\subseteq Cl(R_{i-1} \cup \{r_i\}) \\ R_{i-1} & \text{otherwise.} \end{cases}$$

Now consider $H' = \langle C, \bigcup_i R_i \rangle$. Clearly, $X \subseteq H'$. Suppose that $R = \{r'_1, \dots, r'_n\}$ is contained in $Cl(R_{H'})$. Since Cl is compact, there exists a finite set $R'_j \subseteq R_{H'}$ such that $r'_j \in Cl(R'_j)$ for $j = 1, \dots, n$. But $R_i \subseteq R_{i+1}$ and so $R \subseteq Cl(R_j)$ for some j , which is a contradiction. We conclude that $R \not\subseteq Cl(R_{H'})$.

The other conditions for $H' \in H \perp R$ are easy to verify. ■

Theorem 23. Proof. *Inclusion* follows trivially and *success* follows directly from the upper-bound lemma with $X = \emptyset$. To prove *relevance* note that if $r' \in R_H \setminus \bigcap \mathcal{Y}(H, r)$ then there is $X \in \mathcal{Y}(H, r)$ such that $r' \notin X$. Of course $\bigcap \mathcal{Y}(H, r) \subseteq X \subseteq H$, $r \notin Cl(X)$ and, since X is maximal, then $r \in Cl(R_X \cup \{r'\})$. This same argument holds if r' is an axiom and \mathcal{Y} is a weak subset selection function, hence *weak relevance* also holds. Finally, if H is closed then *closure* follows directly from Lemma 9. ■

Theorem 24. Proof. We will show only a sketch of a proof for strong subset selection function. The proof for weak subset selection function is completely analogous. Let $\mathcal{Y}(H, r) = \{X \in H \perp \{r\} : H - r \subseteq X\}$ if $H \perp \{r\} \neq \emptyset$ and $\mathcal{Y}(H, r) = \{H\}$ otherwise. We need to show that \mathcal{Y} is well defined, that it is a selection function and that $H - r = \bigcap \mathcal{Y}(H, r)$.

Proving that \mathcal{Y} is well defined is trivial, since we defined \mathcal{Y} over the generators (that is, the pairs $\langle H, r \rangle$).

It is also trivial to verify that $\mathcal{Y}(H, r) \subseteq H \perp \{r\}$. From *success*, *inclusion* and the upper-bound lemma, we show that $\mathcal{Y}(H, r) \neq \emptyset$.

Now if $H \perp \{r\} = \emptyset$ then $r \in Cl(\emptyset)$. In this case $\bigcap \mathcal{Y}(H, r) = H$ and, by *failure* and *inclusion* we have that $H - r = H$. If $H \perp \{r\} \neq \emptyset$ then trivially $H - r \subseteq \bigcap \mathcal{Y}(H, r)$. To prove the converse, suppose by absurdum that $r' \notin H - r$. If $r' \notin H$ then $r' \notin \bigcap \mathcal{Y}(H, r)$ and we are done. Consider then that $r' \in H$. Then by *relevance* there is H' such that $H - r \subseteq H' \subseteq H$, $r \notin Cl(H')$ and $r \in Cl(R_{H'} \cup \{r'\})$. By the upper bound lemma there is X such that $H' \subseteq X \in H \perp \{r\}$. It follows that $r' \notin X \in \mathcal{Y}(H, r)$ and, hence, $r' \notin \bigcap \mathcal{Y}(H, r)$. ■

⁴ This proof was adapted from a very similar to the proof of Lindenbaum's lemma found in Wójc88.

Theorem 25: **Proof.** We will sketch the proof of maxi-choice characterization. The proof for weak maxi-choice characterization is completely analogous.

Let $\Psi(H, r) = H - r$. We must prove that $\Psi(H, r) \in H \perp \{r\}$ if $H \perp \{r\} \neq \emptyset$ and $\Psi(H, r) = H$ otherwise.

The second situation follows directly from *failure*. Now let us assume that $r \notin Cl(\emptyset)$, then $\Psi(H, r) \in H \perp \{r\}$ by *success*, *inclusion* and *fullness*. ■

A Tight Upper Bound on the Number of Variables for Average-Case k -Clique on Ordered Graphs

Benjamin Rossman

Tokyo Institute of Technology
2-12-1 Ookayama, Meguro-ku, Tokyo, 152-8550, Japan

Abstract. A first-order sentence φ defines k -clique in the average-case if

$$\lim_{n \rightarrow \infty} \Pr_{G=G(n,p)} [G \models \varphi \Leftrightarrow G \text{ contains a } k\text{-clique}] = 1$$

where $G = G(n, p)$ is the Erdős-Rényi random graph with $p (= p(n))$ the exact threshold such that $\Pr[G(n, p) \text{ has a } k\text{-clique}] = 1/2$. We are interested in the question: *How many variables are required to define average-case k -clique in first-order logic?* Here we consider first-order logic in vocabularies which, in addition to the adjacency relation of G , may include fixed “background” relations on the vertex set $\{1, \dots, n\}$ (for example, linear order). Some previous results on this question:

- With no background relations, $k/2$ variables are necessary and $k/2 + O(1)$ variables are sufficient (Ch. 6 of [7]).
- With arbitrary background relations, $k/4$ variables are necessary [6].
- With arithmetic background relations ($<, +, \times$), $k/4 + O(1)$ variables are sufficient (Amano [1]).

In this paper, we tie up a loose end (matching the lower bound of [6] and improving the upper bound of [1]) by showing that $k/4 + O(1)$ variables are sufficient with only a linear order in the background.

1 Introduction

The *number of variables* in a first-order formula φ refers to the number of distinct variable symbols (x, y, z , etc.) occurring in φ . This number includes both free and bound variables, and we allow variables to be quantified multiple times. For example, the following 2-variable sentence¹ expresses “the universe has ≥ 5 elements” on the class of linear orders:

$$\exists x \exists y (x < y \wedge \exists x (y < x \wedge \exists y (x < y \wedge \exists x (y < x))))).$$

The number of variables is an important measure of the complexity of a first-order formula. Under a well-known descriptive complexity characterization of

¹ Recall that a *sentence* is a formula with no free variables.

first-order logic in terms of the complexity class AC^0 , every s -variable formula has an equivalent AC^0 circuit of size $O(n^s)$ [3].

A well-studied question in model theory and finite model theory is: over which classes of structures does first-order logic increase in express power with respect to the number of variables? That is, when is the so-called *variable hierarchy* strict? For instance, 2 variables are enough to express every first-order property over the class of finite linear orders, whereas 3 variables are enough over the class of finite words [5]. On the other hand, the variable hierarchy is strict on the class of finite graphs. A longstanding open question was whether the variable hierarchy is strict on the class of finite ordered graphs (see [2]). Answering this question, in [6] we showed that the property “there exists a k -clique” requires $k/4$ variables on the class of finite ordered graphs. This lower bound is in fact an average-case hardness result: in the first-order language of ordered graphs, $k/4$ variables are required even to express “there exists a k -clique” with high probability on a certain natural distribution (the Erdos-Renyi random graph $G(n, p)$ for $p = p(n)$ an appropriate threshold).

Following [6], Kazayuki Amano [1] gave uniform AC^0 circuits of size $n^{k/4+O(1)}$ which define k -clique in the same average-case sense. Under the descriptive complexity characterization of uniformity, Amano’s circuits are equivalent to a sentence in the first-order language of graphs on $\{1, \dots, n\}$ with arithmetic background relations $<$, $+$ and \times . In the author’s Ph.D. thesis [7], it was noted that the “ $k/2$ -extension axiom” (famous from the 0-1 law for first-order logic) implies a lower bound of $k/2$ variables for the average-case definability of k -clique in the absence of background relation; together with Joel Spencer, an upper bound of $k/2 + O(1)$ was also shown. One question left open from all this work is whether $k/4 + O(1)$ or $k/2 + O(1)$ (or something in-between) is the true number of variables required to define k -clique in the average-case with only a linear order in the background. Tying up this loose end, in this paper we show that $k/4 + O(1)$ variables suffice.

2 Preliminaries

Let k be a fixed constant (independent of n). Let $p = p(n) = n^{-2/(k-1)}$, although everything we write holds for any $p(n) = \Theta(n^{-2/(k-1)})$ including the exact threshold for k -clique (see any standard text such as [4] for background on random graphs). Let G be the Erdős-Rényi random graph $G(n, p)$, viewed as a linearly ordered graph. That is, G is random structure with universe $[n] = \{1, \dots, n\}$ and binary relations E and $<$ where $<$ is the standard linear on $[n]$ and E is an anti-reflexive symmetric binary relation such that events $\{(u, v) \in E\}$ occur independently with probability p over pairs (u, v) such that $1 \leq u < v \leq n$. Throughout this note, “almost surely” means “with probability tending to 1 as $n \rightarrow \infty$ ”. For vertices $u, v \in [n]$ such that $u \leq v$, we denote by $[u, v]$ the interval of vertices including and between u and v .

In this paper, we prove the following:

Theorem 1. *There is a sentence φ in the first-order language of ordered graphs with only $k/4 + O(1)$ variables such that, almost surely, $G \models \varphi$ if and only if G has a k -clique.*

To prove Theorem 1 we first define a property \mathcal{P} of finite ordered graphs (Definition 4) such that \mathcal{P} implies the existence of a k -clique. We then show that \mathcal{P} is first-order definable with $k/4 + O(1)$ variables (Lemma 1). Finally, we show that almost surely, if G has a k -clique then G has property \mathcal{P} (Lemma 8).

For simplicity, we treat the case where $k \geq 7$ and $k \equiv 3 \pmod 4$. The proof holds with minor modifications when $k \not\equiv 3 \pmod 4$. Let $t = (k - 1)/2$ and $s = (k - 7)/4$. Note that $s \geq 0$ and $t = 2s + 3$ are integers and $p = n^{-1/t}$.

3 Proof Sketch

Before defining property \mathcal{P} in the next section, we give some basic intuition. We start by showing how to define k -CLIQUE almost surely with $k/2 + O(1)$ variables. Suppose that G contains a k -clique $\{v_1, \dots, v_k\}$ (i.e. condition on this event). Then almost surely vertices v_{t+2}, \dots, v_k are the only common neighbors of v_1, \dots, v_{t+1} . This is seen by the following union bound:

$$\begin{aligned} & \Pr[v_1, \dots, v_{t+1} \text{ have a common neighbor beside } v_{t+2}, \dots, v_k] \\ & \leq \sum_{w \in [n] \setminus \{v_1, \dots, v_k\}} \Pr[w \text{ is a common neighbor of } v_1, \dots, v_{t+1}] \\ & = (n - k)p^{t+1} < p = o(1). \end{aligned}$$

Denote by \mathcal{Q} the following property: *there exist distinct vertices x_1, \dots, x_{t+1} such that x_1, \dots, x_{t+1} form a clique and have $\geq t$ common neighbors and every two common neighbors of x_1, \dots, x_{t+1} are adjacent.* Note that property \mathcal{Q} implies the existence of a k -clique (as $k = 2t + 1$). The above inequality also shows that, almost surely, if G has a k -clique then G has property \mathcal{Q} ; hence \mathcal{Q} is almost surely equivalent to k -CLIQUE with respect to the random graph G .

We claim that \mathcal{Q} is definable with only $t + 3 = k/2 + O(1)$ variables on the class of finite ordered graphs. (Here the linear order is indispensable: \mathcal{Q} is not definable with fewer than k variables on the class of finite graphs.) The key observation is that saying “ x_1, \dots, x_{t+1} have $\geq t$ common neighbors” can be achieved with only 2 bound variables in addition to free variables x_1, \dots, x_{t+1} : letting $\nu(\vec{x}, y) \equiv \bigwedge_{i \in \{1, \dots, t+1\}} \text{Edge}(x_i, y)$, we have

$$\begin{aligned} & \text{“}x_1, \dots, x_{t+1} \text{ have } \geq t \text{ common neighbors”} \equiv \\ & \exists y, \nu(\vec{x}y) \wedge (\exists z, y < z \wedge \nu(\vec{x}z) \wedge (\exists y, z < y \wedge \nu(\vec{x}y) \wedge (\exists z, z < y \wedge \nu(\vec{x}z) \wedge \dots))) \end{aligned}$$

where there are t existential quantifiers in total. Hence, property \mathcal{Q} can be expressed with $t + 3$ variables as follows:

$$\begin{aligned} \mathcal{Q} \equiv & \exists x_1 \dots \exists x_{t+1}, \bigwedge_{1 \leq i < j \leq t+1} \text{Edge}(x_i, x_j) \\ & \wedge \text{“}x_1, \dots, x_{t+1} \text{ have } \geq t \text{ common neighbors”} \\ & \wedge \forall y \forall z, (\nu(\vec{x}, y) \wedge \nu(\vec{x}, z) \wedge y \neq z) \rightarrow \text{Edge}(y, z). \end{aligned}$$

Property \mathcal{P} is similar to property \mathcal{Q} , except that we must use $k/4+O(1)$ variables to *isolate* the $k/2 + O(1)$ vertices x_1, \dots, x_{t+1} that make up the first half of a possible k -clique in the graph G . (As with property \mathcal{Q} , once we isolate these $t + 1$ vertices, it will be easy to say that they belong to a k -clique using just $O(1)$ additional free variables.) What do we mean by *isolate*? Well, with only $k/4+O(1)$ parameters, there is no hope of defining the set $\{x_1, \dots, x_{t+1}\}$ exactly. But we can define a sequence of *intervals* $I_1, \dots, I_{t+1} \subseteq [n]$ where I_j contains x_j and is not too large; in fact, I_j has size roughly $n^{j/t}$. This sequence will *isolate* x_1, \dots, x_{t+1} in the sense that for all $j \in \{1, \dots, t\}$, x_{j+1} is the unique common neighbor of x_1, \dots, x_j in the interval I_j . This property allows us to efficiently define x_j (with $O(1)$ extra variables) given formulas defining I_1, \dots, I_{t+1} . As to defining intervals I_1, \dots, I_{t+1} using just $k/4+O(1)$ variables, this is accomplished by using a single variable for each of I_1, \dots, I_4 and a single variable for each pair $(I_5, I_{t+1}), (I_6, I_t), (I_7, I_{t-1}), \dots, (I_{s+4}, I_{s+5})$; that is, $s + 4 = k/4 + O(1)$ total variables.

4 Property \mathcal{P}

The following definitions refer to a fixed but arbitrary finite ordered graph. Without loss of generality, we assume this graph has vertex set $[n]$ under the standard ordering. For a vertex $v \in [n]$, we denote by $v+1$ and $v-1$ the successor and predecessor of v (when defined).

Definition 1. A sequence I_1, \dots, I_ℓ of subsets of $[n]$ is an ℓ -clique isolator if $|I_1| = 1$ and there exists $(u_1, \dots, u_\ell) \in I_1 \times \dots \times I_\ell$ such that for every $i \in \{2, \dots, \ell\}$, u_i is the unique common neighbor of u_1, \dots, u_{i-1} in the set I_i .

Remark 1. The notion of an ℓ -clique isolator will be useful for the following reason. Suppose I_1, \dots, I_ℓ are given by unary relation symbols. Then the statement “ I_1, \dots, I_ℓ is an ℓ -clique isolator” can be expressed in first-order logic using only 2 variables. To see this, we inductively define formulas $\psi_i(x)$ such that $\psi_i(x)$ is true iff I_1, \dots, I_i is an i -clique isolator and $x = u_i$ (i.e., for the unique i -clique $\{u_1, \dots, u_i\}$ with $(u_1, \dots, u_i) \in I_1 \times \dots \times I_i$). In the base case,

$$\psi_1(x) \equiv I_1(x) \wedge (\forall y, y \neq x \rightarrow \neg I_1(y)).$$

For $i \in \{2, \dots, \ell\}$, define

$$\begin{aligned} \psi_i(x) &\equiv \theta_i(x) \wedge (\forall y, y \neq x \rightarrow \neg \theta_i(y)) \\ \text{where } \theta_i(x) &\equiv I_i(x) \wedge \bigwedge_{j \in \{1, \dots, i-1\}} (\exists y, \psi_j(y) \wedge \text{Edge}(x, y)). \end{aligned}$$

The statement “ I_1, \dots, I_ℓ is an ℓ -clique isolator” is equivalent to the 2-variable formula $\exists x, \psi_\ell(x)$. A corollary of this observation is that if each set I_i is definable by an m -variable formula, then the statement “ I_1, \dots, I_ℓ is an ℓ -clique isolator” is equivalent to a formula with $m + 2$ variables.

Definition 2. A vertex $v \in [n]$ is a pointer if $v \geq t + 1$ and $v, v - 1, \dots, v - t$ (i.e., v and its t predecessors) have a unique common neighbor. If v is a pointer, we denote by v^* the unique common neighbor of $v, v - 1, \dots, v - t$.

Remark 2. The predicate “ x is a pointer and $x^* = y$ ” is definable with 3 variables (i.e., 1 variable in addition to x and y).

$$\begin{aligned}
 & \text{“}x \text{ is a pointer and } x^* = y\text{”} \\
 & \equiv \text{“}x \text{ has } \geq t \text{ predecessors”} \wedge \gamma(x, y) \wedge \left(\forall z, z \neq y \rightarrow \neg \gamma(x, z) \right) \\
 & \text{where } \gamma(x, y) \equiv \bigwedge_{j \in \{0, \dots, t\}} \left(\exists z, \text{“}z = x - j\text{”} \wedge \text{Edge}(y, z) \right).
 \end{aligned}$$

We leave it as an exercise to show that “ x has $\geq t$ predecessors” is definable with 1 variable in addition to x and “ $z = x - j$ ” (for fixed j) is definable with 1 variable in addition to x and z .

Remark 3. For any $v \in [n]$ such that $v \geq t + 1$, the probability that v is a pointer in G is roughly p . Conditioning on v being a pointer, v^* is uniformly distributed in $[n] \setminus \{v, v - 1, \dots, v - t\}$. (These facts come up in the proof of Lemma 3.)

Definition 3. For $j \geq 1$ and $v \in [n]$, denote by $f_j(v)$ the minimal $w \in [n]$ such that $w > v$ and w is a common neighbor of $v + 1, \dots, v + j$ (i.e., the j successors of v); in cases where $f_j(v)$ would be undefined (either because $v > n - j$ or because $v + 1, \dots, v + j$ have no common neighbor greater than v), we set $f_j(v) = n$.

Remark 4. For fixed $j \geq 1$, the predicate “ $f_j(x) = y$ ” is definable with 3 variables (cf. Remark 2).

Remark 5. For any $j \in \{1, \dots, t - 1\}$ and $v \in [n]$ such that $v < n - n^{1-\varepsilon}$, we expect $f_j(v)$ to be around $v + p^{-j} = v + n^{j/t}$ in the random graph G . Indeed, for any constant $\varepsilon > 0$, it holds almost surely that $v + n^{(j/t)-\varepsilon} < f_j(v) < v + n^{(j/t)+\varepsilon}$. Moreover, this is true even if we condition on arbitrary events in G depending only on edges outside of the interval $[v + n^{(j/t)-\varepsilon}, v + n^{(j/t)+\varepsilon}]$.

Definition 4. A finite ordered graph has property \mathcal{P} if there exist vertices v_1, v_2, v_3, v_4 and w_1, \dots, w_s such that

- (i) w_1, \dots, w_s are pointers and
- (ii) the following sequence of subsets of $[n]$ is a $(t + 1)$ -clique isolator:

$$\{v_1\}, \{v_2\}, \{v_3\}, \{v_4\}, \underbrace{[w_1, f_3(w_1)], \dots, [w_s, f_{s+2}(w_s)]}_{[w_i, f_{i+2}(w_i)] \text{ for } i=1, \dots, s}, \underbrace{[w_s^*, f_{t-s}(w_s^*)], \dots, [w_1^*, f_{t-1}(w_1^*)]}_{[w_i^*, f_{t-i}(w_i^*)] \text{ for } i=s, \dots, 1}$$
- (iii) for the unique $(t + 1)$ -clique $\{v_1, \dots, v_{t+1}\}$ isolated by this sequence, v_1, \dots, v_{t+1} have exactly t common neighbors and these common neighbors form a t -clique.

Lemma 1. *There is a formula with $k/4 + O(1)$ variables that defines property \mathcal{P} on the class of finite ordered graphs.*

Proof. The formula defining \mathcal{P} begins with $\exists v_1, v_2, v_3, v_4, w_1, \dots, w_s$. Each set $[w_i, f_{i+2}(w_i)]$ and $[w_i^*, f_{t-i}(w_i^*)]$ is definable with $C = O(1)$ variables in addition to parameter w_i (cf. Remarks [2](#) and [4](#)). Therefore, the statement that

$$\{v_1\}, \{v_2\}, \{v_3\}, \{v_4\}, [w_1, f_3(w_1)], \dots, [w_s, f_{s+2}(w_s)], [w_s^*, f_{t-s}(w_s^*)], \dots, [w_1^*, f_{t-1}(w_1^*)]$$

is a $(t + 1)$ -clique isolator can be expressed with only $C + 2$ variables in addition to parameters $v_1, v_2, v_3, v_4, w_1, \dots, w_s$; moreover, the individual elements v_1, \dots, v_{t+1} of the unique $(t + 1)$ -clique isolated by this sequence are definable with the same $s + O(1)$ total variables (cf. Remark [11](#)). Using the order, we can express that v_1, \dots, v_{t+1} have exactly t common neighbors with only 3 additional variables. To express that these common neighbors form a k -clique, we can say any two common neighbors are adjacent, using just 2 additional variables. So in total we require $s + O(1) = k/4 + O(1)$ variables (in fact, $k/4 + 10$ are sufficient).

5 Almost Surely, G Has a k -Clique Iff G Has Property \mathcal{P}

Let $\varepsilon > 0$ be a sufficiently small constant ($\varepsilon = 1/k$ will do).

Definition 5. *A tuple (u_1, \dots, u_ℓ) of vertices in $[n]$ is well-spaced if*

$$n^{1-\varepsilon} < u_1 < \dots < u_\ell < n - n^{1-\varepsilon}$$

and $u_{i+1} - u_i > n^{1-\varepsilon}$ for $i \in \{1, \dots, \ell - 1\}$.

Lemma 2. *Almost surely, if G contains a k -clique, then G contains a well-spaced k -clique.*

Proof. Condition on G containing a k -clique. Sample $\{v_1, \dots, v_k\}$ uniformly from among the k -cliques of G where $v_1 < \dots < v_k$. Notice that (v_1, \dots, v_k) is uniformly distributed among increasing k -tuples in $[n]^k$. The lemma follows from the observation that a uniform random increasing k -tuple in $[n]^k$ is well-spaced with high probability.

Lemma 3. *Let $u, u' \in [n]$ be a fixed well-spaced pair of vertices and let $i \in \{1, \dots, s\}$. Almost surely in G , there is a vertex w such that*

$$\begin{aligned} u - n^{\frac{i+2}{t}-\varepsilon} < w < u < f_{i+2}(w) < u + n^{\frac{i+2}{t}+\varepsilon}, \\ u' - n^{\frac{i-i-1}{t}+3\varepsilon} < w^* < u' < f_{t-i}(w^*) < u' + n^{\frac{i-i}{t}+\varepsilon}. \end{aligned}$$

Proof. Let $M = \{1, \dots, \lceil n^{\frac{i+2}{t}-2\varepsilon} \rceil\}$ and for $m \in M$, let $x_m = u - 2tm$ and denote by Z_m the event that x_m is a pointer and $u' - n^{\frac{i-i-1}{t}+3\varepsilon} < x_m^* < u'$. Note that events Z_m are mutually independent (using the fact that u, u' are well-spaced). We have

$$\Pr[Z_m] \sim n^{-\frac{i+2}{t}+3\varepsilon}.$$

This is obtained from the following inequalities:

- $\Pr[Z_m] = \Pr[x_m \text{ is a pointer}] \Pr[u' - n^{\frac{t-i-1}{t}+3\epsilon} < x_m^* < u' \mid x_m \text{ is a pointer}],$
- $\Pr[x_m \text{ is a pointer}]$
 - $= \Pr[x_m, x_m - 1, \dots, x_m - t \text{ have a unique common neighbor}]$
 - $= \Pr[\geq 1 \text{ common neighbor}] - \Pr[\geq 2 \text{ common neighbors}],$
- $\Pr[\geq 1 \text{ common neighbor}] = 1 - (1 - p^{t+1})^{n-t-1} \sim 1 - \exp(n^{1+(1/t)})^{n-t-1} \sim n^{-1/t},$
- $\Pr[\geq 2 \text{ common neighbors}] \leq \binom{n-t-1}{2} (p^{t+1})^2 < n^{-2/t},$
- $\Pr[u' - n^{\frac{t-i-1}{t}+3\epsilon} < x_m^* < u' \mid x_m \text{ is a pointer}] \sim n^{-\frac{i+1}{t}+3\epsilon}$

since x_m^* is uniformly distributed in $[n] \setminus \{x, x-1, \dots, x-t\}$ conditioned on x_m being a pointer.

By independence of Z_m 's, we have

$$\Pr\left[\bigwedge_{m \in M} \neg Z_m\right] = \prod_{m \in M} \Pr[\neg Z_m] \leq (1 - n^{-\frac{i+2}{t}+3\epsilon} + o(n^{-\frac{i+2}{t}+3\epsilon}))^{n^{\frac{i+2}{t}-2\epsilon}} \sim \exp(n^{-\epsilon}).$$

Therefore, almost surely at least one of the events Z_m holds in $G_{\vec{v}}$.

Now observe the following (cf. Remark 5)

$$\Pr\left[x_m + n^{\frac{i+2}{t}-\epsilon} < f_{i+2}(x_m) < x_m + n^{\frac{i+2}{t}+\epsilon} \mid Z_m\right] = 1 - o(1),$$

$$\Pr\left[x_m^* + n^{\frac{t-i}{t}-\epsilon} < f_{t-i}(x_m^*) < x_m^* + n^{\frac{t-i}{t}+\epsilon} \mid Z_m\right] = 1 - o(1).$$

It follows that for any $m \in M$ such that Z_m holds in $G_{\vec{v}}$, the vertex x_m is almost surely a suitable witness for w .

We now fix an arbitrary well-spaced k -tuple of vertices $\vec{v} = (v_1, \dots, v_k) \in [n]^k$. Denote by $G_{\vec{v}}$ the random graph G conditioned on \vec{v} being a k -clique (that is, $G_{\vec{v}} = G \cup \{k\text{-clique supported on } v_1, \dots, v_k\}$).

Lemma 4. *The following hold almost surely in $G_{\vec{v}}$.*

1. *For all $j \in \{1, \dots, t\}$, v_{j+1} is the unique common neighbor of v_1, \dots, v_j in the interval $[v_{j+1} - n^{\frac{j}{t}-\epsilon}, v_{j+1} + n^{\frac{j}{t}-\epsilon}]$. Hence, the sequence*

$$\{v_1\}, [v_2 - n^{\frac{1}{t}-\epsilon}, v_2 + n^{\frac{1}{t}-\epsilon}], [v_3 - n^{\frac{2}{t}-\epsilon}, v_3 + n^{\frac{2}{t}-\epsilon}], \dots, [v_{t+1} - n^{1-\epsilon}, v_{t+1} + n^{1+\epsilon}]$$

is almost surely a $(t+1)$ -clique isolator in $G_{\vec{v}}$.

2. *v_{t+2}, \dots, v_k are the only common neighbors of v_1, \dots, v_{t+1} .*

Proof. Taking union bounds, we have

1. $\Pr\left[\begin{array}{l} v_1, \dots, v_j \text{ have a common neighbor beside} \\ v_{j+1} \text{ in } [v_{j+1} - n^{\frac{j}{t}-\epsilon}, v_{j+1} + n^{\frac{j}{t}-\epsilon}] \text{ in } G_{\vec{v}} \end{array}\right] \leq 2n^{\frac{j}{t}-\epsilon} p^j = 2n^{-\epsilon} = o(1),$
2. $\Pr\left[\begin{array}{l} v_1, \dots, v_{t+1} \text{ have a common neighbor} \\ \text{beside } v_{t+2}, \dots, v_k \text{ in } G_{\vec{v}} \end{array}\right] \leq (n-k)p^{t+1} < p = o(1).$

For the next two lemmas, it will be convenient to relabel the first $t + 1 (= 2s + 4)$ vertices in \vec{v} as follows. Let

$$v_1, \dots, v_{t+1} = v_1, v_2, v_3, v_4, v'_1, \dots, v'_s, v''_s, \dots, v''_1.$$

That is, $v'_i = v_{i+4}$ and $v''_i = v_{t-i+2}$ for $i \in \{1, \dots, s\}$.

Lemma 5. *Almost surely in $G_{\vec{v}}$, there exist vertices w_1, \dots, w_s such that*

$$\begin{aligned} v'_i - n^{\frac{i+2}{t}-\varepsilon} < w_i < v'_i < f_{i+2}(w_i) < v'_i + n^{\frac{i+2}{t}+\varepsilon}, \\ v''_i - n^{\frac{t-i-1}{t}+3\varepsilon} < w_i^* < v''_i < f_{t-i}(w_i^*) < v''_i + n^{\frac{t-i}{t}+\varepsilon}. \end{aligned}$$

Proof. This is pretty much a corollary of the argument in Lemma 3. Whereas Lemma 3 concerns a single well-separated pair (u, u') in the random graph G , we now consider s well-separated pairs $(v'_1, v''_1), \dots, (v'_s, v''_s)$ in the random graph $G_{\vec{v}}$. However, we can apply the argument in Lemma 3 independently to each pair (v'_i, v''_i) using the fact that $(v'_1, \dots, v'_s, v''_s, \dots, v''_1)$ is well-separated; conditioning on $\{v_1, \dots, v_k\}$ being a clique does not affect the argument.

Lemma 6. *Almost surely in $G_{\vec{v}}$, there exist vertices w_1, \dots, w_s such that*

- $v'_i \in [w_i, f_{i+2}(w_i)]$ and $v''_i \in [w_i^*, f_{t-i}(w_i^*)]$ for all $i \in \{1, \dots, s\}$,
- the sequence

$\{v_1\}, \{v_2\}, \{v_3\}, \{v_4\}, [w_1, f_3(w_1)], \dots, [w_s, f_{s+2}(w_s)], [w_s^*, f_{t-s}(w_s^*)], \dots, [w_1^*, f_{t-1}(w_1^*)]$
is a $(t + 1)$ -clique isolator (and hence isolates the clique $\{v_1, \dots, v_{t+1}\}$).

Proof. Condition on the almost sure properties of $G_{\vec{v}}$ given by Lemma 4(1) and 5. For vertices w_1, \dots, w_s as in Lemma 5, we have $v'_i \in [w_i, f_{i+2}(w_i)]$ and $v''_i \in [w_i^*, f_{t-i}(w_i^*)]$ for all $i \in \{1, \dots, s\}$. The claim that the sequence

$$\{v_1\}, \{v_2\}, \{v_3\}, \{v_4\}, \underbrace{[w_1, f_3(w_1)]}_{\ni v_5}, \dots, \underbrace{[w_s, f_{s+2}(w_s)]}_{\ni v_{s+2}}, \underbrace{[w_s^*, f_{t-s}(w_s^*)]}_{\ni v_{s+3} = v_{t-i+2}}, \dots, \underbrace{[w_1^*, f_{t-1}(w_1^*)]}_{\ni v_{t+1}}$$

is a $(t + 1)$ -clique isolator follows from the fact that is subsumed by the $(t + 1)$ -clique isolator

$$\{v_1\}, [v_2 - n^{\frac{1}{t}-\varepsilon}, v_2 + n^{\frac{1}{t}-\varepsilon}], [v_3 - n^{\frac{2}{t}-\varepsilon}, v_3 + n^{\frac{2}{t}-\varepsilon}], \dots, [v_{t+1} - n^{1-\varepsilon}, v_{t+1} + n^{1-\varepsilon}].$$

That is, we have $\{v_{i_0}\} \subseteq [v_{i_0} - n^{\frac{i_0-1}{t}-\varepsilon}, v_{i_0} + n^{\frac{i_0-1}{t}-\varepsilon}]$ trivially for $i_0 \in \{2, 3, 4\}$, while for $i \in \{1, \dots, s\}$, we have

$$[w_i, f_{i+2}(w_i)] \subseteq [v'_i - n^{\frac{i+2}{t}-\varepsilon}, v'_i + n^{\frac{i+2}{t}+\varepsilon}] \subseteq [v_{i+4} - n^{\frac{i+3}{t}-\varepsilon}, v_{i+4} + n^{\frac{i+3}{t}-\varepsilon}],$$

$$[w_i^*, f_{t-i}(w_i^*)] \subseteq [v''_i - n^{\frac{t-i-1}{t}+3\varepsilon}, v''_i + n^{\frac{t-i}{t}+\varepsilon}] \subseteq [v_{t-i+2} - n^{\frac{t-i+1}{t}-\varepsilon}, v_{t-i+2} + n^{\frac{t-i+1}{t}-\varepsilon}].$$

Lemma 7. *Almost surely, $G_{\vec{v}}$ has property \mathcal{P} .*

Proof. Condition on the almost sure properties of $G_{\vec{v}}$ given by Lemmas 4(2) and 6. Vertices v_1, v_2, v_3, v_4 together with w_1, \dots, w_s from Lemma 6 witness property \mathcal{P} . Lemma 6 takes care of conditions (i) and (ii) in Definition 4, while Lemma 4(2) takes care of condition (iii).

Lemma 8. *Almost surely, G contains a k -clique iff G has property \mathcal{P} .*

Proof. Property \mathcal{P} implies the existence of a k -clique (with probability 1). The other direction follows from Lemmas 2 and 7. Almost surely, if G contains a k -clique then it contains a well-spaced k -clique. But for any well-spaced k -clique $\vec{v} = (v_1, \dots, v_k)$ that we condition on, $G_{\vec{v}}$ has property \mathcal{P} almost surely. Therefore, the existence of a k -clique in G implies that property \mathcal{P} holds almost surely.

References

1. Amano, K.: k -Subgraph isomorphism on AC^0 circuits. *Computational Complexity* 19(2), 183–210 (2010)
2. Dawar, A.: How many first-order variables are needed on finite ordered structures? In: *We Will Show Them: Essays in Honour of Dov Gabbay*, pp. 489–520 (2005)
3. Immerman, N.: *Descriptive Complexity*. Graduate Texts in Computer Science. Springer, New York (1999)
4. Janson, S., Łuczak, T., Ruciński, A.: *Random Graphs*. John Wiley (2000)
5. Poizat, B.: Deux ou trois choses que je sais de L_n . *Journal of Symbolic Logic* 47(3), 641–658 (1982)
6. Rossman, B.: On the constant-depth complexity of k -clique. In: *STOC 2008: Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pp. 721–730 (2008)
7. Rossman, B.: *Average-Case Complexity of Detecting Cliques*. PhD thesis. MIT (2010)

Preservation under Substructures modulo Bounded Cores

Abhisekh Sankaran, Bharat Adsul, Vivek Madan,
Pritish Kamath, and Supratik Chakraborty

Indian Institute of Technology Bombay, India

{abhisekh, adsul, vivekmadan, pritishkamath, supratik}@cse.iitb.ac.in

Abstract. We investigate a model-theoretic property that generalizes the classical notion of preservation under substructures. We call this property *preservation under substructures modulo bounded cores*, and present a syntactic characterization via Σ_2^0 sentences for properties of arbitrary structures definable by FO sentences. Towards a sharper characterization, we conjecture that the count of existential quantifiers in the Σ_2^0 sentence equals the size of the smallest bounded core. We show that this conjecture holds for special fragments of FO and also over special classes of structures. We present a (not FO-definable) class of finite structures for which the conjecture fails, but for which the classical Łoś-Tarski preservation theorem holds. As a fallout of our studies, we obtain combinatorial proofs of the Łoś-Tarski theorem for some of the aforementioned cases.

Keywords: Model theory, First Order logic, Łoś-Tarski preservation theorem.

1 Introduction

Preservation theorems have traditionally been an important area of study in model theory. These theorems provide syntactic characterizations of semantic properties that are preserved under model-theoretic operations. One of the earliest preservation theorems is the Łoś-Tarski theorem, which states that over arbitrary structures, a First Order (FO) sentence is preserved under taking substructures iff it is equivalent to a Π_1^0 sentence [5]. Subsequently many other preservation theorems were studied, e.g. preservation under unions of chains, homomorphisms, direct products, etc. With the advent of finite model theory, the question of whether these theorems hold over finite structures became interesting. It turned out that several preservation theorems fail in the finite [11, 7, 9]. This inspired research on preservation theorems over special classes of finite structures, e.g. those with bounded degree, bounded tree-width etc. These efforts eventually led to some preservation theorems being “recovered” [2, 3]. Among the theorems whose status over the class of all finite structures was open for long was the homomorphism preservation theorem. This was recently resolved in [10], which showed that the theorem survives in the finite.

In this paper, we look at a generalization of the preservation under substructures property that we call *preservation under substructures modulo bounded cores*. In Section 2, we show that for FO sentences, this property has a syntactic characterization in terms of Σ_2^0 sentences over arbitrary structures. Towards a sharper characterization,

we conjecture that for core sizes bounded by a number B , there is a syntactic characterization in terms of Σ_2^0 sentences that use at most B existential quantifiers. In Section 3 we discuss how the notion of *relativization* can be used to prove the conjecture in special cases. We present our studies of the conjecture for special classes of FO and over special classes of structures in Sections 4 and 5. As a fallout of our studies, we obtain combinatorial proofs of the classical Łoś-Tarski theorem for some of the aforesaid special cases, and also obtain semantic characterizations of natural subclasses of the Δ_2^0 fragment of FO. We conclude with questions for future work in Section 6.

We assume that the reader is familiar with standard notation and terminology used in the syntax and semantics of FO (see [8]). A *vocabulary* τ is a set of predicate, function and constant symbols. In this paper, we will restrict ourselves to finite vocabularies only. A *relational vocabulary* has only predicate and constant symbols, and a *purely relational vocabulary* has only predicate symbols. We denote by $FO(\tau)$, the set of all FO formulae over vocabulary τ . A sequence (x_1, \dots, x_k) of variables is denoted by \bar{x} . We will abbreviate a block of quantifiers of the form $Qx_1 \dots Qx_k$ by $Q\bar{x}$, where $Q \in \{\forall, \exists\}$. By Σ_k^0 (resp. Π_k^0), we mean FO sentences in Prenex Normal Form (PNF) over an arbitrary vocabulary, whose quantifier prefix begins with a \exists (resp. \forall) and consists of $k-1$ alternations of quantifiers. We use the standard notions of τ -structures, substructures and extensions, as in [8]. Given τ -structures M and N , we denote by $M \subseteq N$ that M is a substructure of N (or N is an extension of M). Given M and a subset S (resp. a tuple \bar{a} of elements) of its universe, we denote by $M(S)$ (resp. $M(\bar{a})$) the smallest substructure (under set inclusion ordering of the universe) of M containing S (resp. underlying set of \bar{a}) and call it the substructure of M induced by S (resp. underlying set of \bar{a}). Finally, by *size* of M , we mean the cardinality of its universe and denote it by $|M|$. As a final note of convention, whenever we talk of FO definability in the paper, we mean definability via FO sentences (as opposed to theories), unless stated otherwise.

2 Preservation under Substructures Modulo Cores

We denote by $\mathbb{P}\mathcal{S}$ the collection of all classes of structures, in any vocabulary, that are closed under taking substructures. This includes classes that are not definable in any logic. We let PS denote the collection of FO definable classes in $\mathbb{P}\mathcal{S}$. We identify classes in PS with their defining FO sentences and will henceforth treat PS as a set of sentences. We now consider a natural generalization of $\mathbb{P}\mathcal{S}$. Our discussion will concern arbitrary (finite) vocabularies and arbitrary structures over them.

2.1 The Case of Finite Cores

Definition 1 (*Preservation under substructures modulo finite cores*)

A class of structures S is said to be preserved under substructures modulo a finite core (denoted $S \in \mathbb{P}\mathcal{S}\mathcal{C}_f$), if for every structure $M \in S$, there exists a finite subset C of elements of M such that if $M_1 \subseteq M$ and M_1 contains C , then $M_1 \in S$. The set C is called a core of M w.r.t. S . If S is clear from context, we will call C as a core of M .

Note that any finite subset of the universe of M containing a core is also a core of M . Also, there can be multiple cores of M having the same size. A *minimal* core of M is a core, no subset of which is a core of M .

We will use \mathbb{PSC}_f to denote the collection of all classes preserved under substructures modulo a finite core. Similarly, we will use PSC_f to denote the collection of FO definable classes in \mathbb{PSC}_f . We identify classes in PSC_f with their defining FO sentences, and will henceforth treat PSC_f as a set of sentences.

Example 1: Let S be the class of all graphs containing cycles. For any graph in S , the vertices of any cycle is a core of the graph. Thus $S \in \mathbb{PSC}_f$.

Note that $\mathbb{PS} \subseteq \mathbb{PSC}_f$ since for any class in \mathbb{PS} and for any structure in the class, any element is a core. However it is easy to check that S in above example is not in \mathbb{PS} ; so \mathbb{PSC}_f strictly generalizes \mathbb{PS} . Further, the FO inexpressibility of S shows that \mathbb{PSC}_f contains classes not definable in FO.

Example 2: Consider $\phi = \exists x \forall y E(x, y)$. In any graph satisfying ϕ , any witness for x is a core of the graph. Thus $\phi \in PSC_f$. In fact, one can put a uniform bound of 1 on the minimal core size for all models of ϕ .

Again it is easy to see that $PS \subsetneq PSC_f$. Specifically, the sentence ϕ in Example 2 is not in PS . This is because a directed graph with exactly two nodes a and b , and having all directed edges except the self loop on a models ϕ but the subgraph induced by a does not model ϕ . Hence $PS \subsetneq PSC_f$. Extending the example above, one can show that for any sentence φ in Σ_2^0 , in any model of φ , any witness for the \exists quantifiers in φ forms a core of the model. Hence $\Sigma_2^0 \subseteq PSC_f$. In fact, for any sentence in Σ_2^0 , the number of \exists quantifiers serves as a uniform bound on the minimal core size for all models. Surprisingly, even for an arbitrary $\phi \in PSC_f$, it is possible to bound the minimal core size for all models!

Towards the result, we use the notions of *chain* and *union of chain* from the literature. The reader is referred to [5] for the definitions. We denote a chain as $M_1 \subseteq M_2 \subseteq \dots$ and its union as $\bigcup_{i \geq 0} M_i$. We say that a sentence ϕ is *preserved under unions of chains* if for every chain of models of ϕ , the union of the chain is also a model of ϕ . We now recall the following characterization theorem from the '60s [5].

Theorem 1. (Chang-Łoś-Suszko) *A sentence ϕ is preserved under unions of chains iff it is equivalent to a Π_2^0 sentence.*

Now we have the following theorem.

Theorem 2. *A sentence $\phi \in PSC_f$ iff ϕ is equivalent to a Σ_2^0 sentence.*

Proof: We infer from Theorem 1 the following equivalences.

ϕ is equivalent to a Σ_2^0 sentence iff

$\neg\phi$ is equivalent to a Π_2^0 sentence iff

$\forall M_1, M_2, \dots ((M_1 \subseteq M_2 \subseteq \dots) \wedge (M = \bigcup_{i \geq 1} M_i) \wedge \forall i (M_i \models \neg\phi)) \rightarrow M \models \neg\phi$
iff

$\forall M_1, M_2, \dots ((M_1 \subseteq M_2 \subseteq \dots) \wedge (M = \bigcup_{i \geq 1} M_i) \wedge (M \models \phi)) \rightarrow \exists i (M_i \models \phi)$

Assume $\phi \in PSC_f$. Suppose $M_1 \subseteq M_2 \subseteq \dots$ is a chain, $M = \bigcup_{i \geq 0} M_i$ and $M \models \phi$. Then, there exists a finite core C of M . For any $a \in C$, there exists an ordinal i_a s.t. $a \in M_{i_a}$ (else a would not be in the union M). Since C is finite, let $i = \max\{i_a \mid a \in C\}$. Since $i_a \leq i$, we have $M_{i_a} \subseteq M_i$; hence $a \in M_i$ for all $a \in C$. Thus M_i contains C . Since C is a core of M and $M_i \subseteq M$, $M_i \models \phi$ by definition of PSC_f . By the equivalences shown above, ϕ is equivalent to a Σ_2^0 sentence. We have seen earlier that $\Sigma_2^0 \subseteq PSC_f$. ■

Corollary 1. *If $\phi \in PSC_f$, there exists $B \in \mathbb{N}$ such that every model of ϕ has a core of size at most B .*

Proof: Take B to be the number of \exists quantifiers in the equivalent Σ_2^0 sentence. ■

Given Corollary 1 it is natural to ask if B is computable. In this context, the following recent (unpublished) result by Rossman [11] is relevant. Let $|\phi|$ denote the size of ϕ .

Theorem 3. (Rossman) *There is no recursive function $f : \mathbb{N} \rightarrow \mathbb{N}$ such that if $\phi \in PS$, then there is an equivalent Π_1^0 sentence of size at most $f(|\phi|)$. The result holds even for relational vocabularies and further even if PS is replaced with $PS \cap \Sigma_2^0$.*

Corollary 2. *There is no recursive function $f : \mathbb{N} \rightarrow \mathbb{N}$ such that if $\phi \in PS$, then there is an equivalent Π_1^0 sentence with at most $f(|\phi|)$ universal variables. The result holds even for relational vocabularies and further even if PS is replaced with $PS \cap \Sigma_2^0$.*

Proof: Let $\varphi = \forall^n \bar{z} \psi(\bar{z})$ be a Π_1^0 sentence equivalent to ϕ where $n = f(|\phi|)$. Let k be the number of atomic formulae in ψ . Since ϕ and ψ have the same vocabulary, $k \in O(|\phi| \cdot n^{|\phi|})$. The size of the Disjunctive Normal Form of ψ is therefore bounded above by $O(k \cdot n \cdot 2^k)$. Hence $|\varphi|$ is a recursive function of $|\phi|$ if f is recursive. ■

Theorem 3 strengthens the non-elementary lower bound given in [6]. Corollary 2 gives us the following.

Lemma 1. *There is no recursive function $f : \mathbb{N} \rightarrow \mathbb{N}$ s.t. if $\phi \in PSC_f$, then every model of ϕ has a core of size at most $f(|\phi|)$.*

Proof: Consider such a function f . For any sentence ϕ in a relational vocabulary τ s.t. $\phi \in PS$, $\neg\phi$ is equivalent to a Σ_1^0 sentence by Łoś-Tarski theorem. Hence $\neg\phi \in PSC_f$. By assumption about f , the size of minimal models of $\neg\phi$ is bounded above by $n = f(|\phi|) + k$, where k is the number of constants in τ . Therefore, $\neg\phi$ is equivalent to an \exists^n sentence and hence ϕ is equivalent to a \forall^n sentence. Corollary 2 now forbids n , and hence f , from being recursive. It is easy to see that the result extends to vocabularies with functions too (by using functions in a trivial way). ■

Corollary 1 motivates us to consider sentences with bounded cores since all sentences in PSC_f have bounded cores.

2.2 The Case of Bounded Cores

We first give a more general definition.

Definition 2 (Preservation under substructures modulo a bounded core). A class of structures S is said to be preserved under substructures modulo a bounded core (denoted $S \in \mathbb{PSC}$), if $S \in \mathbb{PSC}_f$ and there exists a natural number B dependent only on S such that every structure in S has a core of size at most B .

The collection of all such classes is denoted by \mathbb{PSC} . Let $\mathbb{PSC}(B)$ be the sub-collection of \mathbb{PSC} in which each class has minimal core sizes bounded by B . Then $\mathbb{PSC} = \bigcup_{B>0} \mathbb{PSC}(B)$. An easy observation is that $\mathbb{PSC}(i) \subseteq \mathbb{PSC}(j)$ for $i \leq j$. As before, \mathbb{PSC} and each $\mathbb{PSC}(B)$ contain non-FO definable classes. As an example, the class of forests is in $\mathbb{PSC}(0)$. Let PSC (resp. $PSC(B)$) be the FO definable classes in \mathbb{PSC} (resp. $\mathbb{PSC}(B)$). Observe that $PSC(0)$ is exactly PS and $PSC = \bigcup_{B \geq 0} PSC(B)$. Therefore, PSC generalizes PS . Further, the hierarchy in PSC is strict. Consider $\phi \in PSC(k)$ given by $\phi = \exists x_1 \dots \exists x_k \bigwedge_{1 \leq i < j \leq k} \neg(x_i = x_j)$. Then $\phi \notin PSC(l)$ for $l < k$. From Corollary 1 we have

Lemma 2. $PSC = PSC_f$.

As noted earlier, a Σ_2^0 sentence ϕ with B existential quantifiers is in PSC_f with minimal core size bounded by B . Hence $\phi \in PSC(B)$. In the other direction, Theorem 2 and Lemma 2 together imply that for a sentence $\phi \in PSC(B)$, there is an equivalent Σ_2^0 sentence. We can then ask the following sharper question: For $\phi \in PSC(B)$, is there an equivalent Σ_2^0 sentence having B existential quantifiers?

The remainder of the paper is an account of our studies for a number of special cases of the above question. Since the answer in all of these cases in which arbitrary structures were considered turned out positive, we put forth the following conjecture 1.

Conjecture 1. A sentence $\phi \in PSC(B)$ iff it is equivalent to a Σ_2^0 sentence with B existential quantifiers.

3 Revisiting Relativization

For purposes of our discussion in this and in the remaining sections of the paper, we will assume relational vocabularies (only predicates and constants).

A notion that has proved immensely helpful in proving most of our positive cases for the conjecture is that of *relativization*. Informally speaking, given a sentence ϕ , we would like to define a formula (with free variables \bar{x}) which asserts that ϕ is true in the submodel induced by \bar{x} . The following lemma shows the existence of such a formula.

Lemma 3. If τ is a relational vocabulary, for every $FO(\tau)$ sentence ϕ and variables $\bar{x} = (x_1, \dots, x_k)$, there exists a quantifier-free formula $\phi|_{\bar{x}}$ with free variables \bar{x} such

¹ Post submission of this paper, we have obtained a proof of the conjecture, over arbitrary structures, using non-combinatorial model-theoretic arguments. However, this has not benefited from the scrutiny of the anonymous reviewers. Details of our proof may be found in [12].

that the following holds: Let M be a model and $\bar{a} = (a_1, \dots, a_k)$ be a sequence of elements of M . Then

$$(M, a_1, \dots, a_k) \models \phi|_{\bar{x}} \text{ iff } M(\{a_1, \dots, a_k\}) \models \phi$$

Proof: Let $X = \{x_1, \dots, x_k\}$ and C be the set of constants in τ . First, replace every \forall quantifier in ϕ by $\neg\exists$. Then, replace every subformula of ϕ of the form $\exists x\chi(x, y_1, \dots, y_k)$ by $\bigvee_{z \in X \cup C} \chi(z, y_1, \dots, y_k)$. ■

We refer to $\phi|_{\bar{x}}$ as ‘ ϕ relativized to \bar{x} ’. For clarity of exposition, we will abuse notation and use $\phi|_{\{x_1, \dots, x_k\}}$ to denote $\phi|_{\bar{x}}$ (although \bar{x} is a sequence and $\{x_1, \dots, x_k\}$ is a set), whenever convenient.

We begin with the following observation.

Lemma 4. *Over any given class \mathcal{C} of structures in \mathbb{PS} , if $\phi \leftrightarrow \forall z_1 \dots \forall z_n \varphi$ where φ is quantifier-free, then $\phi \leftrightarrow \psi$ where $\psi = \forall z_1 \dots \forall z_n \phi|_{\{z_1, \dots, z_n\}}$.*

Proof: It is easy to see that $\phi \rightarrow \psi$. Let $M \in \mathcal{C}$ be s.t. $M \models \psi$. Let \bar{a} be an n -tuple from M . Then, by Lemma 3, $M(\bar{a}) \models \phi$. Since $\mathcal{C} \in \mathbb{PS}$, $M(\bar{a}) \in \mathcal{C}$ so that $M(\bar{a}) \models \forall z_1 \dots \forall z_n \varphi$. Then $M(\bar{a}) \models \varphi(\bar{a})$ and hence $M \models \varphi(\bar{a})$. Then $M \models \forall z_1 \dots \forall z_n \varphi$ and hence $M \models \phi$. ■

Using Łoś-Tarski theorem and the above lemma, it follows that a sentence ϕ in PS has an equivalent universal sentence whose matrix is ϕ itself relativized to the universal variables. However we give a proof of this latter fact directly using relativization, and hence an alternate proof of the Łoś-Tarski theorem. We emphasize that our proof works only for relational vocabularies (Łoś-Tarski is known to hold for arbitrary vocabularies). This would show that relativization helps us resolve the conjecture for the case of $B = 0$.

3.1 A Proof of Łoś-Tarski Theorem Using Relativization

We first introduce some notation. Given a τ -structure M , we denote by τ_M , the vocabulary obtained by expanding τ with as many constant symbols as the elements of M - one constant per element. We denote by \mathcal{M} the τ_M structure whose τ -reduct is M and in which each constant in τ_M is interpreted as the element of M corresponding to the constant. It is clear that M uniquely determines \mathcal{M} . Finally, $\mathcal{D}(M)$ denotes the diagram of M - the collection of quantifier free τ_M -sentences true in \mathcal{M} .

Theorem 4. (*Łoś-Tarski*) *A FO sentence ϕ is in PS iff there exists an $n \in \mathbb{N}$ such that ϕ is equivalent to $\forall z_1 \dots \forall z_n \phi|_{\{z_1, \dots, z_n\}}$.*

Proof:

Consider a set of sentences $\Gamma = \{\xi_k \mid k \in \mathbb{N}, \xi_k = \forall z_1 \dots \forall z_k \phi|_{\{z_1, \dots, z_k\}}\}$. Observe that $\xi_{k+1} \rightarrow \xi_k$ so that a finite collection of ξ_k s will be equivalent to ξ_{k^*} where k^* is the highest index k appearing in the collection. We will show that $\phi \leftrightarrow \Gamma$. Once we show this, by compactness theorem, $\phi \leftrightarrow \Gamma_1$ for some finite subset Γ_1 of Γ and by the preceding observation, ϕ is equivalent to $\xi_n \in \Gamma_1$ for some n .

If $M \models \phi$, then since $\phi \in PS$, every substructure of it models ϕ - in particular, the substructure induced by any k -elements of M . Then $M \models \xi_k$ for every k and hence $M \models \Gamma$.

Conversely, suppose $M \models \Gamma$. Then every finite substructure of M models ϕ . Let \mathcal{M} be the τ_M structure corresponding to M . Consider any finite subset S of the diagram $\mathcal{D}(M)$ of M . Let C be the finite set of constants referred to in S . Clearly $\mathcal{M}|_{\tau \cup C}$, namely the $(\tau \cup C)$ -reduct of \mathcal{M} models S since $\mathcal{M} \models \mathcal{D}(M)$. Then consider the substructure \mathcal{M}_1 of $\mathcal{M}|_{\tau \cup C}$ induced by the interpretations of the constants of C - this satisfies S . Now since C is finite, so is \mathcal{M}_1 . Then the τ -reduct of \mathcal{M}_1 - a finite substructure of M models ϕ .

Thus $S \cup \{\phi\}$ is satisfiable by \mathcal{M}_1 . Since S was arbitrary, every finite subset of $\mathcal{D}(M) \cup \{\phi\}$ is satisfiable so that by compactness, $\mathcal{D}(M) \cup \{\phi\}$ is satisfiable by some structure say \mathcal{N} . Then the τ -reduct N of \mathcal{N} is s.t. (i) M is embeddable in N and (ii) $N \models \phi$. Since $\phi \in PS$, the embedding of M in N models ϕ and hence $M \models \phi$. ■

The above proof shows that for $\phi \in PS$, there is an equivalent universal sentence whose matrix is ϕ itself, relativized to the universal variables. In fact, by Lemma 4, there is an optimal (in terms of the number of universal variables) such sentence.

An observation from the proof of Theorem 4 is that, the Łoś-Tarski theorem is true over any class of structures satisfying compactness - hence in particular the class of structures definable by a FO theory (indeed this result is known). But there are classes of structures which are not definable by FO theories but still satisfy compactness: Consider any FO theory having infinite models and consider the class of models of this theory whose cardinality is not equal to a given infinite cardinal. This class satisfies compactness but cannot be definable by any FO theory due to Löwenheim-Skolem theorem. Yet Łoś-Tarski theorem would hold over this class.

Having seen the usefulness of relativization in proving Conjecture 1 when B equals 0, it is natural to ask if this technique works for higher values of B too. We answer this negatively.

3.2 Limitations of Relativization

We show by a concrete example that relativization cannot be used to prove the conjecture in general. This motivates us to derive necessary and sufficient conditions for relativization to work.

Example 3: Consider $\phi = \exists x \forall y E(x, y)$ over $\tau = \{E\}$. Note that ϕ is in $PSC(1)$. Suppose ϕ is equivalent to $\psi = \exists x \forall^n \bar{y} \phi|_{x\bar{y}}$ for some n . Consider the structure $M = (\mathbb{Z}, \leq)$ namely the integers with usual \leq linear order. Any finite substructure of M satisfies ϕ since it has a minimum element (under the linear order). Then taking x to be any integer, we see that $M \models \psi$. However $M \not\models \phi$ since M has no minimum element - a contradiction. The same argument can be used to show that ϕ cannot be equivalent to any sentence of the form $\exists^n \bar{x} \forall^m \bar{y} \phi|_{\bar{x}\bar{y}}$.

We now give necessary and sufficient conditions for relativization to work. Towards this, we introduce the following notion. Consider $\phi \in FO(\tau)$ s.t. $\phi \in PSC(B)$.

Consider a vocabulary τ_B obtained by expanding τ with B fresh constants. Consider the class S_ϕ^{all} of τ_B -structures with the following properties:

1. For each $(M, a_1, \dots, a_B) \in S_\phi^{\text{all}}$ where M is a τ -structure and $a_1, \dots, a_B \in M$, $M \models \phi$ and $\{a_1, \dots, a_B\}$ forms a core of M w.r.t. ϕ .
2. For each model M of ϕ , for each core $C = \{a_1, \dots, a_l\}$ of M w.r.t. ϕ s.t. $l \leq B$ and for each function $p : \{1, \dots, B\} \rightarrow C$ with range C , it must be that $(M, p(1), \dots, p(B)) \in S_\phi^{\text{all}}$.

We now have the following.

Theorem 5. *Given $\phi \in PSC(B)$, the following are equivalent.*

1. S_ϕ^{all} is finitely axiomatizable.
2. ϕ is equivalent to $\exists^B \bar{x} \forall^n \bar{y} \phi|_{\bar{x}\bar{y}}$ for some $n \in \mathbb{N}$.
3. ϕ is equivalent to a $\exists^B \forall^*$ sentence ψ such that in any model M of ψ and ϕ , the following hold:
 - (a) The underlying set of any witness for ψ is a core of M w.r.t. ϕ .
 - (b) Conversely, if C is a core of M w.r.t. ϕ , x_1, \dots, x_B are the \exists variables of ψ and $f : \{x_1, \dots, x_B\} \rightarrow C$ is any function with range C , then $(f(x_1), \dots, f(x_B))$ is witness for ψ in M .

Proof:

(1) \rightarrow (2): Let S_ϕ^{all} be finitely axiomatizable. Check that $S_\phi^{\text{all}} \in \mathbb{PS}$ so that by Łoś-Tarski theorem, it is axiomatizable by a $\Pi_1^0 FO(\tau_B)$ -sentence ψ having say $n \forall$ quantifiers. Further, by Lemma 4 ψ is equivalent to $\gamma = \forall^n \bar{z} \psi|_{\bar{z}}$. Now consider $\varphi = \exists^B \bar{x} \forall^n \bar{y} \phi|_{\bar{x}\bar{y}}$. Firstly, from Lemma 5 $\phi \rightarrow \varphi$. Conversely, suppose $M \models \varphi$. Let a_1, \dots, a_B be witnesses and consider the τ_B -structure $M_B = (M, a_1, \dots, a_B)$. Now $M_B \models \forall^n \bar{y} \phi|_{\bar{x}\bar{y}}$. We will show that $M_B \models \gamma$. Consider $b_1, \dots, b_n \in M$ and let $M_1 = M_B(\{b_1, \dots, b_n\})$. Then $M_1 \models \forall^n \bar{y} \phi|_{\bar{x}\bar{y}}$. Check that the τ -reduct of M_1 (i) models ϕ and (ii) contains $\{a_1, \dots, a_B\}$ as a core. Then $M_1 \in S_\phi^{\text{all}}$ and hence $M_1 \models \psi$. Since b_1, \dots, b_n were arbitrary, $M_B \models \gamma$. Since $\gamma \leftrightarrow \psi$ and ψ axiomatizes S_ϕ^{all} , the τ -reduct of M_B , namely M , models ϕ .

(2) \rightarrow (3): Take ψ to be $\exists^B \bar{x} \forall^n \bar{y} \phi|_{\bar{x}\bar{y}}$. Consider a model M of ϕ and ψ . The set C of elements of any witness for ψ forms a core of M w.r.t. ψ . Then since $\phi \leftrightarrow \psi$, C is also a core of M w.r.t. ϕ . Conversely, consider a core C of M w.r.t. ϕ . Then any substructure of M containing C satisfies ϕ . Then check that elements of C form a witness for ψ .

(3) \rightarrow (1): Let $\phi \leftrightarrow \psi$ where $\psi = \exists^B \bar{x} \forall^n \bar{y} \beta(\bar{x}, \bar{y})$ where β is quantifier free and ψ satisfies the conditions mentioned in (3). Consider $\varphi = \forall^n \bar{y} \beta[x_1 \mapsto c_1, \dots, x_B \mapsto c_B]$ where c_1, \dots, c_B are B fresh constants and $x_i \mapsto c_i$ means replacement of x_i by c_i . If $M_B = (M, a_1, \dots, a_B) \models \varphi$, then $M \models \psi$ and hence $M \models \phi$. Since a_1, \dots, a_B are witnesses for ψ in M , they form a core of M w.r.t. ϕ by assumption, so that $M_B \in S_\phi^{\text{all}}$. Conversely, if $M_B = (M, a_1, \dots, a_B) \in S_\phi^{\text{all}}$, then $M \models \phi$ and a_1, \dots, a_B form a core in M . Then by assumption, $M \models \psi$ and a_1, \dots, a_B are witnesses for ψ . Then $M_B \models \varphi$. To sum up, φ axiomatizes S_ϕ^{all} . ■

Consider ϕ and M in Example 3 above. Take any finite substructure M_1 of M - it models ϕ . There is exactly one witness for ϕ in M_1 , namely the least element under \leq . However every element in M_1 serves as a core. The above theorem shows that no $\exists\forall^*$ sentence will be able to capture exactly all the cores through its \exists variable.

In the following sections, we shall study the conjecture for several special classes of FO and over special structures. Interestingly, in most of the cases in which the conjecture turns out true, relativization works! However we also show a case for the conjecture in which relativization does not work, yet the conjecture is true.

4 Positive Cases for the Conjecture

4.1 The Conjecture Holds for Special Fragments of FO

Unless otherwise stated, we consider relational vocabularies throughout the section. The following lemma will be repeatedly used in the subsequent results.

Lemma 5. *Let $\phi \in PSC(B)$. For every $n \in \mathbb{N}$, ϕ implies $\exists^B \bar{x} \forall^n \bar{y} \phi|_{\bar{x}\bar{y}}$.*

Proof: Suppose $M \models \phi$. Since $\phi \in PSC(B)$, there is a core C of M of size at most B . Interpret \bar{x} to include all the elements of C (in any which way). Since C is a core, for any n -tuple \bar{d} of elements of M , having underlying set D , the substructure of M induced by $C \cup D$ models ϕ . Then $(M, \bar{a}, \bar{d}) \models \phi|_{\bar{x}\bar{y}}$ for all \bar{d} from M . ■

Lemma 6. *Let τ be a monadic vocabulary containing k unary predicates. Let $\phi \in FO(\tau)$ be a sentence of rank r s.t. $\phi \in PSC(B)$. Then ϕ is equivalent to ψ where $\psi = \exists^B \bar{x} \forall^n \bar{y} \phi|_{\bar{x}\bar{y}}$ where $n = r \times 2^k$. For $B = 0$, n is optimal i.e. there is an FO sentence in $PSC(0)$ for which any equivalent Π_1^0 sentence has at least n quantifiers.*

Proof: That ϕ implies ψ follows from Lemma 5. For the converse, suppose $M \models \psi$ where $n = r \times 2^k$. By an Ehrenfeucht-Früssé game argument, we can show that M contains a substructure M_S such that (i) $M \equiv_r M_S$, with $|M_S| \leq n$ and (ii) for any extension M' of M_S in M , $M' \equiv_r M_S$. The substructure M_S is obtained by taking up to r elements of each colour $c \in 2^\tau$ present in M . An element a in structure M is said to have colour c if for every predicate $P \in \Sigma$, $M \models P(a)$ iff $P \in c$. Since $M \models \psi$, there exists witnesses \bar{a} for ψ in M . Choose \bar{b} to be an n -tuple which includes the elements of M_S . This is possible because $|M_S| \leq n$. Then we have, $(M, \bar{a}, \bar{b}) \models \phi|_{\bar{x}\bar{y}}$ so that $M(\bar{a}\bar{b}) \models \phi$. But $M_S \subseteq M(\bar{a}\bar{b}) \subseteq M$ so that $M(\bar{a}\bar{b}) \equiv_r M$. Then $M \models \phi$.

To see the optimality of n for $B = 0$, consider the sentence ϕ which states that there exists at least one colour $c \in 2^\tau$ such that there exist at most $r - 1$ elements with colour c . The sentence ϕ can be written as a formula with rank r , as the disjunction over all colours, of sentences of the form, $\exists x_1 \exists x_2 \dots \exists x_{r-1} \forall x_r (\bigwedge_{i=1}^{r-1} x_r \neq x_i \rightarrow \neg C(x_r))$. From the preceding paragraph, $\phi \leftrightarrow \forall^n \bar{y} \phi|_{\bar{y}}$ where $n = r \times 2^k$. Suppose ϕ is equivalent to a \forall^s sentence for some $s < n$. Then by Lemma 4, $\phi \leftrightarrow \varphi$ where $\varphi = \forall^s \bar{y} \phi|_{\bar{y}}$. Then consider the structure M , which has r elements of each colour. Clearly, $M \not\models \phi$. However check that every s -sized substructure of M models ϕ . Then $M \models \varphi$ and hence $M \models \phi$ - a contradiction. ■

Lemma 7. *Let $S \in \text{PSC}(B)$ be a finite collection of τ -structures so that S is definable by a Σ_2^0 sentence $\phi \in \text{PSC}(B)$. Then S is definable by the sentence ψ where $\psi = \exists^B \bar{x} \forall^{n\bar{y}} \phi|_{\bar{x}\bar{y}}$ for some $n \in \mathbb{N}$.*

Proof: Check that all structures in S must be of finite size so that ϕ exists. Let the size of the largest structure in S be at most n . Consider ψ . Lemma 5 shows that $\phi \rightarrow \psi$. Conversely, suppose $M \models \psi$. Then there exists a witness \bar{a} s.t. any extension of $M(\bar{a})$ within M with at most n additional elements models ϕ . Since M is of size at most n , taking the extension M of $M(\bar{a})$, we have $M \models \phi$. Since ϕ defines S so does ψ . ■

Lemma 8. *Consider $\phi \in \Pi_2^0$ given by $\phi = \forall^n \bar{x} \exists^m \bar{y} \beta(\bar{x}, \bar{y})$ where β is quantifier free. If $\phi \in \text{PSC}(B)$, then ϕ is equivalent to ψ where $\psi = \exists^B \bar{u} \forall^n \bar{v} \phi|_{\bar{u}\bar{v}}$.*

Proof: From Lemma 5 $\phi \rightarrow \psi$. For the converse, let $M \models \psi$ and let \bar{a} be a witness. Consider an n -tuple \bar{b} from M . Then $M_1 = M(\bar{a}\bar{b})$ is s.t. $M_1 \models \phi$. Then for $\bar{x} = \bar{b}$, there exists $\bar{y} = \bar{d}$ s.t. \bar{d} is an m -tuple from M_1 and $M_1 \models \beta(\bar{b}, \bar{d})$. Then $M \models \beta(\bar{b}, \bar{d})$ since $M_1 \subseteq M$. Hence $M \models \phi$. ■

Lemma 9. *Suppose $\phi \in \text{PSC}(B)$ and $\neg\phi \in \text{PSC}(B')$. Then ϕ is equivalent to ψ where $\psi = \exists^B \bar{x} \forall^{B'} \bar{y} \phi|_{\bar{x}\bar{y}}$.*

Proof: From Lemma 5 ϕ implies ψ . For the converse, suppose $M \models \psi$. Then there is a witness \bar{a} for ψ s.t. for any B' -tuple \bar{b} , the substructure induced by $\bar{a}\bar{b}$ i.e. $M(\bar{a}\bar{b})$ models ϕ . Suppose $M \not\models \phi$. Then $M \models \neg\phi$ so that there is a core C of M w.r.t. $\neg\phi$, of size at most B' . Let \bar{d} be a B' -tuple which includes all the elements of C . Then $M(\bar{a}\bar{d}) \models \phi$. But $M(\bar{a}\bar{d}) \subseteq M$ contains C so that $M(\bar{a}\bar{d}) \models \neg\phi$ – a contradiction. ■

Observe that for the special case of $B = 0$, we get combinatorial proofs of Łoś-Tarski theorem for the fragments mentioned above. Moreover all of these proofs and hence the results hold in the finite. We mention that the result of Lemma 8 holding in the finite was proved by Compton too (see [7]). We were unaware of this until recently and have independently arrived at the same result. The reader is referred to [12] for our studies on more positive cases of Łoś-Tarski in the finite.

Interestingly, Lemma 9 has implications for the Δ_2^0 fragment of FO. Define $\Delta_2^0(k, l) \subseteq \Delta_2^0$ to be the class of sentences which have a $\exists^k \forall^*$ and a $\forall^l \exists^*$ equivalent. Note that $\Delta_2^0 = \bigcup_{l, k \geq 0} \Delta_2^0(k, l)$. Lemma 9 gives us the following right away.

Theorem 6. *The following are equivalent:*

1. $\phi \in \text{PSC}(k)$ and $\neg\phi \in \text{PSC}(l)$.
2. ϕ is equivalent to a $\exists^k \forall^l$ and a $\forall^l \exists^k$ sentence.
3. $\phi \in \Delta_2^0(k, l)$.

As a corollary, we see that $\Delta_2^0(k, l)$ is a finite class up to equivalence. We are not aware of any other semantic characterization of these natural fragments of Δ_2^0 . This highlights the importance of the notion of cores and the sizes thereof.

4.2 The Conjecture over Special Classes of Structures

We first look at the conjecture over finite words. These are finite structures in the vocabulary containing one binary predicate \leq (always interpreted as a linear order) and a finite number of unary predicates (which form a partition of the universe). Interestingly, we obtain something stronger than the conjecture. Towards this, we note that the idea of relativization can be naturally extended to MSO. Given ϕ in MSO and a set of variables $Z = \{z_1, \dots, z_n\}$, $\phi|_Z$ is obtained by first converting all $\forall X$ to $\neg\exists X$ and then replacing every subformula $\exists X\chi(X, \dots)$ with $\bigvee_{Y\subseteq Z}((\bigwedge_{z\in Y} X(z) \wedge \bigwedge_{z\in Z\setminus Y} \neg X(z)) \wedge \chi(X, \dots))$. The resulting FO formula is then relativized to Z and simplified to eliminate the (original) SO variables. As before, abusing notation, we use $\phi|_Z$ and $\phi|_{\bar{z}}$ interchangeably.

Theorem 7. *Over words, a MSO sentence ϕ is in $\mathbb{PSC}(B)$ iff it is equivalent to ψ where $\psi = \exists^B \bar{x} \forall^k \bar{y} \phi|_{\bar{x}\bar{y}}$ for some $k \in \mathbb{N}$.*

Proof sketch: We use the fact that over words, by the Büchi-Elgot-Trakhtenbrot theorem [4], MSO sentences define regular languages. The ‘If’ direction is easy. For the ‘Only if’ direction, let the regular language L defined by ϕ be recognized by an n state automaton, say \mathcal{M} . If there is no word of length $> N = (B + 1) \times n$ in L , then L is a finite language of finite words and hence from Lemma 7 we are done. Else suppose there is a word of length $> N$ in L . Then consider ψ above for $k = N$. It is easy to observe that ϕ implies ψ . In the other direction, suppose $w \models \psi$ for some word w . Then there exists a set A of elements i_1, \dots, i_m s.t. (i) $m \leq B$ and $i_1 < i_2 \dots < i_m$ and (ii) every substructure of w of size at most $N + m$ containing A models ϕ . We claim (proof sketched below) that there exists a substructure w_1 of w containing A such that (i) $|w_1| \leq N$ and (ii) $w_1 \in L$ iff $w \in L$. Then w_1 models ϕ and hence $w \models \phi$. Thus ψ implies ϕ and hence is equivalent to ϕ .

The proof of the claim used in the argument above proceeds as follows. Let q_j be the state reached by automaton \mathcal{M} upon reading the subword $w[1 \dots i_j]$. The subword $w[(i_j + 1), \dots, i_{j+1}]$ takes \mathcal{M} from q_j to q_{j+1} through a sequence S of states. Since \mathcal{M} has only n states, if $w[(i_j + 1), \dots, i_{j+1}]$ is long, then S will contain at least one loop. Then getting rid of the subwords that give rise to loops, we will be able to obtain a subword of $w[(i_j + 1), \dots, i_{j+1}]$ that takes \mathcal{M} from q_j to q_{j+1} without causing \mathcal{M} to loop in between. It follows that this subword must be of length at most n . Collecting such subwords of $w[(i_j + 1), \dots, i_{j+1}]$ for each j and concatenating them, we get a subword of w of length at most N containing set A that takes \mathcal{M} from the initial state to the same state as w . Details can be found in [12]. ■

For the special case of $B = 0$, we obtain Łoś-Tarski theorem for words and also give a bound for the number of \forall s in the equivalent Π_1^0 sentence in terms of the number of states of the automaton for ϕ (A simpler proof of Łoś-Tarski using Higman’s lemma can be found in [12] though this does not tell anything about the number of \forall s). We have not encountered this result in our literature survey.

So far, relativization has worked in all the cases we have seen. We now give an example of a class of structures over which relativization fails, yet the conjecture is true.

Consider a subclass \mathcal{C} of bounded degree graphs in which each graph is a collection (finite or infinite) of *oriented* paths (finite or infinite). For clarity, by oriented path we mean a graph isomorphic to a connected induced subgraph of the graph (V, E) where $V = \mathbb{Z}$ and $E = \{(i, i + 1) \mid i \in \mathbb{Z}\}$. Observe that \mathcal{C} can be axiomatized by a theory \mathcal{T} which asserts that every node has in-degree at most 1 and out-degree at most 1 and that there is no directed cycle of length k for each $k \geq 0$. We first show the following.

Lemma 10. *For each $B \geq 1$, there is a sentence $\phi \in PSC(B)$ which is not equivalent, over \mathcal{C} , to any ψ of the form $\exists^B \bar{x} \forall^n \bar{y} \phi|_{\bar{x}\bar{y}}$.*

Proof: Consider ϕ which asserts that there are at least B elements of *total* degree at most 1 where total degree is the sum of in-degree and out-degree. Clearly $\phi \in PSC(B)$ since it is expressible as a $\exists^B \forall^*$ sentence. Suppose ϕ is equivalent to ψ of the form above for some $n \in \mathbb{N}$. Consider $M \in \mathcal{C}$ which is a both-ways infinite path so that every node in M has total degree 2 - then $M \not\models \phi$. Consider B distinct points on this path at a distance of at least $2n$ from each other and form a B -tuple say \bar{a} with them. Let \bar{b} be any n -tuple from M . Now observe that $M(\bar{a}\bar{b})$ is a finite structure which has at least B distinct paths (0-sized paths included). Then $M(\bar{a}\bar{b}) \models \phi$ so that $(M, \bar{a}, \bar{b}) \models \phi|_{\bar{x}\bar{y}}$. Since \bar{b} was arbitrary, $M \models \psi$ so that $M \models \phi$. Contradiction. ■

However the conjecture holds over \mathcal{C} ! The proof is currently lengthy so we provide only a sketch and refer the reader to [12] for details.

Theorem 8. *Over the class \mathcal{C} of graphs defined above, $\phi \in PSC(B)$ iff ϕ is equivalent to a $\exists^B \forall^*$ sentence.*

Proof Sketch: If $\tau = \{E\}$ is the vocabulary of ϕ , let τ_B be a vocabulary obtained by adding B fresh constants to τ . Given a class \mathcal{S} of τ -structures, define \mathcal{S}_B to be the class of all τ_B -structures s.t. the τ -reduct of each structure in \mathcal{S}_B is in \mathcal{S} . Then the proof can be divided into two main steps. Below \equiv denotes elementary equivalence.

Step 1: Given ϕ , define class $\mathcal{C}' \subseteq \mathcal{C}$ such that for every structure $A \in \mathcal{C}_B$, there exists a structure $D \in \mathcal{C}'_B$ such that $A \equiv D$ (Property I). Since compactness theorem holds over \mathcal{C}_B (as \mathcal{C}_B is defined by the same theory \mathcal{T} as \mathcal{C}), it also holds over \mathcal{C}'_B .

Step 2: Show that ϕ is equivalent to an $\exists^B \forall^*$ sentence over \mathcal{C}' , hence showing the same over \mathcal{C} as well.

Note: The conditions in **Step 1** imply that for every $A \in \mathcal{C}$, there exists a $D \in \mathcal{C}'$ such that $A \equiv D$. Then since compactness theorem holds over \mathcal{C} , it also holds over \mathcal{C}' .

Suppose the rank of ϕ is m . We define \mathcal{C}' to be the class of graphs $G \in \mathcal{C}$ such that either (a) there exists a bound n_G (dependent on G) such that all paths in G have length less than n_G (this does not mean that G is finite - there could be infinite paths of the same length in G) or (b) there are at least $(B + m + 2)$ paths in G that are infinite in both directions. It can be shown that \mathcal{C}' satisfies Property I (see [12]).

Now, to show **Step 2**, we use the following approach.

Let $P \in \mathcal{C}'$ be s.t. $P \models \phi$. Choose a core Z of P (recall that $\phi \in PSC(B)$). Let $M_P \in \mathcal{C}'_B$ be a τ_B -structure whose τ -reduct is P , and in which each element of Z is assigned to some constant. Let Γ^{M_P} be the set of all \forall^* sentences true in M_P .

We can show that (see [I2]) if $M' \in \mathcal{C}'_B$ is such that $M' \models \Gamma^{M_P}$, then $M' \models \phi$. That is, if every finite substructure of M' is embeddable in M_P , then $M' \models \phi$. Then over \mathcal{C}'_B , $\Gamma^{M_P} \rightarrow \phi$. Now, since \mathcal{C}'_B satisfies the compactness theorem, there exists a finite subset $\Gamma_0^{M_P}$ of Γ^{M_P} such that $\Gamma_0^{M_P} \rightarrow \phi$ over \mathcal{C}'_B . Note that, since $\Gamma_0^{M_P}$ is a conjunction of \forall^* sentences, we can assume that $\Gamma_0^{M_P}$ is a single \forall^* sentence.

Let ϕ_P be the τ -sentence of the form $\exists^B \forall^*$ obtained by replacing the B constants in $\Gamma_0^{M_P}$ with B fresh variables, and by existentially quantifying these variables. We can then show that $\phi_P \rightarrow \phi$. It is also easy to see that $\phi \rightarrow \bigvee_{P \in \mathcal{C}', P \models \phi} \phi_P$, since if $P \models \phi$, then the witnesses of the \exists quantifiers in ϕ_P can be chosen to be the core Z mentioned above. By the compactness theorem over \mathcal{C}' , there exists a finite set of structures, say $\{P_1, \dots, P_m\}$, such that $P_i \in \mathcal{C}'$, $P_i \models \phi$ and $\phi \rightarrow \bigvee_{i=1}^{i=m} \phi_{P_i}$. Then, we have $\phi \leftrightarrow \bigvee_{i=0}^{i=m} \phi_{P_i}$ over \mathcal{C}' . Since each ϕ_{P_i} is of the form $\exists^B \forall^*$, the sentence $\bigvee_{i=0}^{i=m} \phi_{P_i}$ is also of the same form. This completes **Step 2** of the proof. ■

5 Conjecture Fails over Special Classes of Structures

We first look at the class \mathcal{F} of all finite structures. Łoś-Tarski theorem fails over this class and hence so does Conjecture [I] (for $B = 0$). However, we have the following stronger result. We prove it for relational vocabularies (constants permitted).

Lemma 11. *For relational vocabularies, Conjecture [I] fails, over \mathcal{F} , for each $B \geq 0$.*

Proof: We refer to [I] for the counterexample χ for Łoś-Tarski in the finite. Let τ be the vocabulary of χ (i.e. $\{\leq, S, a, b\}$) along with a unary predicate U . Let us call an element x as having colour 0 in a structure if $U(x)$ is true in the structure and having colour 1 otherwise. Let φ be a sentence asserting that there are exactly B elements having colour 0 and these are different from a and b . Then consider $\phi = \neg\chi \wedge \varphi$. Check that since $\neg\chi$ is preserved under substructures in the finite, in any model of ϕ , the B elements of colour 0 form a core of the model w.r.t. ϕ . Then $\phi \in PSC(B)$. Suppose ϕ is equivalent to ψ given by $\exists^B \bar{x} \forall^n \bar{y} \beta$ where β is quantifier-free. Observe that in any model of ϕ and ψ , any witness for ψ must include all the B elements of colour 0 (else the substructure formed by the witness would not model φ and hence ϕ , though it would model ψ). Consider the structure $M = (\{0, 1, \dots, B + 2n + 3\}, \leq, S, a, b, U)$ where \leq is the usual linear order on numbers, S is the (full) successor relation of \leq , $a = 0, b = B + 2n + 3$ and $U = \{1, \dots, B\}$. Now $M \not\models \phi$ since $M \not\models \neg\chi$. Consider M_1 which is identical to M except that $S(B + n + 1, y)$ is false in M_1 for all y . Then $M_1 \models \phi$ so that $M_1 \models \psi$. Any witness \bar{a} for ψ must include all the B colour 0 elements of M_1 . Then choose exactly the same value, namely \bar{a} , from M to assign to \bar{x} . Choose any \bar{b} as \bar{y} from M . Check that it is possible to choose \bar{d} as \bar{y} from M_1 s.t. $M(\bar{a}\bar{b})$ is isomorphic to $M_1(\bar{a}\bar{d})$ under the isomorphism f given by $f(0) = 0, f(B + 2n + 3) = B + 2n + 3, f(a_i) = a_i$ and $f(b_i) = d_i$ where $\bar{a} = (a_1, \dots, a_B), \bar{b} = (b_1, \dots, b_n)$ and $\bar{d} = (d_1, \dots, d_n)$. Then since $M_1 \models \beta(\bar{a}, \bar{d}), M \models \beta(\bar{a}, \bar{b})$. Then M models ψ , and hence ϕ . But that is a contradiction. ■

The example expressed by χ can also be written as a sentence in a purely relational vocabulary. Then one can do a similar proof as above to show that for purely relational vocabularies too, for each $B \geq 0$, Conjecture [I] fails over \mathcal{F} (see [I2]).

So far, in all the cases we have seen, it has always been the case that Conjecture [1](#) and Łoś-Tarski theorem either are both true or are both false. We then finally have the following result which is our first instance of a class of structures over which Łoś-Tarski theorem holds but the conjecture fails.

Theorem 9. *Over the class \mathcal{C} of graphs in which each graph is a finite collection of finite undirected paths, for each $B \geq 2$, there is a sentence $\phi \in \text{PSC}(B)$ which is not equivalent to any $\exists^B \forall^*$ sentence. However, Łoś-Tarski theorem holds over \mathcal{C} .*

Proof: Łoś-Tarski theorem holds from the results of Dawar et al. over bounded degree structures [\[2\]](#). As a counterexample to the conjecture for $B \geq 2$, consider the property D which asserts that there are at least B paths in the graph (0 length included). It can be shown (see [\[12\]](#)) that D is equivalent to the following condition D' parametrized by B : (The number of nodes of degree 0) + $\frac{1}{2} \times$ (the number of nodes of degree 1) $\geq B$. Then given B , take ϕ to be the sentence expressing D' for B . We reason out for the case of $B = 2$ since for the other cases an analogous reasoning can be done (see [\[12\]](#)).

Every model N of ϕ has at least 2 paths of length ≥ 0 . Consider set A formed by an end point of one path and an end point of the other path. Check that A is a core of N w.r.t. ϕ so that $\phi \in \text{PSC}(2)$. Suppose ϕ is equivalent over \mathcal{C} to $\psi = \exists^2 \bar{x} \forall^n \bar{y} \beta$ where β is quantifier-free. Consider a model N of ϕ having exactly 2 paths each of length $\geq 5n$. Then since $N \models \psi$, consider the witnesses a_1, a_2 for ψ . It cannot be that a_1, a_2 are both from the same path else the path by itself would be a model for ψ and hence ϕ . Now consider a structure M containing a single path that is of length $\geq 5n$ with end points p_1, p_2 . If a_1 (resp. a_2) is at a distance of $\leq n$ from any end point in N , choose a point b_1 (resp. b_2) at the same distance from p_1 (resp. p_2) in M . Else choose b_1 (resp. b_2) at a distance of $n + 1$ from p_1 (resp. p_2). Choose any \bar{d} as \bar{y} from M . Check that it is possible to choose \bar{e} as \bar{y} from N s.t. $M(b_1 b_2 \bar{d})$ is isomorphic to $N(a_1 a_2 \bar{e})$ under the isomorphism f given by $f(b_i) = a_i, f(d_j) = e_j$ where $\bar{d} = (d_1, \dots, d_n)$ and $\bar{e} = (e_1, \dots, e_n)$. Since $N \models \beta(a_1, a_2, \bar{e}), M \models \beta(b_1, b_2, \bar{d})$. Then M models ψ , and hence ϕ . Contradiction. ■

Interestingly however, the conjecture holds over \mathcal{C} for $B = 1$. We also give a simpler proof for the case of $B = 0$ i.e. Łoś-Tarski over \mathcal{C} (see [\[12\]](#)).

6 Conclusion and Future Work

For future work, we would like to investigate cases for which combinatorial proofs of Conjecture [1](#) can be obtained. This would potentially improve our understanding of the conditions under which combinatorial proofs can be obtained for the Łoś-Tarski theorem as well. An important direction of future work is to investigate whether the conjecture holds for important classes of finite structures for which the Łoś-Tarski theorem holds. Examples of such classes include those considered by Atserias et al in [\[2\]](#). We have also partially investigated how preservation theorems can be used to show FO inexpressibility for many typical examples (see [\[13\]](#)). We would like to pursue this line of work as well in future.

Acknowledgements. We are extremely thankful to Anand Pillay for helping us prove Theorem 2 which inspired us to go further to pose our conjecture and study it. Our sincere thanks to Ben Rossman for giving us a patient hearing and for sharing with us his unpublished result (Theorem 3). Many thanks to Nutan Limaye and Akshay Sundararaman for discussions on inexpressibility proofs using preservation theorems.

References

1. Alechina, N., Gurevich, Y.: Syntax vs. Semantics on Finite Structures. In: Mycielski, J., Rozenberg, G., Salomaa, A. (eds.) Structures in Logic and Computer Science. LNCS, vol. 1261, pp. 14–33. Springer, Heidelberg (1997)
2. Atserias, A., Dawar, A., Grohe, M.: Preservation under extensions on well-behaved finite structures. *SIAM J. Comput.* 38(4), 1364–1381 (2008)
3. Atserias, A., Dawar, A., Kolaitis, P.G.: On preservation under homomorphisms and unions of conjunctive queries. *J. ACM* 53(2), 208–237 (2006)
4. Büchi, J.R.: Weak second-order arithmetic and finite automata. *Z. Math. Logik Grundlagen Math.* 6, 66–92 (1960)
5. Chang, C.C., Keisler, H.J.: *Model Theory*, 3rd edn. Elsevier Science Publishers (1990)
6. Dawar, A., Grohe, M., Kreutzer, S., Schweikardt, N.: Model Theory Makes Formulas Large. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 913–924. Springer, Heidelberg (2007)
7. Gurevich, Y.: Toward logic tailored for computational complexity. In: *Computation and Proof Theory*, pp. 175–216. Springer (1984)
8. Libkin, L.: *Elements of Finite Model Theory*. Springer (2004)
9. Rosen, E.: Some aspects of model theory and finite structures. *Bulletin of Symbolic Logic* 8(3), 380–403 (2002)
10. Rossman, B.: Homomorphism preservation theorems. *J. ACM* 55(3), 15:1–15:53 (2008)
11. Rossman, B.: Personal Communication (2012)
12. Sankaran, A., Adsul, B., Madan, V., Kamath, P., Chakraborty, S.: Preservation under substructures modulo bounded cores. CoRR, abs/1205.1358 (2012)
13. Sankaran, A., Limaye, N., Sundararaman, A., Chakraborty, S.: Using preservation theorems for inexpressibility results in first order logic. Technical report (2012), <http://www.cfdvs.iitb.ac.in/reports/index.php>

A Logic of Plausible Justifications

L. Menasché Schechter*

Department of Computer Science, Federal University of Rio de Janeiro, Brazil
luisms@dcc.ufrj.br

Abstract. In this work, we combine the frameworks of Justification Logics and Logics of Plausibility-Based Beliefs to build a logic for Multi-Agent Systems where each agent can explicitly state his justification for believing in a given sentence. Our logic is a normal modal logic based on the standard Kripke semantics, where we provide a semantic definition for the evidence terms and define the notion of plausible evidence for an agent, based on plausibility relations in the model. This way, unlike traditional Justification Logics, justifications can be actually faulty and unreliable. In our logic, agents can disagree not only over whether a sentence is true or false, but also on whether some evidence is a valid justification for a sentence or not. After defining our logic and its semantics, we provide a strongly complete axiomatic system for it and show that it has the finite model property and is decidable. Thus, this logic seems to be a good first step for the development of a dynamic logic that can model the processes of argumentation and debate in multi-agent systems.

1 Introduction and Motivation

Epistemic logics [15] are a particular kind of modal logics [10] where the modalities are used to describe epistemic notions such as *knowledge* and *belief* of agents. Traditional epistemic logics are expressive enough to describe knowledge and belief of multiple agents in a multi-agent system, including higher-order notions, such as the knowledge of one agent about the knowledge of another, and some notions of knowledge and belief that are related to groups of agents, such as “everybody in a group knows...” or “it is common knowledge in a group...”.

Nevertheless, such epistemic logics have two important limitations. The first is that the knowledge or belief of an agent is static, i.e., it does not change over time. One of the reasons for this is that, in such logics, it is not possible to describe communication between the agents. The second is that the knowledge modeled by such logics is *implicit*, which means that if the agent knows something, then he knows it for some reason that remains unspecified.

In order to deal with the first limitation, *Dynamic Epistemic Logics* [11] were developed. In these logics, we can describe acts of communication between the agents. Such acts consist of *truthful* announcements that are made by one of the agents (or an external observer) to the other agents (or a sub-group of them).

* The author was supported by a grant from the Brazilian Research Agency FAPERJ (grant number E-26/110.716/2011).

In works such as [4, 5, 7, 8], this framework of dynamic logics was extended so that not only knowledge, but also beliefs (which, unlike knowledge, may turn out to be actually false) could evolve over time. The semantics of such logics of *dynamic beliefs* is based on *Plausibility Models*, where each agent has a plausibility order for the possible states of the model and he believes in those sentences that are true in the most plausible states according to his plausibility order. The change of beliefs is then modeled as changes in the plausibility orders of the agents.

In order to deal with the second limitation, *Justification Logics* [1-3] were developed. In these logics, instead of formulas simply stating “Agent i knows φ ”, we have formulas that state “ t is agent i ’s justification (or evidence) for knowing φ ”. Thus, these are logics of *explicit* knowledge, where every knowledge that an agent has is accompanied by an explicit justification for it. This is why Justification Logics are also called *Logics of Explicit Knowledge* or *Logics of Evidence-Based Knowledge*.

Justification Logics came from a previous framework, called Logic of Proofs [1], where justifications described formal proofs of arithmetical theorems. Thus, justifications in traditional Justification Logics are usually rather strong and the presence of a justification for a logical sentence entails the truth of that sentence [2, 3].

In the processes of argumentation and debate, be it an internal debate or a public debate where each agent tries to convince an external observer of his particular point of view, it is unrealistic to say that all of the announcements are *truthful*. The realistic assumption is that the announcements are merely *sincere*, i.e., each agent believes in what he announces. However, in order to convince others of their belief, an agent should not only state what he believes in, but also *why* he believes in it. So, the appropriate logic to model these processes would be a *dynamic logic of evidence-based beliefs*. In order to build such a logic, we combine aspects of Justification Logics with Plausibility Models, while considering now, unlike traditional Justification Logics, that justifications can actually be faulty and unreliable (so they no longer entail truth). Using Plausibility Models, we give a notion of plausible evidence, or plausible justification, for an agent. So, if an agent has a plausible evidence for a sentence, then he will believe in that sentence, but, as the evidence can possibly be faulty, this belief has the possibility to be false.

In the present work, we take a first step in order to build such a logic for the description of the processes of argumentation and debate. We build a normal modal logic (for the definition of normal modal logics, [10] can be consulted) where we can describe the plausibility of evidences for all the different agents, give a sound and strongly complete axiomatic system for this logic and show that it has the finite model property and is decidable.

As our next step, we plan to build a dynamic logic of *explicit* beliefs, adding to the present logic the actions that would model the communications between agents during the processes of argumentation and debate. This is not a trivial task. The standard announcements that describe changes of knowledge [11],

sometimes called *hard announcements*, are too strong for our needs, since they are required to be *truthful* and not merely *sincere* (using such announcements without respecting the requisite that the announced formula should be true can generate logical inconsistencies). On the other hand, the standard announcements that describe changes of beliefs, called *belief upgrades* or *soft announcements* [6] are too weak, since, even though they are only required to be *sincere*, they still make the agents receiving the announcement start believing in it, regardless of their current beliefs. In our desired framework, the announcement of a sentence should be accompanied by a justification as to why the agent performing the announcement believes in it. Then, each agent receiving the announcement should judge by himself whether he should start believing or not in the announced sentence, based on his current beliefs both about what was announced and about the justification that was given.

There are, in the literature, works that combine aspects of Justification Logics and Dynamic Epistemic Logics. [19] developed the first proposal of a Justification Logic with communication between the agents. However, these communication actions were extremely simple. Later, the series of works [16–18] developed logics that add to Justification Logics a series of communication actions, some rather complex. However, those actions are all from the family of *hard announcements*, so they cannot be used for our purpose. Our combination of Justification Logics with explicit evidence terms and Plausibility Models and our use of evidence terms to model explicit *beliefs* instead of explicit *knowledge* seems to be a novel approach. [9] also developed a logic of evidence-based beliefs, but, unlike our logic, it has no explicit evidence terms in the language and some of the modalities are non-normal. Besides that, also unlike our logic, no complete proof system for that logic is presented.

The rest of this paper is organized as follows. In Section 2, we introduce the necessary concepts that are used as building blocks for our logic: Justification Logic and Plausibility Models. The language and semantics of our logic, called Logic of Plausible Justifications (LPJ), is presented in Section 3, where we also show that our logic has the finite model property and is decidable and present a sound and strongly complete axiomatic system for it. Finally, in Section 4, we state our final remarks and point out potential further developments, including the one which originally motivated this work: the construction of a dynamic logic that can model argumentation and debate in multi-agent systems.

2 Background Concepts

This section presents two important concepts for the construction of our logic: Justification Logic and Plausibility Models.

2.1 Justification Logic

In this section, we provide a brief account of Justification Logic. For more details, [1–3, 12] can be consulted.

Definition 1. *The language of basic Justification Logic consists of a countable set Φ of proposition symbols, a countable set \mathcal{C} of evidence constants, a countable set \mathcal{X} of evidence variables, all pairwise disjoint, and the boolean connectives \neg and \wedge . The formulas φ and the evidence terms t of the language are defined as follows:*

$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid t : \varphi, \text{ with } t ::= c \mid x \mid t_1 \cdot t_2 \mid t_1 + t_2 \mid !t ,$$

where $p \in \Phi$, $c \in \mathcal{C}$ and $x \in \mathcal{X}$. We denote the set of all evidence terms of the language by \mathcal{T} and the set of all formulas by F .

In this logic and in every other logic described in this paper, we use the standard abbreviations $\perp \equiv \neg\top$, $\varphi \vee \phi \equiv \neg(\neg\varphi \wedge \neg\phi)$, $\varphi \rightarrow \phi \equiv \neg(\varphi \wedge \neg\phi)$ and $\varphi \leftrightarrow \phi \equiv (\varphi \rightarrow \phi) \wedge (\phi \rightarrow \varphi)$.

Initially, the basic Justification Logic was defined in a purely syntactic manner, through an axiomatic system. Later, Fitting [12] provided a modal semantics for the logic.

Definition 2. *A frame for Justification Logic is a tuple $\mathcal{F} = (W, R)$ where W is a non-empty set of states and $R \subseteq W \times W$ is a binary relation that is reflexive and transitive.*

Definition 3. *A Fitting Model for Justification Logic is a tuple $\mathcal{M} = (\mathcal{F}, \mathbf{V}, \mathcal{E})$, where \mathcal{F} is a frame, \mathbf{V} is a valuation function $\mathbf{V} : \Phi \mapsto 2^W$ and \mathcal{E} is an evidence function $\mathcal{E} : W \times \mathcal{T} \rightarrow F$ satisfying the following conditions:*

- If $(\varphi \rightarrow \psi) \in \mathcal{E}(w, s)$ and $\varphi \in \mathcal{E}(w, t)$, then $\psi \in \mathcal{E}(w, s \cdot t)$.
- $\mathcal{E}(w, s) \cup \mathcal{E}(w, t) \subseteq \mathcal{E}(w, s + t)$.
- If $\varphi \in \mathcal{E}(w, t)$, then $t : \varphi \in \mathcal{E}(w, !t)$.
- If wRw' , then $\mathcal{E}(w, t) \subseteq \mathcal{E}(w', t)$.
- If $\varphi \in \mathcal{E}(w, c)$ and $c \in \mathcal{C}$, then φ must be valid, as defined below.

Definition 4. *Let $\mathcal{M} = (\mathcal{F}, \mathbf{V}, \mathcal{E})$ be a Fitting model. The notion of satisfaction of a formula φ in a model \mathcal{M} at a state w , notation $\mathcal{M}, w \Vdash \varphi$, can be inductively defined as follows:*

- $\mathcal{M}, w \Vdash p$ iff $w \in \mathbf{V}(p)$.
- $\mathcal{M}, w \Vdash \top$ always.
- $\mathcal{M}, w \Vdash \neg\varphi$ iff $\mathcal{M}, w \not\Vdash \varphi$.
- $\mathcal{M}, w \Vdash \varphi_1 \wedge \varphi_2$ iff $\mathcal{M}, w \Vdash \varphi_1$ and $\mathcal{M}, w \Vdash \varphi_2$.
- $\mathcal{M}, w \Vdash t : \varphi$ iff $\varphi \in \mathcal{E}(w, t)$ and, for all w' such that wRw' , $\mathcal{M}, w' \Vdash \varphi$.

If $\mathcal{M}, w \Vdash \varphi$ for every state w , we say that φ is *globally satisfied* in the model \mathcal{M} , notation $\mathcal{M} \Vdash \varphi$. If φ is globally satisfied in all models \mathcal{M} of a frame \mathcal{F} , we say that φ is *valid* in \mathcal{F} , notation $\mathcal{F} \Vdash \varphi$. Finally, if φ is valid in all frames, we say that φ is *valid*, notation $\Vdash \varphi$.

From the semantical definition above, we can think of \cdot as a form of evidence-controlled Modus Ponens, $+$ as a form of evidence combination, $!$ as a constructor of evidence for formulas that already contain evidence terms and evidence constants as atomic evidence for formulas that do not need further justification (since they are valid).

2.2 Plausibility Models

In this section, we present Plausibility Models for the single-agent case. The multi-agent case is covered in the presentation of our logic in the next section. Plausibility Models in the present form were introduced in [4, 5] and [7, 8].

Definition 5. A Plausibility Frame is a tuple $\mathcal{F} = (W, \geq)$, where W is a non-empty set of states and $\geq \subseteq W \times W$ is a relation that satisfies reflexivity, transitivity (thus, is a pre-order) and local connectivity (for all $v, w \in W$, $v \geq w$ or $w \geq v$ or both). \geq is called a plausibility order.

When we think about the relation \geq as an epistemic relation, we consider that, if $v \geq w$, then the agent does not know for sure in which of the states v or w he actually is, but he considers that the state w is *more* plausible than the state v . The choice of the most plausible states as the minimal states according to the pre-order \geq , which seems counter-intuitive, comes from the fact that if we add the hypothesis that \geq is well-founded, then we guarantee that the set of most plausible states is always well-defined.

As \geq is transitive and locally connected, the agent considers that it is possible for him to be in any state of the model, but he considers some more plausible than others. In the particular case that $v \geq w$ and $w \geq v$, the agent considers both states v and w to be equally plausible.

Definition 6. A Plausibility Model is a tuple $\mathcal{M} = (\mathcal{F}, \mathbf{V})$, where \mathcal{F} is a Plausibility Frame and \mathbf{V} is a valuation function $\mathbf{V} : \Phi \rightarrow 2^W$, mapping proposition symbols to sets of states.

Let us now discuss the kinds of *belief* that can be described in a Plausibility Model. One thing that all sorts of beliefs have in common, and what differentiates them from *knowledge*, is that there is always the possibility that a belief can be false. Nevertheless, beliefs are *consistent*, which means that an agent cannot simultaneously believe in ϕ and $\neg\phi$.

For our discussion of beliefs, let us consider that the model \mathcal{M} is finite or that the relation \geq is well-founded. We can define the set $\text{Best}(\mathcal{M}) = \{w \in W : v \geq w, \text{ for all } v \in W\}$. Then, we can define the weakest notion of belief (denoted by \mathcal{B}) as $\mathcal{M}, w \Vdash \mathcal{B}\varphi$ iff $\mathcal{M}, v \Vdash \varphi$, for all $v \in \text{Best}(\mathcal{M})$. Thus, an agent believes in φ if the formula is satisfied in the most plausible states, according to his plausibility order. The modality \mathcal{B} is a normal modality ($\mathcal{B}(\varphi \rightarrow \psi) \rightarrow (\mathcal{B}\varphi \rightarrow \mathcal{B}\psi)$). However, \mathcal{B} is not the modality directly associated with the relation \geq . Let us then define $\mathcal{M}, w \Vdash \Box\varphi$ iff $\mathcal{M}, v \Vdash \varphi$, for all v such that $w \geq v$. The notion described by \Box is called *safe belief*. An agent has safe belief in a formula if it is satisfied in all states that are more or equally plausible than the current one. Thus, safe belief implies belief. Safe belief is also normal. However, the “safety” of a belief can only be inferred by an external observer, because for an agent to know that one of his beliefs is safe he would need to know in which state he currently is. Finally, we define the notion of *strong belief* in a formula φ if all the states in which φ is satisfied are more plausible than all the states in which φ is not satisfied. Strong belief also implies belief, but strong belief is not normal.

3 Logic of Plausible Justifications

In this section, we present our logic, discuss some of its expressive features and present a sound and strongly complete axiomatic system for it.

3.1 Language and Semantics

We start by defining the language for the formulas of our logic.

Definition 7. *In order to define the language of LPJ, we need to take a finite set $\mathcal{A} = \{1, \dots, n\}$ of agents, a countable set Φ of proposition symbols and countable sets \mathcal{X}_i , $i \in \mathcal{A}$, of evidence variables. We assume that each pair of such sets is disjoint. The formulas φ and the evidence terms t of the language are defined as follows:*

$$\begin{aligned} \varphi ::= p \mid \top \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \mathcal{K}_i\varphi \mid \Box_i\varphi \mid [t?]\varphi \mid t \gg_i \varphi \mid \mathcal{P}_i t \mid t :_i \varphi, \text{ with} \\ t ::= p \mid g \mid x_j^i \mid \bar{t}, \mid t_1 + t_2, \end{aligned}$$

where $p \in \Phi$, $i \in \mathcal{A}$, $j \in \mathbb{N}$, $x_j^i \in \mathcal{X}_i$ and g is a term not occurring in \mathcal{A} , Φ or any of the sets \mathcal{X}_i . We denote the set of all evidence terms of the language by \mathcal{T} .

In the rest of this paper, sometimes it is convenient to use the duals of some of our modalities: $\langle \mathcal{K}_i \rangle \varphi \equiv \neg \mathcal{K}_i \neg \varphi$, $\langle \Box_i \rangle \varphi \equiv \neg \Box_i \neg \varphi$ and $\langle t? \rangle \varphi \equiv \neg [t?] \neg \varphi$. We do not use the duals of \gg_i , \mathcal{P}_i and $:_i$, but they can be defined analogously.

The first thing that we notice is that there are some differences between our evidence terms and the ones in Justification Logic. Our language does not have the operators \cdot and $!$, it does not have a set \mathcal{C} of evidence constants, having instead a single evidence constant g , it has evidence terms of the form \bar{t} and proposition symbols are also evidence terms. We will discuss these differences after we present the semantics of our logic.

Definition 8. *A frame for LPJ is a tuple $\mathcal{F} = (W, \{\sim_i\}_{i \in \mathcal{A}}, \{\geq_i\}_{i \in \mathcal{A}})$ where*

- W is a non-empty set of states
- $\sim_i \subseteq W \times W$ is an equivalence relation.
- $\geq_i \subseteq W \times W$ is a relation that satisfies reflexivity and transitivity.
- For each $i \in \mathcal{A}$, the relations \sim_i and \geq_i satisfy the following property: $\sim_i = \geq_i \cup (\geq_i)^{-1}$, where $(\geq_i)^{-1} = \{(v, w) \in W \times W : (w, v) \in \geq_i\}$.
- Building the relation $\approx = (\bigcup_{i \in \mathcal{A}} \sim_i)$, we have that, for every pair $(v, w) \in W \times W$, $v \approx w$. We call this property weak connectivity.

In an LPJ frame, the relations \geq_i are the plausibility relations for each of the agents. They are pre-orders, just as in the single-agent setting of the previous section. Also as in the previous section, if $v \geq_i w$, then agent i does not know for sure in which of the states v or w he actually is, but he considers state w more plausible than state v . The relations \sim_i denote these relations of indistinguishability of states by each of the agents. This is why they are defined as $\sim_i = \geq_i \cup (\geq_i)^{-1}$.

The relations \geq_i are no longer locally connected in our present multi-agent scenario. However, the property of weak connectivity implies that every pair of states in the frame is indistinguishable to at least one of the agents.

Definition 9. A model for LPJ is a tuple $\mathcal{M} = (\mathcal{F}, \mathbf{V}, \mathcal{E})$, where \mathcal{F} is a frame, \mathbf{V} is a valuation function $\mathbf{V} : \Phi \rightarrow 2^W$, mapping proposition symbols to sets of states, and \mathcal{E} is an evidence function $\mathcal{E} : \mathcal{T} \rightarrow 2^W$, mapping evidence terms to sets of states, that satisfies the following rules:

- $\mathcal{E}(p) = \mathbf{V}(p)$, for all $p \in \Phi$.
- $\mathcal{E}(g) = W$.
- $\mathcal{E}(\bar{t}) = W \setminus \mathcal{E}(t)$.
- $\mathcal{E}(t_1 + t_2) = \mathcal{E}(t_1) \cap \mathcal{E}(t_2)$.

We can see, in this definition, how the semantics for our evidence terms is built. We follow a similar approach to the one presented in [9] and use what seems to be the simplest semantics for evidence: an evidence is a subset of states of the model. Roughly speaking, we say that an evidence term t is an evidence for a formula φ if φ is satisfied in all of the states in $\mathcal{E}(t)$ (in reality, we also have to check the *plausibility* of the evidence, as we discuss below).

As the proposition symbols also denote subsets of states (using the function \mathbf{V}), we also use them as evidence terms. We can think of the evidences denoted by proposition symbols as the “common ground” for all of the agents. Then, we can use the variables in the sets \mathcal{X}_i to denote the “personal views” of each agent. The evidence term g can be considered as the “weakest” evidence, as it contains all the states in the model. It has a similar function to the evidence constants in Justification Logic (looking at the semantics below, we can see that g can be used by any agent as evidence for what he *knows*) and it can also replace the operator $!$, as we can see, from the definition below, that the formula $t :_i \varphi \rightarrow g :_i (t :_i \varphi)$ is valid. We can drop the operator \cdot since, in our semantics, it would be a particular case of the operator $+$ to formulas in implication form. Finally, evidence terms of the form \bar{t} denote evidence *complementation*.

Definition 10. Let $\mathcal{M} = (\mathcal{F}, \mathbf{V}, \mathcal{E})$ be a model. The notion of satisfaction of a formula φ in a model \mathcal{M} at a state w , notation $\mathcal{M}, w \Vdash \varphi$, can be inductively defined as follows:

- $\mathcal{M}, w \Vdash p$ iff $w \in \mathbf{V}(p)$.
- $\mathcal{M}, w \Vdash \top$ always.
- $\mathcal{M}, w \Vdash \neg\varphi$ iff $\mathcal{M}, w \not\Vdash \varphi$.
- $\mathcal{M}, w \Vdash \varphi_1 \wedge \varphi_2$ iff $\mathcal{M}, w \Vdash \varphi_1$ and $\mathcal{M}, w \Vdash \varphi_2$.
- $\mathcal{M}, w \Vdash \mathcal{K}_i\varphi$ iff for all $v \in W$ such that $w \sim_i v$, $\mathcal{M}, v \Vdash \varphi$.
- $\mathcal{M}, w \Vdash \Box_i\varphi$ iff for all $v \in W$ such that $w \geq_i v$, $\mathcal{M}, v \Vdash \varphi$.
- $\mathcal{M}, w \Vdash [t?]\varphi$ iff if $w \in \mathcal{E}(t)$, then $\mathcal{M}, w \Vdash \varphi$.
- $\mathcal{M}, w \Vdash t \gg_i \varphi$ iff for all $v \in W$ such that $w \sim_i v$ and $v \in \mathcal{E}(t)$, $\mathcal{M}, v \Vdash \varphi$.
- $\mathcal{M}, w \Vdash \mathcal{P}_i t$ iff:

1. there is $v \in W$ such that $w \sim_i v$ and $v \in \mathcal{E}(t)$ and
 2. for all $x, y \in W$ such that $w \sim_i x$ and $x \geq_i y$, if $x \in \mathcal{E}(t)$, then $y \in \mathcal{E}(t)$.
- $\mathcal{M}, w \Vdash t :_i \varphi$ iff $\mathcal{M}, w \Vdash t \gg_i \varphi$ and $\mathcal{M}, w \Vdash \mathcal{P}_i t$.

The notions of global satisfaction, validity in a frame and validity are defined as in the previous section. We say that a formula φ is *satisfiable* if there is a model \mathcal{M} and a state w in \mathcal{M} such that $\mathcal{M}, w \Vdash \varphi$. A formula is not satisfiable iff its negation is valid. A (possibly infinite) set Δ of formulas is satisfiable if there is a *single model* \mathcal{M} and a *single state* w in \mathcal{M} such that $\mathcal{M}, w \Vdash \varphi$, for all $\varphi \in \Delta$.

Our modalities $\mathcal{K}_i, \Box_i, [t?], \gg_i$ and $:_i$ are all normal. The modalities \mathcal{K}_i and \Box_i denote the usual notions of *knowledge* (satisfaction in all states indistinguishable from the current one) and *safe belief* (satisfaction in all states more or equally plausible than the current one), respectively.

Our modalities $[t?]$ are inspired by PDL [14] test modalities. They allow us to verify whether a state belongs to an evidence t , by checking whether $\langle t? \rangle \top$ is satisfied at a state. The relation associated to the modality $[t?]$ is $R_t = \{(w, w) : w \in \mathcal{E}(t)\}$. The modalities \gg_i are inspired by Renne’s [16–18] modality of *admissibility*. The semantics of a formula $t : \varphi$ in Justification Logic is composed by two parts (see Section 2.1): the first one related to the evidence function and the second to the relations in the frame. Renne calls the first part *admissibility* and uses the modality \gg to describe it. In our semantics, an agent i considers an evidence t *admissible* for a formula φ ($t \gg_i \varphi$) if, in all the states inside the evidence t that the agent consider possible for him to be, the formula φ is satisfied.

The modalities \mathcal{P}_i are used to denote that agent i considers an evidence to be *plausible*. The notion of *plausibility* of an evidence has some similarities to the notion of *strong belief*. An evidence is considered plausible if there is a state that the agent considers possible inside of the evidence and, among the states that the agent considers possible, all of the states inside the evidence are more plausible than all of the states outside the evidence. Finally, the modality $:_i$ is used to denote that an agent considers an evidence as *plausible evidence* or *plausible justification* for a formula. An evidence is plausible evidence for a formula if the agent considers the evidence to be plausible and considers the evidence to be admissible for the formula.

Theorem 1 (Finite Model Property). *Every satisfiable formula φ is satisfiable in a finite model (i.e., a model with a finite number of states).*

Proof. See appendix A. □

Corollary 1 (Decidability). *The satisfiability problem for LPJ (determining whether a formula φ is satisfiable) is decidable.*

Proof. The proof of Theorem 1 gives an upper bound (as a function of the size of φ) for the size of the models where φ must be checked. As there is a finite number of such models, they can all be verified to determine whether φ is satisfiable. □

3.2 Axiomatic System

We consider the set of axioms and rules in Figure 1, where φ and ψ are arbitrary formulas, t and s are arbitrary evidence terms and p is an arbitrary proposition symbol. We present the axioms divided in groups related to their function.

- | | |
|---|--|
| <p>1. Tautologies, Duals and Normality</p> <p>PL Propositional Tautologies</p> <p>Du_K $\mathcal{K}_i\varphi \leftrightarrow \neg\langle\mathcal{K}_i\rangle\neg\varphi$</p> <p>Du_□ $\Box_i\varphi \leftrightarrow \neg\Diamond_i\neg\varphi$</p> <p>Du_t $[t?]\varphi \leftrightarrow \neg\langle t?\rangle\neg\varphi$</p> <p>K_K $\mathcal{K}_i(\varphi \rightarrow \psi) \rightarrow (\mathcal{K}_i\varphi \rightarrow \mathcal{K}_i\psi)$</p> <p>K_□ $\Box_i(\varphi \rightarrow \psi) \rightarrow (\Box_i\varphi \rightarrow \Box_i\psi)$</p> <p>K_t $[t?](\varphi \rightarrow \psi) \rightarrow ([t?]\varphi \rightarrow [t?]\psi)$</p> <p>2. \sim_i is an equivalence relation</p> <p>T_K $\mathcal{K}_i\varphi \rightarrow \varphi$</p> <p>4_K $\mathcal{K}_i\varphi \rightarrow \mathcal{K}_i\mathcal{K}_i\varphi$</p> <p>5_K $\neg\mathcal{K}_i\varphi \rightarrow \mathcal{K}_i\neg\mathcal{K}_i\varphi$</p> <p>3. \geq_i is reflexive and transitive</p> <p>T_□ $\Box_i\varphi \rightarrow \varphi$</p> <p>4_□ $\Box_i\varphi \rightarrow \Box_i\Box_i\varphi$</p> <p>4. $\sim_i = \geq_i \cup (\geq_i)^{-1}$</p> <p>Rel₁ $\mathcal{K}_i\varphi \rightarrow \Box_i\varphi$</p> <p>Rel₂ $\langle\mathcal{K}_i\rangle\varphi \wedge \langle\mathcal{K}_i\rangle\psi \rightarrow \langle\mathcal{K}_i\rangle(\varphi \wedge \Diamond\psi) \vee \langle\mathcal{K}_i\rangle(\Diamond\varphi \wedge \psi)$</p> | <p>5. Construction of evidence terms</p> <p>E₁ $\langle t?\rangle\varphi \leftrightarrow ((\langle t?\rangle\top) \wedge \varphi)$</p> <p>E₂ $\langle p?\rangle\top \leftrightarrow p$</p> <p>E₃ $\langle g?\rangle\top$</p> <p>E₄ $\langle \bar{t}?\rangle\top \leftrightarrow \neg\langle t?\rangle\top$</p> <p>E₅ $((\langle t?\rangle\top) \wedge \langle s?\rangle\top) \leftrightarrow \langle (s+t)?\rangle\top$</p> <p>6. Admissibility, Plausibility and Justification</p> <p>Adm $(t \gg_i \varphi) \leftrightarrow (\mathcal{K}_i[t?]\varphi)$</p> <p>Pla $\mathcal{P}_i t \leftrightarrow ((\mathcal{K}_i[t?]\Box_i\langle t?\rangle\top) \wedge (\langle\mathcal{K}_i\rangle\langle t?\rangle\top))$</p> <p>Jus $t :_i \varphi \leftrightarrow (t \gg_i \varphi) \wedge \mathcal{P}_i t$</p> <p>7. Rules</p> <p>MP From $\varphi \rightarrow \psi$ and φ, derive ψ</p> <p>Gen From φ, derive $\mathcal{K}_i\varphi$, $\Box_i\varphi$ and $[t?]\varphi$</p> |
|---|--|

Fig. 1. Axiomatic System

From the axioms above, perhaps **Rel₂** is the one with the less clear purpose. **Rel₁** states that every pair in \geq_i is also in \sim_i , while **Rel₂** states that every pair in \sim_i is in \geq_i or in $(\geq_i)^{-1}$. **Rel₂** is an adaptation with two modalities ($\langle\mathcal{K}_i\rangle$ and \Diamond_i) of the so-called **.3** axiom (see [10] for more details about this axiom).

Every formula ϕ derivable from the axiomatic system above is called a *theorem* (denoted by $\vdash \phi$). A formula ϕ is *consistent* iff $\neg\phi$ is not a theorem, i.e., iff $\not\vdash \neg\phi$, and *inconsistent* otherwise. A finite set of formulas $\Delta = \{\phi_1, \dots, \phi_n\}$ is consistent iff the formula $\psi = \phi_1 \wedge \dots \wedge \phi_n$ is consistent. Finally, an infinite set of formulas Δ' is consistent iff every finite subset $\Delta \subset \Delta'$ is consistent.

The axiomatic system is said to be *sound* if every satisfiable formula is consistent (or, in an equivalent definition, if every satisfiable set of formulas is consistent). The axiomatic system is said to be *complete* if every consistent formula is satisfiable. It is said to be *strongly complete* if every consistent set of formulas is satisfiable. Unlike the case of soundness, the two definitions for completeness are not equivalent. Strong completeness implies completeness, but the reciprocal is false.

The proof of the soundness of our axiomatic system is straightforward. It is not difficult to show that each of the axioms is valid according to the LPJ semantics and that the application of each of the rules to valid formulas give formulas that are also valid. The strong completeness proof is given in the following theorem.

Theorem 2 (Strong Completeness). *Every consistent set of formulas is satisfiable in an LPJ model.*

Proof. See appendix [B](#). □

4 Final Remarks and Future Work

In this work, we combine aspects of Justification Logics with Plausibility Models to build a logic of *explicit* beliefs, where each agent can explicitly state which is his justification for believing in a given sentence. Our logic is a normal modal logic based on the standard Kripke semantics, where we provide a semantic definition for the evidence terms and define the notion of plausible evidence for an agent, based on plausibility relations in the model. In our logic, agents can disagree not only over whether a sentence is true or false, but also on whether some evidence is a valid justification for a sentence or not. Thus, unlike traditional Justification Logics, justifications can be faulty and unreliable in our logic. After defining our logic and its semantics, we provide a strongly complete axiomatic system for it and show that it has the finite model property and is decidable.

We feel that this logic is a good first step for the development of a dynamic logic that can model the processes of argumentation and debate in multi-agent systems. We think that the appropriate logic to model these processes would be a *dynamic* logic of evidence-based beliefs. Thus, as our next step to build such a logic, we need to add to the present logic the actions that would model the communications between agents during the processes of argumentation and debate. In our desired framework, the announcement of a sentence should be accompanied by a justification as to why the agent performing the announcement believes in it. Then, each agent receiving the announcement should judge by himself whether he should start believing or not in the announced sentence, based on his current beliefs both about what was announced and about the justification that was given. In a preliminary analysis, it seems that we may have a few possible results for an announcement, depending on whether the agent receiving the announcement currently (before the announcement) believes in what was announced, in the negation of it or in neither and whether he currently considers that the justification that was given is plausible or not.

Beside this main goal, we feel that it would also be interesting to develop other proof systems for this logic, such as a tableau system or a sequent calculus, since they are more suited to be used as automatic provers than an axiomatic system. It would also be interesting to investigate the model-checking problem for this logic and to analyze its complexity. Finally, it would be interesting to analyze extensions of this logic with quantification over evidence terms, as [\[13\]](#) did in the context of traditional Justification Logics.

References

1. Artemov, S.: Logic of proofs. *Annals of Pure and Applied Logic* 67(1-3), 29–59 (1994)
2. Artemov, S.: Justified common knowledge. *Theoretical Computer Science* 357(1-3), 4–22 (2006)
3. Artemov, S., Nogina, E.: Introducing justification into epistemic logic. *Journal of Logic and Computation* 15(6), 1059–1073 (2005)
4. Baltag, A., Smets, S.: Conditional doxastic models: A qualitative approach to dynamic belief revision. In: Queiroz, R., Mints, G. (eds.) *Proceedings of the 13th Workshop on Logic, Language, Information and Computation (WoLLIC 2006)*. *Electronic Notes in Theoretical Computer Science*, vol. 165, pp. 5–21. Elsevier, Amsterdam (2006)
5. Baltag, A., Smets, S.: Dynamic belief revision over multi-agent plausibility models. In: Bonano, G., van der Hoek, W., Wooldridge, M. (eds.) *Proceedings of the 7th Conference on Logic and the Foundations of Game and Decision Theory (LOFT 2006)*, pp. 11–24. Liverpool (2006)
6. Baltag, A., Smets, S.: Talking your way into agreement: Belief merge by persuasive communication. In: Baldoni, M., et al. (eds.) *Proceedings of the Second Multi-Agent Logics, Languages, and Organisations Federated Workshops. CEUR Workshop Proceedings*, vol. 494, pp. 129–141. CEUR-WS.org, Aachen (2009), <http://ceur-ws.org/Vol-494/>
7. van Benthem, J.: Dynamic logic for belief revision. *Journal of Applied Non-Classical Logics* 17(2), 129–155 (2007)
8. van Benthem, J., Liu, F.: Dynamic logic of preference upgrade. *Journal of Applied Non-Classical Logics* 17(2), 157–182 (2007)
9. van Benthem, J., Pacuit, E.: Dynamic logics of evidence-based beliefs. *Studia Logica* 99(1-3), 61–92 (2011)
10. Blackburn, P., de Rijke, M., Venema, Y.: *Modal Logic*. *Cambridge Tracts in Theoretical Computer Science*, vol. 53. Cambridge University Press, Cambridge (2001)
11. van Ditmarsch, H., van der Hoek, W., Kooi, B.: *Dynamic Epistemic Logic*. *Synthese Library*. Springer, Heidelberg (2007)
12. Fitting, M.: The logic of proofs, semantically. *Annals of Pure and Applied Logic* 132(1), 1–25 (2005)
13. Fitting, M.: A quantified logic of evidence. *Annals of Pure and Applied Logic* 152(1-3), 67–83 (2008)
14. Harel, D., Kozen, D., Tiuryn, J.: *Dynamic Logic*. *Foundations of Computing*. MIT Press, Cambridge (2000)
15. van der Hoek, W., Verbrugge, R.: Epistemic logic: a survey. In: Petrosjan, L.A., Mazalov, V.V. (eds.) *Game Theory and Applications*, vol. 8, pp. 53–94. Nova Science Publishers, New York (2002)
16. Renne, B.: *Dynamic Epistemic Logic with Justification*. Ph.D. thesis, The City University of New York (2008)
17. Renne, B.: Public communication in justification logic. *Journal of Logic and Computation* 21(6), 1005–1034 (2011)
18. Renne, B.: Multi-agent justification logic: Communication and evidence elimination. *Synthese* 185(S1), 43–82 (2012)
19. Yavorskaya, T.: Interacting explicit evidence systems. *Theory of Computing Systems* 43(2), 272–293 (2008)

A Finite Model Property

In this section, we present a proof of the finite model property for our logic using the standard technique of *filtrations*, adapting it to the particular features of our logic. For more details on this construction, [10] can be consulted.

Definition 11. *Given a formula φ of the language, we built the set of formulas $Cl(\varphi)$, called closure of φ , which is the smallest set that contains φ and satisfies the following properties:*

1. If $\psi \in Cl(\varphi)$, then, for all subformulas ψ' of ψ , $\psi' \in Cl(\varphi)$.
2. If $\blacksquare\psi \in Cl(\varphi)$, then $\neg\blacklozenge\neg\psi \in Cl(\varphi)$, where $\blacksquare \in \{\mathcal{K}_i, \square_i, [t?]\}$ and \blacklozenge is the corresponding dual.
3. If $t \gg_i \psi \in Cl(\varphi)$, then $\mathcal{K}_i[t?]\psi \in Cl(\varphi)$.
4. If $\mathcal{P}_i t \in Cl(\varphi)$, then $\mathcal{K}_i[t?]\square_i\langle t?\rangle\top \in Cl(\varphi)$ and $\langle \mathcal{K}_i \rangle \langle t?\rangle\top \in Cl(\varphi)$.
5. If $t :_i \psi \in Cl(\varphi)$, then $t \gg_i \psi \in Cl(\varphi)$ and $\mathcal{P}_i t \in Cl(\varphi)$.
6. If $\langle t?\rangle\psi \in Cl(\varphi)$, then $\langle t?\rangle\top \in Cl(\varphi)$.

It is important to notice that $Cl(\varphi)$ is a finite set. Using this set of formulas, we can define an equivalence relation in the set of states of an LPJ model $\mathcal{M} = (W, \{\sim_i\}_{i \in \mathcal{A}}, \{\geq_i\}_{i \in \mathcal{A}}, \mathbf{V}, \mathcal{E})$. We define $w \rightsquigarrow w'$ if, for all formulas $\psi \in Cl(\varphi)$, $\mathcal{M}, w \Vdash \psi$ iff $\mathcal{M}, w' \Vdash \psi$. We denote the equivalence class of state w by this equivalence relation as $|w|$. We can then use this equivalence relation to build a new model \mathcal{M}^f from \mathcal{M} .

Definition 12. *Let $\mathcal{M}^f = (W^f, \{\sim_i^f\}_{i \in \mathcal{A}}, \{\geq_i^f\}_{i \in \mathcal{A}}, \mathbf{V}^f, \mathcal{E}^f)$, where:*

- $W^f = \{|w| : w \in W\}$.
- $|w| \sim_i^f |v|$ iff, for all formulas $\langle \mathcal{K}_i \rangle \psi \in Cl(\varphi)$, $\mathcal{M}, w \Vdash \langle \mathcal{K}_i \rangle \psi$ iff $\mathcal{M}, v \Vdash \langle \mathcal{K}_i \rangle \psi$.
- $|w| \geq_i^f |v|$ iff, for all formulas $\diamond_i \psi \in Cl(\varphi)$, $\mathcal{M}, v \Vdash \diamond_i \psi$ implies that $\mathcal{M}, w \Vdash \diamond_i \psi$.
- $\mathbf{V}^f(p) = \{|w| \in W^f : w \in \mathbf{V}(p)\}$, for all proposition symbols $p \in Cl(\varphi)$.
- $\mathcal{E}^f(t) = \{|w| \in W^f : w \in \mathcal{E}(t)\}$, for all evidence terms t such that $\langle t?\rangle\top \in Cl(\varphi)$.

It is straightforward to verify that the relations \sim_i^f and \geq_i^f satisfy the necessary conditions stated in [10] for filtration relations and also to check that \mathcal{M}^f indeed satisfies all the properties of an LPJ model: \sim_i^f is an equivalence relation, \geq_i^f is reflexive and transitive, we have $\sim_i^f = \geq_i^f \cup (\geq_i^f)^{-1}$, \mathcal{M}^f is weakly connected and the evidence function satisfies the desired inductive rules. We call the model \mathcal{M}^f the filtration of \mathcal{M} through $Cl(\varphi)$. It is important to notice that, if $|Cl(\varphi)| = k$, then $|W^f| = 2^k$, so \mathcal{M}^f is a *finite model*.

Theorem 3 (Filtration Theorem). *For all formulas $\psi \in Cl(\varphi)$ and all states w in \mathcal{M} , $\mathcal{M}, w \Vdash \psi$ iff $\mathcal{M}^f, |w| \Vdash \psi$.*

Proof. The proof is by induction on the structure of the formula ψ .

- If ψ is a proposition symbol, $\psi = \top$, $\psi = \neg\psi_1$ or $\psi = \psi_1 \wedge \psi_2$, the proof is straightforward from the definition of \mathbf{V}^f and from item [11](#) of definition [11](#).
- If $\psi = \mathcal{K}_i\psi_1$, then, by items [11](#) and [12](#) of definition [11](#), $\phi = \langle \mathcal{K}_i \rangle \phi_1 \in Cl(\varphi)$, where $\phi_1 = \neg\psi_1$. Then, using the induction step for negation, it is sufficient to show the result for ϕ .
 (\Rightarrow) Suppose that $\mathcal{M}, w \Vdash \langle \mathcal{K}_i \rangle \phi_1 \in \Gamma$. Then, there is v in \mathcal{M} such that $w \sim_i v$ and $\mathcal{M}, v \Vdash \phi_1$. By the induction hypothesis, $\mathcal{M}^f, |v| \Vdash \phi_1$. Now, as \sim_i^f satisfies the necessary conditions for filtration relations, we have that $w \sim_i v$ implies $|w| \sim_i^f |v|$. Thus, $\mathcal{M}^f, |w| \Vdash \langle \mathcal{K}_i \rangle \phi_1$.
 (\Leftarrow) Suppose that $\mathcal{M}^f, |w| \Vdash \langle \mathcal{K}_i \rangle \phi_1$. Then, there exists $|v|$ in \mathcal{M}^f such that $|w| \sim_i^f |v|$ and $\mathcal{M}, |v| \Vdash \phi_1$. By the induction hypothesis, $\mathcal{M}, v \Vdash \phi_1$. Now, as \sim_i^f satisfies the necessary conditions for filtration relations, we have that $|w| \sim_i^f |v|$ and $\mathcal{M}, v \Vdash \phi_1$ implies that $\mathcal{M}, w \Vdash \langle \mathcal{K}_i \rangle \phi_1$.
- If $\psi = \Box_i\psi_1$, the proof is entirely analogous to the previous item.
- If $\psi = [t?]\psi_1$, then, using the same reasoning of the previous two cases, it is sufficient to show the result for formulas of the form $\phi = \langle t? \rangle \phi_1$. Now, $\mathcal{M}, w \Vdash \langle t? \rangle \phi_1$ iff $w \in \mathcal{E}(t)$ and $\mathcal{M}, w \Vdash \phi_1$. Then, by the induction hypothesis, item [6](#) of definition [11](#) and the definition of $\mathcal{E}^f(t)$, this happens iff $|w| \in \mathcal{E}^f(t)$ and $\mathcal{M}^f, |w| \Vdash \phi_1$ iff $\mathcal{M}^f, |w| \Vdash \langle t? \rangle \phi_1$.
- If $\psi = t \gg_i \psi_1$, $\psi = \mathcal{P}_i t$ or $\psi = t :_i \psi_1$, the proof is straightforward from the previous cases and items [3](#), [4](#) and [5](#) of the definition [11](#), respectively. \square

Theorem 4 (Finite Model Property). *Every satisfiable formula φ is satisfiable in a finite model.*

Proof. If $\mathcal{M}, w \Vdash \varphi$, we build the set $Cl(\varphi)$ and the finite model \mathcal{M}^f , which is the filtration of \mathcal{M} through $Cl(\varphi)$. Then, by the Filtration Theorem, we have $\mathcal{M}^f, |w| \Vdash \varphi$, so φ is satisfiable in a finite model. \square

B Strong Completeness

In this section, we present a strong completeness proof for our axiomatization using the standard technique of the construction of *canonical models*, adapting it to the particular features of our logic. For more details on this construction, [10](#) can be consulted.

Given a consistent set of formulas Δ , we can expand this set using the standard Lindenbaum construction [10](#) to obtain a maximal consistent set (MCS) $\Delta^+ \supseteq \Delta$. A set Γ is a MCS if it is consistent and all sets $\Gamma' \supsetneq \Gamma$ are inconsistent. Any MCS Γ has the following important property: for every formula φ in the language, $\varphi \in \Gamma$ iff $\neg\varphi \notin \Gamma$.

Definition 13 (Canonical Pre-Model). *The canonical LPJ pre-model is the tuple $\mathcal{M}^C = (W^C, \{\sim_i^C\}_{i \in \mathcal{A}}, \{\geq_i^C\}_{i \in \mathcal{A}}, \mathbf{V}^C, \mathcal{E}^C)$, where:*

- W^C is the set of all MCS's.
- If $\Gamma, \Gamma' \in W^C$, then $\Gamma \sim_i^C \Gamma'$ iff, for all formulas $\psi \in \Gamma'$, $\Gamma \cup \{\langle \mathcal{K}_i \rangle \psi\}$ is consistent (or, in an equivalent definition, $\langle \mathcal{K}_i \rangle \psi \in \Gamma$, since Γ is a MCS).
- If $\Gamma, \Gamma' \in W^C$, then $\Gamma \geq_i^C \Gamma'$ iff, for all formulas $\psi \in \Gamma'$, $\Gamma \cup \{\diamond_i \psi\}$ is consistent ($\diamond_i \psi \in \Gamma$).
- $\mathbf{V}^C(p) = \{\Gamma \in W^C : p \in \Gamma\}$, for all $p \in \Phi$.
- $\mathcal{E}^C(t) = \{\Gamma \in W^C : \langle t? \rangle \top \in \Gamma\}$, for all $t \in \mathcal{T}$.

Lemma 1 (Sound Evidence Lemma). *The canonical evidence function \mathcal{E}^C as defined above satisfies all of the properties required of an evidence function in an LPJ model (definition [9](#)).*

Proof

- $\Gamma \in \mathcal{E}^C(p)$ iff $\langle p? \rangle \top \in \Gamma$ iff, by axiom **E**₂, $p \in \Gamma$ iff $\Gamma \in \mathbf{V}^C(p)$, so $\mathcal{E}^C(p) = \mathbf{V}^C(p)$, for all $p \in \Phi$.
- $\Gamma \in \mathcal{E}^C(g)$ iff $\langle g? \rangle \top \in \Gamma$, which is always the case by axiom **E**₃, so $\mathcal{E}^C(g) = W^C$.
- $\Gamma \in \mathcal{E}^C(t_1 + t_2)$ iff $\langle (t_1 + t_2)? \rangle \top \in \Gamma$ iff, by axiom **E**₅, $\langle t_1? \rangle \top \in \Gamma$ and $\langle t_2? \rangle \top \in \Gamma$ iff $\Gamma \in \mathcal{E}^C(t_1)$ and $\Gamma \in \mathcal{E}^C(t_2)$, so $\mathcal{E}^C(t_1 + t_2) = \mathcal{E}^C(t_1) \cap \mathcal{E}^C(t_2)$.
- $\Gamma \in \mathcal{E}^C(\bar{t})$ iff $\langle \bar{t}? \rangle \top \in \Gamma$ iff, by axiom **E**₄, $\neg \langle t? \rangle \top \in \Gamma$ iff $\langle t? \rangle \top \notin \Gamma$ iff $\Gamma \notin \mathcal{E}^C(t)$, so $\mathcal{E}^C(\bar{t}) = W^C \setminus \mathcal{E}^C(t)$. \square

It is straightforward to check that \mathcal{M}^C satisfies many of the properties of an LPJ model: \sim_i^C is an equivalence relation (by axioms **T** _{\mathcal{K}} , **4** _{\mathcal{K}} and **5** _{\mathcal{K}}), \geq_i^C is reflexive (by axiom **T** _{\square}) and transitive (by axiom **4** _{\square}), we have $\sim_i^C = \geq_i^C \cup (\geq_i^C)^{-1}$ (by axioms **Rel**₁ and **Rel**₂) and the evidence function satisfies the desired inductive rules (Lemma [11](#)). Nevertheless, \mathcal{M}^C is not weakly connected. That is why we call \mathcal{M}^C the canonical *pre-model*. So, in order to build an LPJ model useful for proving that the consistent set Δ is satisfiable, we take the MCS Δ^+ and build the set W^{Δ^+} as the maximal weakly connected subset of W^C that contains Δ^+ . We then define $\sim_i^{\Delta^+}$, $\geq_i^{\Delta^+}$, \mathbf{V}^{Δ^+} and \mathcal{E}^{Δ^+} as the restrictions of \sim_i^C , \geq_i^C , \mathbf{V}^C and \mathcal{E}^C to the states in W^{Δ^+} , respectively. Then, it is straightforward to verify that $\mathcal{M}^{\Delta^+} = (W^{\Delta^+}, \{\sim_i^{\Delta^+}\}_{i \in \mathcal{A}}, \{\geq_i^{\Delta^+}\}_{i \in \mathcal{A}}, \mathbf{V}^{\Delta^+}, \mathcal{E}^{\Delta^+})$ satisfies all of the above properties that \mathcal{M}^C satisfies and it is also weakly connected by construction. Thus, \mathcal{M}^{Δ^+} is an LPJ model, and we call it the *canonical model* for the consistent set Δ .

Lemma 2 (Existence Lemma). *Let Γ be a MCS in \mathcal{M}^{Δ^+} . If the formula $\langle \mathcal{K}_i \rangle \psi$ belongs to Γ , then there is a MCS Γ' such that $\Gamma \sim_i^{\Delta^+} \Gamma'$ and $\psi \in \Gamma'$. Analogously, if the formula $\diamond_i \psi$ belongs to Γ , then there is a MCS Γ'' such that $\Gamma \geq_i^{\Delta^+} \Gamma''$ and $\psi \in \Gamma''$.*

Proof. We show the proof for the $\langle \mathcal{K}_i \rangle$ modality. The proof for the modality \diamond_i is entirely analogous. Suppose that $\langle \mathcal{K}_i \rangle \psi \in \Gamma$. We want to build a MCS Γ' such that $\psi \in \Gamma'$ and $\Gamma \sim_i^{\Delta^+} \Gamma'$. If $\neg \langle \mathcal{K}_i \rangle \varphi \in \Gamma$ and $\varphi \in \Gamma'$, then we would not have $\Gamma \sim_i^{\Delta^+} \Gamma'$, since Γ is a MCS. So, for all formulas $\neg \langle \mathcal{K}_i \rangle \varphi \in \Gamma$, we must have $\neg \varphi \in \Gamma'$. We can use this fact to build the set $\Sigma = \{\psi\} \cup \{\neg \varphi : \neg \langle \mathcal{K}_i \rangle \varphi \in \Gamma\}$. If Σ were not consistent, we would have $\{\neg \varphi_1, \dots, \neg \varphi_k\} \subset (\Sigma \setminus \{\psi\})$, where $\vdash \neg(\psi \wedge \neg \varphi_1 \wedge \dots \wedge \neg \varphi_k)$. Using propositional reasoning, we get $\vdash \psi \rightarrow (\varphi_1 \vee \dots \vee \varphi_k)$. Using axioms **K_K** and **Du_K**, rules **MP** and **Gen** and propositional reasoning, we get $\vdash \langle \mathcal{K}_i \rangle \psi \rightarrow (\langle \mathcal{K}_i \rangle \varphi_1 \vee \dots \vee \langle \mathcal{K}_i \rangle \varphi_k)$, which contradicts the consistency of Γ . Then, Σ is consistent and can be expanded to a MCS Σ^+ . We can then take $\Gamma' = \Sigma^+$. \square

Lemma 3 (Truth Lemma). *Let \mathcal{M}^{Δ^+} be the canonical model for the set Δ^+ . For all MCS's Γ in the model \mathcal{M}^{Δ^+} and all formulas φ in the language, $\mathcal{M}^{\Delta^+}, \Gamma \Vdash \varphi$ iff $\varphi \in \Gamma$.*

Proof. The proof is by induction on the structure of the formula φ . For the rest of this proof, Γ will denote a MCS in \mathcal{M}^{Δ^+} .

- If $\varphi = p \in \Phi$, $\varphi = \top$, $\varphi = \neg \psi$ or $\varphi = \varphi_1 \wedge \varphi_2$, the proof is straightforward from the definitions of **V^C** and **V^{Δ⁺}** and from the definition of a MCS.
- If $\varphi = \mathcal{K}_i \psi$, then, by the induction step for negation and axiom **Du_K**, it is sufficient to show the result for formulas of the form $\phi = \langle \mathcal{K}_i \rangle \phi_1$.
 (\Rightarrow) Suppose that $\langle \mathcal{K}_i \rangle \phi_1 \in \Gamma$. Then, by the Existence Lemma, there is Γ' in \mathcal{M}^{Δ^+} such that $\Gamma \sim_i^{\Delta^+} \Gamma'$ and $\phi_1 \in \Gamma'$. By the induction hypothesis, $\mathcal{M}^{\Delta^+}, \Gamma' \Vdash \phi_1$, which implies $\mathcal{M}^{\Delta^+}, \Gamma \Vdash \langle \mathcal{K}_i \rangle \phi_1$.
 (\Leftarrow) Suppose that $\mathcal{M}^{\Delta^+}, \Gamma \Vdash \langle \mathcal{K}_i \rangle \phi_1$. Then, there is Γ' in \mathcal{M}^{Δ^+} such that $\Gamma \sim_i^{\Delta^+} \Gamma'$ and $\mathcal{M}^{\Delta^+}, \Gamma' \Vdash \phi_1$. By the induction hypothesis, $\phi_1 \in \Gamma'$ and, by the definitions of \sim_i^C and $\sim_i^{\Delta^+}$, we must have $\langle \mathcal{K}_i \rangle \phi_1 \in \Gamma$.
- If $\varphi = \Box_i \psi$, then the proof is entirely analogous to the previous case, using now axiom **Du_□** and the second part of the Existence Lemma.
- If $\varphi = [t?] \psi$, then, using the same reasoning of the previous two cases and axiom **Du_t**, it is sufficient to show the result for formulas of the form $\phi = \langle t? \rangle \phi_1$. By axiom **E₁**, $\langle t? \rangle \phi_1 \in \Gamma$ iff $\langle t? \rangle \top \in \Gamma$ and $\phi_1 \in \Gamma$. Now, by the definitions of **E^C** and **E^{Δ⁺}**, $\langle t? \rangle \top \in \Gamma$ iff $\Gamma \in \mathcal{E}^{\Delta^+}(t)$ and, by the induction hypothesis, $\phi_1 \in \Gamma$ iff $\mathcal{M}^{\Delta^+}, \Gamma \Vdash \phi_1$. Finally, $\Gamma \in \mathcal{E}^{\Delta^+}(t)$ and $\mathcal{M}^{\Delta^+}, \Gamma \Vdash \phi_1$ iff $\mathcal{M}^{\Delta^+}, \Gamma \Vdash \langle t? \rangle \phi_1$.
- If $\varphi = t \gg_i \psi$, $\varphi = \mathcal{P}it$ or $\varphi = t :_i \psi_1$, the proof is straightforward from the previous cases and axioms **Adm**, **Pla** and **Jus**, respectively. \square

Theorem 5 (Completeness). *Every consistent set of formulas is satisfiable in an LPJ model (definition 9).*

Proof. Let Δ be a consistent set of formulas and $\Delta^+ \supseteq \Delta$ be a MCS obtained from Δ . Consider the canonical model \mathcal{M}^{Δ^+} . Then, by the Truth Lemma, we conclude that $\mathcal{M}^{\Delta^+}, \Delta^+ \Vdash \varphi$, for all $\varphi \in \Delta$, so Δ is satisfiable. \square

Classic-Like Cut-Based Tableau Systems for Finite-Valued Logics

Marco Volpe¹, João Marcos², and Carlos Caleiro³

¹ Dipartimento di Informatica, Università di Verona, Italy

² LoLITA and DIMAp, UFRN, Brazil

³ SQIG, Instituto de Telecomunicações and Depto. de Matemática, IST, Portugal

Abstract. A general procedure is presented for producing classic-like cut-based tableau systems for finite-valued logics. In such systems, cut is the only branching rule, and formulas are accompanied by signs acting as syntactic proxies for the two classical truth-values. The systems produced are guaranteed to be sound, complete and analytic, and they are also seen to polynomially simulate the truth-table method, thus extending the results in [7]. Lukasiewicz's 3-valued logic is used throughout as a simple illustrative example.

1 Introduction

In [4,5], in accordance with the so-called Suszko's Thesis, the authors have shown how to take advantage of the intrinsic bivalence that stems from the distinction between designated and undesignated truth-values in any sufficiently expressive finite-valued logic in order to provide the latter with a (non-truth-functional) bivalent semantics, and ultimately with a classic-like tableau proof system, using 2-signed formulas, associated to a simple decision procedure. However, due to the necessary encoding of the original semantics of the logic in terms of the two classical values, one ends up having to work with tableau rules having a significant number of branches that unavoidably lead to very large derivations.

It is widely known that proofs not involving cuts (or equivalently modus ponens) can be very inefficient. For classical propositional logic, for instance, cut-based proofs can be exponentially smaller than the shortest corresponding cut-free proofs (see [2]). Still, the unrestricted use of the cut rule poses a serious challenge for proof-search. First proposed by Mondadori, KE tableaux for classical logic, thoroughly studied in [6,9,7], are a cut-based tableau system that employs only analytic cuts and which is known to polynomially simulate the truth-table decision method, in the general case, bringing thus an exponential gain over conventional cut-free tableau systems in the worst cases.

Recent interest in KE tableaux (e.g. [10]) stimulated us to consider exploring a similar strategy, but now for producing classic-like cut-based tableau systems for finite-valued logics in general, capitalizing on [4,5], to which an analytic restriction of cut may be imposed, and which might in principle share the benefits of KE tableaux in terms of proof complexity. This paper reports on such an exploration.

2 Background

Consider an alphabet with a denumerable set \mathcal{A} of *atoms* and a finite set Σ of *primitive connectives*. The arity of a given connective $\odot \in \Sigma$ is to be denoted by $\text{arg}\odot$. The set \mathbb{S} of *formulas* is the carrier of the free Σ -algebra generated by \mathcal{A} . In dealing with finite-valued logics, $\mathcal{V}_n = \{v_0, v_1, \dots, v_{n-1}\}$ will be used to denote sets of *truth-values*, given $n \in \mathbb{N}$, and these are supposed to be partitioned into a set $\mathcal{D} = \{v_i \mid 0 \leq i \leq m\}$ of *designated* values and a set $\mathcal{U} = \{v_i \mid m+1 \leq i \leq n-1\}$ of *undesignated* values. As a matter of stipulation, we will denote v_0 by F and v_{n-1} by T . In general, an (*n-valued*) *assignment* of truth-values to the atoms is any mapping $\rho : \mathcal{A} \rightarrow \mathcal{V}_n$, and an (*n-valued*) *valuation* is any extension $w : \mathbb{S} \rightarrow \mathcal{V}_n$ of such an assignment to the set of all formulas. An *n-valent semantics* for \mathbb{S} based on \mathcal{V}_n , then, is simply a collection of *n-valued* valuations. In particular, we will call *bivalent* any (classic-like) semantics where $\mathcal{V}_2 = \{F, T\}$ and $\mathcal{D}_2 = \{T\}$; the corresponding valuations are called *bivaluations*. As usual, we call a valuation w a *model* of $\Delta \subseteq \mathbb{S}$ if $w(\Delta) \subseteq \mathcal{D}$. A canonical notion of *entailment* characterizing a *logic* \mathcal{L} over \mathbb{S} is associated to an *n-valent* semantics Sem by setting $\Gamma \models \alpha$ iff every model of Γ in Sem is a model of $\{\alpha\}$. A remarkable case of *n-valent* semantics corresponds to those we call *truth-functional*: such a semantics is given to the set of formulas \mathbb{S} by defining an appropriate Σ -algebra \mathbb{V} with carrier \mathcal{V}_n , by associating a *k-ary* interpretation operator $\widehat{\odot} : \mathcal{V}_n^k \rightarrow \mathcal{V}_n$ to each $\odot \in \Sigma$ with $\text{arg}\odot = k$, and by collecting in Sem the set of all homomorphisms $\xi : \mathbb{S} \rightarrow \mathbb{V}$. Any such homomorphism, as usual, may be understood as the unique extension of an assignment $\rho : \mathcal{A} \rightarrow \mathcal{V}_n$ into a valuation $\xi_\rho : \mathbb{S} \rightarrow \mathbb{V}$ where $\xi(\odot(\varphi_1, \dots, \varphi_k)) = \widehat{\odot}(\xi(\varphi_1), \dots, \xi(\varphi_k))$. Any logic characterized by truth-functional means, for a given \mathcal{V}_n , is called *n-valued*.

Let us now consider the total mapping $t : \mathcal{V}_n \rightarrow \mathcal{V}_2$ such that $t(v) = T$ iff $v \in \mathcal{D}$ and define, for any valuation $\xi : \mathbb{S} \rightarrow \mathbb{V}$ of an *n-valued* semantics Sem , the bivaluation $b_\xi = t \circ \xi$. Though this bivalent semantics gives up the fundamental feature of truth-functionality, we have shown in previous papers (check [3] and the survey [5]) that it can still be very useful. As explained below, to accomplish the bivalent reduction constructively, in order to be able to distinguish any given value from any other value we just need to associate a unique ‘binary print’ to each truth-value of a given *n-valued* logic \mathcal{L} . Given $v_i, v_j \in \mathcal{V}$, we write $v_i \# v_j$ and say that v_i and v_j are *separated* in case $t(v_i) \neq t(v_j)$. Given two formulas φ_i and φ_j and a valuation ξ such that $v_i = \xi(\varphi_i) \neq \xi(\varphi_j) = v_j$ yet $b_\xi(\varphi_i) = b_\xi(\varphi_j)$, we say that a one-variable formula $\theta^{ij}(p)$ of \mathcal{L} *separates* v_i and v_j if $\xi(\theta^{ij}(\varphi_i)) \# \xi(\theta^{ij}(\varphi_j))$ (or, equivalently, $b_\xi(\theta^{ij}(\varphi_i)) \neq b_\xi(\theta^{ij}(\varphi_j))$). In that case we will also say that the values v_i and v_j of \mathcal{L} are *effectively distinguishable*, as they may be separated using the original linguistic resources of \mathcal{L} . Finally, we will say that the logic \mathcal{L} is *effectively separable* in case its truth-values are pairwise effectively distinguishable, that is, for any pair of distinct values $\langle v_i, v_j \rangle \in \mathcal{D}^2 \cup \mathcal{U}^2$ a one-variable formula $\theta^{ij}(p)$ can be found in \mathcal{L} that separates v_i and v_j . From this point on, for simplicity of exposition, we assume that all the necessary separators belong to the set Σ of primitive connectives of the logic — note that this is not really a restriction, as one can always conservatively extend an

n -valued logic \mathcal{L} with a conveniently interpreted n -ary connective. Let Θ denote a finite sequence $[\theta_r(p)]_{r=0}^s = \langle \theta_0(p), \theta_1(p), \dots, \theta_s(p) \rangle$ of one-variable formulas used as separators, where we assume $\theta_0(p) = p$. Obviously, $\theta_0(p)$ by itself suffices to separate any pair of values $\langle v_i, v_j \rangle \in (\mathcal{D} \times \mathcal{U}) \cup (\mathcal{U} \times \mathcal{D})$. We will call *binary print* of a value $v \in \mathcal{V}$ the sequence $\bar{v} = [b_{\S}(\theta_r(p))]_{r=0}^s$, where $\S(p) = v$. Notice that for every pair of distinct values $\langle v_i, v_j \rangle \in \mathcal{V}^2$ it is now obviously the case that $\bar{v}_i \neq \bar{v}_j$.

Example 1. Our running example will be Lukasiewicz’s 3-valued logic, L_3 . The logic may be described by choosing as primitive connectives $\Sigma = \{\neg, \diamond, \supset\}$, with $\arg\neg = \arg\diamond = 1$ and $\arg\supset = 2$, and by considering the set of truth-values $\mathcal{V}_3 = \{v_0, v_1, v_2\}$, with v_2 as the sole designated value. The operators interpreting the connectives are described in Table 1.

Table 1. Interpretation operators in L_3

x	$\neg x$	$\widehat{\diamond}x$
v_0	v_2	v_0
v_1	v_1	v_2
v_2	v_0	v_2

$x \supset y$	v_0	v_1	v_2
v_0	v_2	v_2	v_2
v_1	v_1	v_2	v_2
v_2	v_0	v_1	v_2

We need to look for a way of separating the two undesigned values v_0 and v_1 , and accordingly we will have to set $\Theta = \langle p, \theta_1(p) \rangle$, for some convenient separator θ_1 . There are two obvious separators already in the alphabet of L_3 . We will here choose Lukasiewicz’s ‘possibility’ operator \diamond as θ_1 . The same choice has in fact been made in [4], but there we have introduced \diamond by abbreviation, noticing that $\widehat{\diamond}x \stackrel{def}{=} (\neg x)\widehat{\supset}x$. Clearly, such choice originates the binary prints $\langle F, F \rangle$, $\langle F, T \rangle$ and $\langle T, T \rangle$, respectively for v_0 , v_1 and v_2 . Note that the sequence $\langle T, F \rangle$ is unrealizable, as it does not correspond to any of the values in \mathcal{V}_3 . Below, when \diamond appears in the role of the separator θ_1 we will write it as θ , to help calling attention to the two different roles played by this connective. In [12] we have studied the effect of choosing Lukasiewicz’s ‘negation’ operator \neg as θ_1 .

In earlier work, we have used this bivalent setting to produce classic-like tableau systems $\mathcal{T}(\mathcal{L}, \Theta)$ for any given n -valued logic \mathcal{L} effectively separable by $\Theta = [\theta_r(p)]_{r=0}^s$. We refer the reader to [4,5] for the full details. However, it is worth mentioning here a few key ingredients of the procedure. Mirroring the classical truth-values $\{F, T\}$, we work with 2-signed formulas $X:\varphi$ such that $X \in \{F, T\}$ and $\varphi \in \mathbb{S}$. As a matter of convention, we shall say that an n -valued valuation \S satisfies a labeled formula $X:\varphi$ if $b_{\S}(\varphi) = X$. The notion of satisfaction extends naturally to sets of labeled formulas. Given a binary print $\bar{v} = [X_r]_{r=0}^s$, we use $\bar{v}^{\mathbb{S}}(\varphi)$ to denote the sequence of signed formulas $[X_r:\varphi]_{r=0}^s$.

The cornerstone of $\mathcal{T}(\mathcal{L}, \Theta)$ is the recipe for obtaining elimination rules for the connectives. Using $\&$ to represent conjunction in the classical metalanguage, \parallel to represent disjunction, \implies to represent implication, and \ast to represent an absurd, we produce a tableau rule for each schematic signed formula $X:\theta(\odot(\varphi_1, \dots, \varphi_k))$ where $X \in \{F, T\}$, $\theta \in \Theta$, and $\odot \in \Sigma$ with $\arg\odot = k$.

We further demand that if $\theta = \theta_0$, then $\odot \notin \Theta$, and we write more simply $X: \odot(\varphi_1, \dots, \varphi_k)$ instead of $X:\theta_0(\odot(\varphi_1, \dots, \varphi_k))$. The elimination rules are produced by collecting the tuples of binary prints that a homomorphic n -valuation \S can assign to the formulas $\varphi_1, \dots, \varphi_k$ in order to satisfy the signed formula. Letting $B_X^{\theta \odot}([\varphi_i]_{i=1}^k) = \{\&[\overline{v}_i^{\S}(\varphi_i)]_{i=1}^k \mid t(\widehat{\theta}(\widehat{\odot}([\varphi_i]_{i=1}^k))) = X\}$, the corresponding tableau rule is then given by

$$X:\theta(\odot([\varphi_i]_{i=1}^k)) \Longrightarrow \parallel B_X^{\theta \odot}([\varphi_i]_{i=1}^k).$$

In our metalanguage the above expression represents a tableau rule: the antecedent of each rule is the head, and the succedent describes the children nodes that may be created once the head matches a node of a previously given branch.

Example 2. In the case of L_3 with the single separator $\theta = \diamond$, the above described recipe would produce, for instance, a rule of the form

$$T:\theta(\neg\varphi_1) \Longrightarrow (F:\varphi_1 \ \& \ F:\theta(\varphi_1)) \parallel (F:\varphi_1 \ \& \ T:\theta(\varphi_1))$$

simply because $\S(\diamond(\neg\varphi_1)) = v_2$ if and only if $\widehat{\odot}(\widehat{\neg}(\S(\varphi_1))) = v_2$ if and only if $\S(\varphi_1) = v_0$ or $\S(\varphi_1) = v_1$. Note that $\langle F, F \rangle$ and $\langle F, T \rangle$ are precisely the binary prints associated respectively to v_0 and v_1 .

Another example, now using the identity θ_0 , would yield

$$\begin{aligned} F:\varphi_1 \supset \varphi_2 \Longrightarrow & (F:\varphi_1 \ \& \ T:\theta(\varphi_1) \ \& \ F:\varphi_2 \ \& \ F:\theta(\varphi_2)) \\ & \parallel (T:\varphi_1 \ \& \ T:\theta(\varphi_1) \ \& \ F:\varphi_2 \ \& \ T:\theta(\varphi_2)) \\ & \parallel (T:\varphi_1 \ \& \ T:\theta(\varphi_1) \ \& \ F:\varphi_2 \ \& \ F:\theta(\varphi_2)) \end{aligned}$$

because $\S(\varphi_1 \supset \varphi_2) \neq v_2$ if and only if $\S(\varphi_1) \widehat{\supset} \S(\varphi_2) \neq v_2$ if and only if $\S(\varphi_1) = v_1$ and $\S(\varphi_2) = v_0$, or $\S(\varphi_1) = v_2$ and $\S(\varphi_2) = v_0$, or $\S(\varphi_1) = v_2$ and $\S(\varphi_2) = v_1$.

Such rules may be streamlined using classical equivalences in the metalanguage, and completeness of the tableau system is attained by the addition of suitable closure rules (see [A]).

As it might be expected, the tableau systems produced using the above recipe originate in general very redundant and highly branching derivations. The next sections will show how to use a similar approach to obtain more efficient systems, in which the only branching rule is an analytic version of the cut rule.

Before proceeding, we introduce some extra useful terminology and notation. As usual, each φ_i , for $1 \leq i \leq k$, is called an *immediate subformula* of $\odot(\varphi_1, \dots, \varphi_k)$. The set of *proper subformulas* of a given $\odot(\varphi_1, \dots, \varphi_k)$ contains the immediate subformulas of this formula and the immediate subformulas of any formula therein contained. We here dub Θ -*immediate subformula* of $\odot(\varphi_1, \dots, \varphi_k)$ any formula of the form $\theta(\varphi_i)$, for $1 \leq i \leq k$ and $\theta \in \Theta$. The set of *proper Θ -subformulas* of a given formula has the obvious definition. A Θ -formula is called *atomic* if it has no Θ -immediate subformulas. We also define the *size* of a formula (signed or not) to be the cardinality of its set of subformulas (forgetting the sign, in the case of a signed formula). For convenience, we will assume $F^C = T$ and $T^C = F$ as the *conjugates* of the two classical truth-values, and extend the notation accordingly to the syntactic labels T and F.

In the next section we will illustrate the ideas behind our novel rule-extraction algorithm by discussing what happens in the running example of L_3 . After that we will present and study our general method in full detail.

3 A Cut-Based Tableau System for L_3

The idea here is to find a suitable way of defining a tableau system for L_3 whose only branching rule is a cut rule, in a way that generalizes the KE tableaux of [6,9], proposed for classical logic. Recall that we consider L_3 separators $\Theta = \langle p, \theta(p) \rangle$, where $\theta = \diamond$. Our tableau system will consist of three classes of rules: the cut rule, elimination rules, and closure rules.

The cut rule is the only branching rule, i.e., the only rule with more than one branch in the succedent, and has the following typical form:

$$(L_3.\text{Cut}) \quad \implies F:\varphi \parallel T:\varphi$$

In Section 4 we will show that it is possible to restrict its use only to analytic applications.

We will now take full advantage of the classic-like semantics of L_3 introduced by its corresponding bivalent semantics, obtained following the procedure detailed in [3], and extract from it suitable elimination and closure rules for our novel cut-based system.

As explained and illustrated in Section 2, we will need suitable elimination rules for signed formulas of the forms $X:\neg\varphi_1$, $X:\varphi_1 \supset \varphi_2$, $X:\theta(\neg\varphi_1)$, $X:\theta(\varphi_1 \supset \varphi_2)$ and $X:\theta(\diamond(\varphi_1))$, where $\theta = \diamond$ and $X \in \{F, T\}$. Recall that, given a formula φ , we can express its 3-valued truth-table as a bivalent one, where the value of φ depends only on the values of its Θ -subformulas. Given that the procedure is systematic, let us focus at a fragment of it, and consider the bivalent version of

Table 2. The bivalent version of \supset

combination	φ_1	$\theta(\varphi_1)$	φ_2	$\theta(\varphi_2)$	$\varphi_1 \supset \varphi_2$
0	F	F	F	F	T
1	F	F	F	T	T
2	F	F	T	F	-
3	F	F	T	T	T
4	F	T	F	F	F
5	F	T	F	T	T
6	F	T	T	F	-
7	F	T	T	T	T
8	T	F	F	F	-
9	T	F	F	T	-
10	T	F	T	F	-
11	T	F	T	T	-
12	T	T	F	F	F
13	T	T	F	T	F
14	T	T	T	F	-
15	T	T	T	T	T

the truth-table corresponding to the formula $\varphi_1 \supset \varphi_2$. In Table 2 we include all the combinations for the signs of φ_1 , $\theta(\varphi_1)$, φ_2 , $\theta(\varphi_2)$. A dash (-) in the last column indicates that the corresponding combination contains a sequence $\langle T, F \rangle$ for some $\langle \varphi, \theta(\varphi) \rangle$ that corresponds to no binary print \bar{v} , for $v \in \mathcal{V}_3$.

From Table 2 we can mechanically extract a set of elimination rules for L_3 's 'implication' connective \supset . Indeed, consider the partial bivaluation b^j described at combination j of the table, in such a way that we shall say that $X_j:\psi$ is satisfied if ψ is at the head of some column and the j -th combination below it contains value X_j . In our cut-based tableau system there will be a rule corresponding to each collection R of signed formulas satisfied by some partial bivaluation b_j with the requirement that such collection must contain $X_j:\varphi_1 \supset \varphi_2$. For instance, some possible such collections are $\{F:\varphi_1 \supset \varphi_2\}$, $\{F:\varphi_1 \supset \varphi_2, T:\varphi_1\}$ and $\{T:\varphi_1 \supset \varphi_2, F:\theta(\varphi_1), T:\theta(\varphi_2)\}$. Each such collection R , read as a conjunction, will form the antecedent of a tableau rule. Let $\text{Mod}(R)$ be the set of all partial bivaluations corresponding to combinations that satisfy R . The succedent of the corresponding rule will contain the (possibly empty) collection, read as a conjunction, of all signed formulas that are simultaneously satisfied by all the bivaluations in $\text{Mod}(R)$. As an example, let $\{F:\varphi_1 \supset \varphi_2\}$ be the antecedent of a given rule. Then we can restrict our attention to the combinations 4, 12 and 13 from Table 2. We may easily notice that $\{T:\theta(\varphi_1), F:\varphi_2\}$ is an invariant in these combinations. The corresponding tableau rule will in this case read:

$$(L_3.\supset 1^*) \quad F:\varphi_1 \supset \varphi_2 \implies T:\theta(\varphi_1) \ \& \ F:\varphi_2.$$

Note that we omit the (empty) rules originating from partial bivaluations for which in the derived restricted table we have no invariants (other than the signed formulas fixed for the antecedent). For example, we do not have any rule with $\{T:\varphi_1 \supset \varphi_2\}$ as antecedent, since $T:\varphi_1 \supset \varphi_2$ itself is the only invariant in the corresponding restricted table (it suffices to contrast combinations 0 and 15). A general and formal account of this rule-extraction procedure will be given in Section 4. Table 3 contains the full set of rules obtained, in particular, for the connective \supset .

It is clear that the procedure described above for the mechanical extraction of elimination rules may generate a lot of redundancies. As a trivial example, one may notice that the rule $(L_3.\supset 2^*)$ of Table 3 is redundant in the presence of $(L_3.\supset 1^*)$ since they have the same succedent and the collection of antecedents of one of them is included in the other. One may notice that the rule $(L_3.\supset 4^*)$ is also redundant in the presence of $(L_3.\supset 1^*)$, given that the latter has a more informative succedent than the former, even if it contains less hypotheses in the antecedent. After the elimination of all such redundant rules, and repeating the procedure for all connectives, with and without the separator θ , we obtain the elimination rules in Table 4.

Finally, with respect to the closure rules, we follow [4] to the letter. Besides the traditional closure rule for 2-signed tableaux, which says that a branch is closed once it contains two signed formulas of the form $F:\varphi$ and $T:\varphi$, additional closure rules will be needed in order to exclude unrealizable binary prints — in

this case of L_3 and Θ , we are talking about $\langle T, F \rangle$. Hence, an additional closure rule will say that branches containing both a signed formula of the form $T:\varphi$ and a signed formula of the form $F:\theta(\varphi)$ may be closed. One might represent the above mentioned such closure rules by writing:

$$\begin{aligned} (L_3.C0) \quad & F:\varphi \ \& \ T:\varphi \quad \Longrightarrow \ * \\ (L_3.C1) \quad & T:\varphi \ \& \ F:\theta(\varphi) \quad \Longrightarrow \ * \end{aligned}$$

Table 4. Streamlined elimination rules of the tableau system for L_3

$(L_3.\neg 1)$	$F:\neg\varphi_1$	$\Longrightarrow T:\theta(\varphi_1)$
$(L_3.\neg 2)$	$T:\neg\varphi_1$	$\Longrightarrow F:\varphi_1 \ \& \ F:\theta(\varphi_1)$
$(L_3.\supset 1)$	$F:\varphi_1 \supset \varphi_2$	$\Longrightarrow T:\theta(\varphi_1) \ \& \ F:\varphi_2$
$(L_3.\supset 2)$	$F:\varphi_1 \supset \varphi_2 \ \& \ T:\theta(\varphi_2)$	$\Longrightarrow T:\varphi_1 \ \& \ T:\theta(\varphi_1) \ \& \ F:\varphi_2$
$(L_3.\supset 3)$	$F:\varphi_1 \supset \varphi_2 \ \& \ F:\varphi_1$	$\Longrightarrow T:\theta(\varphi_1) \ \& \ F:\varphi_2 \ \& \ F:\theta(\varphi_2)$
$(L_3.\supset 4)$	$T:\varphi_1 \supset \varphi_2 \ \& \ F:\theta(\varphi_2)$	$\Longrightarrow F:\varphi_1 \ \& \ F:\theta(\varphi_1) \ \& \ F:\varphi_2$
$(L_3.\supset 5)$	$T:\varphi_1 \supset \varphi_2 \ \& \ F:\varphi_2$	$\Longrightarrow F:\varphi_1$
$(L_3.\supset 6)$	$T:\varphi_1 \supset \varphi_2 \ \& \ T:\varphi_2$	$\Longrightarrow T:\theta(\varphi_2)$
$(L_3.\supset 7)$	$T:\varphi_1 \supset \varphi_2 \ \& \ F:\theta(\varphi_1)$	$\Longrightarrow F:\varphi_1$
$(L_3.\supset 8)$	$T:\varphi_1 \supset \varphi_2 \ \& \ F:\theta(\varphi_1) \ \& \ T:\varphi_2$	$\Longrightarrow F:\varphi_1 \ \& \ T:\theta(\varphi_2)$
$(L_3.\supset 9)$	$T:\varphi_1 \supset \varphi_2 \ \& \ T:\theta(\varphi_1)$	$\Longrightarrow T:\theta(\varphi_2)$
$(L_3.\supset 10)$	$T:\varphi_1 \supset \varphi_2 \ \& \ T:\theta(\varphi_1) \ \& \ F:\varphi_2$	$\Longrightarrow F:\varphi_1 \ \& \ T:\theta(\varphi_2)$
$(L_3.\supset 11)$	$T:\varphi_1 \supset \varphi_2 \ \& \ T:\varphi_1$	$\Longrightarrow T:\theta(\varphi_1) \ \& \ T:\varphi_2 \ \& \ T:\theta(\varphi_2)$
$(L_3.\theta \neg 1)$	$F:\theta(\neg\varphi_1)$	$\Longrightarrow T:\varphi_1 \ \& \ T:\theta(\varphi_1)$
$(L_3.\theta \neg 2)$	$T:\theta(\neg\varphi_1)$	$\Longrightarrow F:\varphi_1$
$(L_3.\theta \supset 1)$	$F:\theta(\varphi_1 \supset \varphi_2)$	$\Longrightarrow T:\varphi_1 \ \& \ T:\theta(\varphi_1) \ \& \ F:\varphi_2 \ \& \ F:\theta(\varphi_2)$
$(L_3.\theta \supset 2)$	$T:\theta(\varphi_1 \supset \varphi_2) \ \& \ F:\theta(\varphi_2)$	$\Longrightarrow F:\varphi_1 \ \& \ F:\varphi_2$
$(L_3.\theta \supset 3)$	$T:\theta(\varphi_1 \supset \varphi_2) \ \& \ T:\varphi_2$	$\Longrightarrow T:\theta(\varphi_2)$
$(L_3.\theta \supset 4)$	$T:\theta(\varphi_1 \supset \varphi_2) \ \& \ F:\theta(\varphi_1)$	$\Longrightarrow F:\varphi_1$
$(L_3.\theta \supset 5)$	$T:\theta(\varphi_1 \supset \varphi_2) \ \& \ F:\theta(\varphi_1) \ \& \ T:\varphi_2$	$\Longrightarrow F:\varphi_1 \ \& \ T:\theta(\varphi_2)$
$(L_3.\theta \supset 6)$	$T:\theta(\varphi_1 \supset \varphi_2) \ \& \ T:\varphi_1$	$\Longrightarrow T:\theta(\varphi_1) \ \& \ T:\theta(\varphi_2)$
$(L_3.\theta (\diamond 1))$	$F:\theta(\diamond(\varphi_1))$	$\Longrightarrow F:\varphi_1 \ \& \ F:\theta(\varphi_1)$
$(L_3.\theta (\diamond 2))$	$T:\theta(\diamond(\varphi_1))$	$\Longrightarrow T:\theta(\varphi_1)$

Figure 1 shows an example of a tableau for L_3 using the set of rules obtained as described above. In this example we get (2.1) and (2.2) by applying rule $(L_3.\supset 1)$ to the formula (1). The same rule applies to (2.2) to originate (3.1) and (3.2). An application of $(L_3.\supset 3)$ to (1) and (3.2) gives (4.1). Then we apply $(L_3.\theta \supset 1)$ to (4.1) and get (5.1) and (5.2). We close the tableau by applying $(L_3.C0)$ to (2.1) and (5.2). Note that the derivation tree is linear as no use of $(L_3.Cut)$ was necessary.

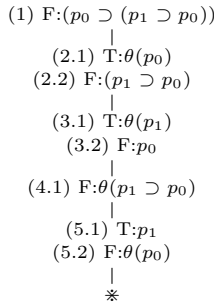


Fig. 1. A refutation of $p_0 \supset (p_1 \supset p_0)$ in the cut-based tableau system for L_3

4 The Tableau System

4.1 Rules

Let \mathcal{L} be an effectively separable n -valued logic with a set of formulas \mathbb{S} generated over the set of connectives Σ by the set of atoms \mathcal{A} , and having $\mathcal{D} \subseteq \mathcal{V}_n$ as its set of designated values. We assume also that its binary prints are produced by a convenient sequence of separators $\Theta = [\theta_r(p)]_{r=0}^s$, where $\theta_0(p) = p$. In the following, we will exhibit the rules of our novel cut-based tableau system for \mathcal{L} .

As explained before, the only branching rule of our system is:

$$(\mathcal{L}.Cut) \qquad \qquad \qquad \Longrightarrow F:\varphi \parallel T:\varphi$$

Below in this section, we will show that it is possible to restrict the use of such cut rule only to analytic applications, that is, applications to tableau branches of which φ is a Θ -subformula.

Let now $\mathbf{BP} = \{F, T\}^{s+1}$ be the set of all $(s + 1)$ -long binary prints and let a *partial binary print* be any sequence $\bar{c}_R = [c_r]_{r \in R}$ such that $R \subseteq \{0, 1, \dots, s\}$ and each $c_r \in \{F, T\}$ (this definition includes, of course, all binary prints in \mathbf{BP} , as strict partiality occurs precisely when R is a proper subset of $\{0, 1, \dots, s\}$). We say that a partial binary print \bar{d}_U *extends* \bar{c}_R if $R \subseteq U$ and $d_r = c_r$ for every $r \in R$.

We say that a sequence $[\bar{v}_i]_{i=1}^k$ of binary prints *satisfies* a signed formula $X:\theta(\odot([\varphi_i]_{i=1}^k))$ if $t(\hat{\theta}(\hat{\odot}([v_i]_{i=1}^k))) = X$. Further, we say that a signed formula is *satisfiable by a sequence* $[\bar{c}_{iR_i}]_{i=1}^k$ *of partial binary prints* if it is satisfied by some sequence of binary prints that extends $[\bar{c}_{iR_i}]_{i=1}^k$ componentwise.

Let $R_i, U_i \subseteq \{0, 1, \dots, s\}$ be such that $R_i \cap U_i = \emptyset$, for each $1 \leq i \leq k$, let $[\bar{c}_{iR_i}]_{i=1}^k$ and $[\bar{d}_{iU_i}]_{i=1}^k$ be two disjoint sequences of partial binary prints, and let δ be the signed formula $X:\theta(\odot([\varphi_i]_{i=1}^k))$. We say that $[\bar{c}_{iR_i}]_{i=1}^k$ *entails* $[\bar{d}_{iU_i}]_{i=1}^k$ *with respect to* δ when, for every sequence $[\bar{v}_i]_{i=1}^k$ of binary prints satisfying δ , if $[\bar{v}_i]_{i=1}^k$ extends $[\bar{c}_{iR_i}]_{i=1}^k$ then $[\bar{v}_i]_{i=1}^k$ extends $[\bar{d}_{iU_i}]_{i=1}^k$.

We now produce elimination rules for each signed formula $\delta = X:\theta(\odot([\varphi_i]_{i=1}^k))$ such that if $\theta = \theta_0$, then $\odot \notin \Theta$. We consider, for each sequence of partial binary prints $[\bar{c}_{iR_i}]_{i=1}^k$ that satisfies δ , the following rule:

$$(\mathcal{L}.R_X^{\theta \odot} [\bar{c}_{iR_i}]_{i=1}^k) \qquad X:\theta(\odot([\varphi_i]_{i=1}^k)) \ \& \ (\&[\bar{c}_{iR_i}^S(\varphi_i)]_{i=1}^k) \Longrightarrow \&[\bar{d}_{iU_i}^S(\varphi_i)]_{i=1}^k$$

where $[\bar{d}_{iU_i}]_{i=1}^k$ is the largest sequence of partial binary prints entailed by $[\bar{c}_{iR_i}]_{i=1}^k$ with respect to δ . That is to say that \bar{d}_{iU_i} extends any other sequence of partial binary prints entailed by $[\bar{c}_{iR_i}]_{i=1}^k$ with respect to δ . Note that such a largest partial binary print is well-defined. Indeed, given the fact that δ is satisfiable, any two entailed sequences of partial binary prints $[\bar{e}_{iV_i}]_{i=1}^k$ and $[\bar{f}_{iW_i}]_{i=1}^k$ are compatible, i.e., for each i , if $j \in V_i \cap W_i$ then $e_{ij} = f_{ij}$, and can thus be joined into $[\bar{g}_{iV_i \cup W_i}]_{i=1}^k$ such that, for each i , $g_{ij} = e_{ij}$ if $j \in V_i$ and $g_{ij} = f_{ij}$ if $j \in W_i$. Clearly, $[\bar{g}_{iV_i \cup W_i}]_{i=1}^k$ extends both sequences and is also entailed by $[\bar{c}_{iR_i}]_{i=1}^k$ with respect to δ .

The set of elimination rules listed above might contain a lot of redundancies. We can see an elimination rule as a pair of sets $\langle \Pi_1, \Pi_2 \rangle$ where Π_1 contains the signed formulas in the antecedent and Π_2 the signed formulas in the succedent of the rule. In this case, we say that a rule (Δ_1, Δ_2) is *redundant* in a system \mathcal{T} if there is a different rule (Γ_1, Γ_2) in \mathcal{T} such that: (i) $\Gamma_1 \subseteq \Delta_1$; and (ii) $\Delta_2 \subseteq \Gamma_2$.

Finally, closure rules look precisely as in the system of [5]. We briefly explain the procedure below, for the sake of self-containment.

We consider first the usual classic-like closure rule:

$$(\mathcal{L}.C0) \quad F:\varphi \ \& \ T:\varphi \implies *$$

In addition, we have to consider the unrealizable binary prints. Let $\mathbf{CS} = \mathbf{BP} \setminus \{\bar{v} \mid v \in \mathcal{V}_n\}$ be the set of all the bivalent sequences that are *not* produced as binary prints of truth-values of \mathcal{L} . Intuitively, any *closing sequence* $\bar{c} \in \mathbf{CS}$ brings about information that is unobtainable, allowing one thus to close a tableau branch that contains it. Information, even if partial, leading unambiguously to a sequence in \mathbf{CS} should always give rise to a closed tableau. Indeed, closing information is carried by any partial binary print \bar{c}_R such that all of its $2^{\#(\Theta) - \#(R)}$ possible total extensions are in \mathbf{CS} . Hence, it would be reasonable to add a different closure rule for each such partial closing information. However, it suffices to take into account just the minimal closing situations, that is, closing partial sequences \bar{c}_R that cannot be obtained as extensions of any other closing partial sequence. In general, where $\bar{c}_R = [c_r]_{r \in R}$ is some partial binary print, we write $\bar{c}_R^{\mathbb{S}}(\varphi) = [s(c_r):\theta_r(\varphi)]_{r \in R}$ for the linguistic 2-signed version of such sequence, where $s(c_r) = \mathbf{T}$ if $c_r = T$ and $s(c_r) = \mathbf{F}$ if $c_r = F$. Accordingly, for each minimal closing partial binary print \bar{c}_R , we consider an additional closure rule:

$$(\mathcal{L}.Ck) \quad \&(\bar{c}_R^{\mathbb{S}}(\varphi)) \implies *$$

Finally, we get further closure rules as particular cases in the production of elimination rules. Namely, we need to consider when the formula $X:\theta(\odot([\varphi_i]_{i=1}^k))$ is not satisfiable. For any such a case, we consider the additional closure rule:

$$(\mathcal{L}.C_X^{\theta \odot}) \quad X:\theta(\odot([\varphi_i]_{i=1}^k)) \implies *$$

We can now define our full cut-based tableau system.

Definition 1. *The tableau system $\mathcal{T}^{\text{cut}}(\mathcal{L}, \Theta)$ for the logic \mathcal{L} with respect to Θ is composed of rule $(\mathcal{L}.Cut)$, non-redundant elimination rules $(\mathcal{L}.R_X^{\theta \odot}[\bar{c}_{i R_i}]_{i=1}^j)$, and closure rules $(\mathcal{L}.C0)$, $(\mathcal{L}.Ck)$, $(\mathcal{L}.C_X^{\theta \odot})$ defined as above.*

Tableaux are built as usual, by applying the above rules, given an initial sequence of 2-signed formulas, and a branch is said to be *closed* if its closure is obtained by the application of any of the (Ck) rules, including $(C0)$, or of any $C_X^{\theta \odot}$ rule. Branches that are not closed are said to be *open*. A tableau is said to be *closed* in case all of its branches are closed.

4.2 Properties

We will now check the soundness and completeness of our cut-based tableau systems $\mathcal{T}^{\text{cut}}(\mathcal{L}, \Theta)$.

As usual, we say that the system is *sound* if the root of any closed tableau is unsatisfiable. Conversely, we say that the system is *complete* if every unsatisfiable finite set of signed formulas is the root of some closed tableau.

Theorem 1. *The tableau system $\mathcal{T}^{\text{cut}}(\mathcal{L}, \Theta)$ is sound and complete.*

Proof. For soundness, it is sufficient to show that if a homomorphic n -valuation $\xi : \mathbb{S} \rightarrow \mathcal{V}_n$ satisfies the head of a rule then it must satisfy one of the branches of its succedent. This is clearly the case for the cut rule. The property also holds for the closure rules, as shown in [45]. We are thus left with proving the claim for the linear elimination rules ($\mathcal{L}.R_X^{\theta \odot}[\overline{c}_{i R_i}]_{i=1}^k$), which holds basically by construction. Indeed, if ξ satisfies $X:\theta(\odot([\varphi_i]_{i=1}^j))$ and $[\overline{c}_{i R_i}^{\mathbb{S}}(\varphi_i)]_{i=1}^k$ then ξ must also satisfy $[\overline{d}_{i U_i}^{\mathbb{S}}(\varphi_i)]_{i=1}^k$ because $[\overline{c}_{i R_i}^{\mathbb{S}}(\varphi_i)]_{i=1}^k$ entails $[\overline{d}_{i U_i}^{\mathbb{S}}(\varphi_i)]_{i=1}^k$ with respect to $X:\theta(\odot([\varphi_i]_{i=1}^j))$.

We prove completeness of $\mathcal{T}^{\text{cut}}(\mathcal{L}, \Theta)$ by exploiting the completeness of the tableau system $\mathcal{T}(\mathcal{L}, \Theta)$ defined in [45]. Clearly, it is enough to show that all the rules of $\mathcal{T}(\mathcal{L}, \Theta)$ are derivable in $\mathcal{T}^{\text{cut}}(\mathcal{L}, \Theta)$. Closure rules are common to both systems. Thus, we just need to show that it is possible to simulate in $\mathcal{T}^{\text{cut}}(\mathcal{L}, \Theta)$ the branching elimination rules of $\mathcal{T}(\mathcal{L}, \Theta)$, extracted as explained in Section 2. Let us pick one arbitrary such rule

$$X:\theta(\odot([\varphi_i]_{i=1}^k)) \implies || B_X^{\theta \odot}([\varphi_i]_{i=1}^k)$$

where we recall that $B_X^{\theta \odot}([\varphi_i]_{i=1}^k) = \{\&[\overline{v}_i^{\mathbb{S}}(\varphi_i)]_{i=1}^k \mid t(\widehat{\theta}(\widehat{\odot}([v_i]_{i=1}^k))) = X\}$.

Given the root $X:\theta(\odot([\varphi_i]_{i=1}^k))$, we start by using ($\mathcal{L}.\text{Cut}$) to cut on all the immediate Θ -subformulas of $\odot([\varphi_i]_{i=1}^k)$. This will produce $2^{k \cdot \#(\Theta)}$ branches corresponding to all possible combinations of classical values for $\theta_j(\varphi_i)$ with $j = 0, 1, \dots, s$ and $i = 1, \dots, k$. The branches that correspond to combinations that satisfy the head of the rule coincide precisely with the elements of $B_X^{\theta \odot}([\varphi_i]_{i=1}^k)$. Thus we are left with showing that the remaining branches can all be closed. Some of these branches may close simply by means of an application of some ($\mathcal{L}.Ck$) rule because they correspond to combinations that include some unrealizable binary print (as the dashed lines in Table 2). Hence, we only need to analyze what happens with the branches corresponding to valid combinations that assign the value X^C to $\theta(\odot([\varphi_i]_{i=1}^k))$. Consider the sequence of elements in one such branch and take its largest prefix that turns $X:\theta(\odot([\varphi_i]_{i=1}^k))$ satisfiable. It is, of course, a proper prefix. Assume also that $Y:\theta_j(\varphi_i)$ is the next element in the sequence. Clearly, the prefix corresponds to some sequence $[\overline{c}_{i R_i}]_{i=1}^k$ of partial binary prints whose associated rule ($\mathcal{L}.R_X^{\theta \odot}[\overline{c}_{i R_i}]_{i=1}^k$) will produce $Y^C:\theta_j(\varphi_i)$ (or a simpler rule if this one is redundant). Finally, we may close the branch using the rule ($\mathcal{L}.C0$). \square

The strategy used in the completeness proof above is simple but often builds unnecessarily complex tableaux. Below, when we study the proof complexity

of our cut-based systems, we will show that such tableaux can be significantly simplified. In any case, most importantly, the proof of Theorem [1](#) also shows the completeness of the analytic version of our cut-based systems, i.e., a restriction that allows applications of cut only to Θ -subformulas of the formulas occurring in the root of the tree.

Corollary 1. *The analytic restriction of $\mathcal{T}^{\text{cut}}(\mathcal{L}, \Theta)$ is complete.*

In the light of the analyticity result in Corollary [1](#), the cut-based tableau system $\mathcal{T}^{\text{cut}}(\mathcal{L}, \Theta)$ can be used as a decision procedure for the logic \mathcal{L} . Since finite-valued logics are already known to be decidable by the ‘brute force’ truth-table method, it will be interesting to know more about the computational complexity of the decision procedure associated to $\mathcal{T}^{\text{cut}}(\mathcal{L}, \Theta)$. As in the case of the KE system for classical logic (see [6](#)), it is expectable that our cut-based tableaux for finite-valued logics fare significantly better than conventional tableaux in terms of proof complexity, and in general not worse than the truth-table method. We adapt from [8](#) the definition of some typical complexity measures to be used below.

Definition 2. *The size of a tableau π , denoted by $|\pi|$ is the total number of formulas occurring in π . The λ -complexity of a tableau π , denoted by $\lambda(\pi)$, is the number of nodes in π . The ρ -complexity of a tableau π , denoted by $\rho(\pi)$, is the maximum number of formulas in a node of π .*

As an example, for the tableau π in Figure [1](#) we have $|\pi| = 9$, $\lambda(\pi) = 6$ and $\rho(\pi) = 2$. Clearly, the following relation holds in general: $|\pi| \leq \lambda(\pi) \cdot \rho(\pi)$. Note that in the case of a tableau π produced within $\mathcal{T}^{\text{cut}}(\mathcal{L}, \Theta)$, the ρ -complexity of π is bounded by $\rho(\pi) \leq k(s + 1)$, where $s + 1$ is the cardinality of Θ and k is the maximum arity of any connective from the alphabet of \mathcal{L} .

The following theorem shows that the cut-based tableau systems given by Definition [1](#) can polynomially simulate (p-simulate) the truth-table method.

Theorem 2. *Given a valid signed formula $X:\varphi$ of \mathcal{L} with size a and containing v distinct atoms, there is a refutation π of $X^C:\varphi$ in $\mathcal{T}^{\text{cut}}(\mathcal{L}, \Theta)$ of complexity $\lambda(\pi) = O(a \cdot \#(\Theta) \cdot 2^{v \cdot \#(\Theta)})$.*

Proof. First we apply (\mathcal{L} .Cut) to all the atomic Θ -subformulas of φ . This will generate a tree with $2^{v \cdot \#(\Theta)}$ branches. Then, for each such branch, we proceed by applying (\mathcal{L} .Cut) to a Θ -subformula φ_i of φ such that all of its immediate Θ -subformulas are already in the branch. By construction, such a φ_i exists. We note that at least one of the two branches thereby generated gives rise to a contradiction and may be closed by applying at most one elimination rule and one closure rule. Indeed, by the definition of the system, either the system contains an elimination rule for φ_i whose application gives rise to a contradiction on one of the Θ -subformulas of φ_i or, as a trivial case, φ_i is of the form $\theta(\odot(\dots))$ and we can apply a closure rule ($\mathcal{L}.C_X^{\theta(\odot)}$), that is, either $F:\varphi_i \implies *$ or $T:\varphi_i \implies *$. If one of the branches does not close, we can reiterate on it the same procedure,

by applying ($\mathcal{L}.Cut$) to a further Θ -subformula of φ such that all its immediate Θ -subformulas are in the branch.

We conclude by noticing that all the initial $2^{v \cdot \#(\Theta)}$ branches may be closed by following the above described procedure, i.e., by applying ($\mathcal{L}.Cut$) to at most the Θ -subformulas of φ , and so linearly in $a \cdot \#(\Theta)$. \square

We can further show that $\mathcal{T}^{cut}(\mathcal{L}, \Theta)$ is not worse than $\mathcal{T}(\mathcal{L}, \Theta)$. Intuitively, we must be able to reproduce efficiently in $\mathcal{T}^{cut}(\mathcal{L}, \Theta)$ any tableau constructed within $\mathcal{T}(\mathcal{L}, \Theta)$, and in particular more efficiently than we managed to do in the proof of Theorem 1. To illustrate how we proceed, we show in particular how it is possible to efficiently simulate in the cut-based tableau system for L_3 (Section 3) the branching rule for $F:p \supset q$ (Example 2). While the tree on the left of Figure 2 portrays an application of the rule obtained in Section 2, the one on the right represents its efficient simulation by means of rules of the cut-based system. In particular, we use ($L_3.\supset 1$) to derive (2.1) and (2.2); then we cut on p and obtain (3.1) and (3.2); finally, we obtain (4) by using ($L_3.\supset 3$) on (1) and (3.1) and we obtain (5.1) and (5.2) by cutting on $\theta(q)$.

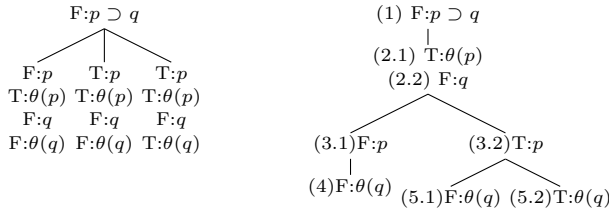


Fig. 2. Finding efficient simulations of branching elimination rules for L_3

The proof of the following theorem uses a similar strategy.

Theorem 3. *For every proof π in the system $\mathcal{T}(\mathcal{L}, \Theta)$, there exists a proof π^{cut} with the same root in the system $\mathcal{T}^{cut}(\mathcal{L}, \Theta)$ such that $|\pi^{cut}| \leq |\pi|$.*

Proof. Building upon the proof of Theorem 1, it is enough to show that each branching elimination rule of $\mathcal{T}(\mathcal{L}, \Theta)$ can be efficiently derived in the cut-based system. Let us consider an arbitrary such rule

$$X:\theta(\odot([\varphi_i]_{i=1}^k)) \implies || B_X^{\theta \odot}([\varphi_i]_{i=1}^k)$$

where $B_X^{\theta \odot}([\varphi_i]_{i=1}^k) = \{ \&[\overline{v}_i^S(\varphi_i)]_{i=1}^k \mid t(\widehat{\theta}(\widehat{\odot}([v_i]_{i=1}^k))) = X \}$, as in Section 2.

Starting with root $X:\theta(\odot([\varphi_i]_{i=1}^k))$, in $\mathcal{T}^{cut}(\mathcal{L}, \Theta)$ we can follow a procedure consisting in applying linear elimination rules for $X:\theta(\odot([\varphi_i]_{i=1}^k))$ whenever possible, or ($\mathcal{L}.Cut$) on some missing Θ -subformula if none of the elimination rules can be applied. It is easy to see that, by construction, the amount of information in the simulating tree is not bigger than the one produced by the given rule, i.e., each formula in such a simulating tree also occurs in at least one branch of the rule. \square

The existence of extremely bad cases, in general, for $\mathcal{T}(\mathcal{L}, \Theta)$ is very likely, although exploring that path lies beyond the scope of this paper. Together with the above results, one would then certainly expect to be able to show, as in the case of classical logic, that the cut-based systems allow in general for a significantly better performance.

5 Conclusions

Other paths could have been explored for defining appropriate cut-based versions of the tableau systems in [4,5]. Yet, we believe that the path explored here achieves a good trade-off between efficiency of proof construction and usability of the system. On what concerns the first aspect, as it is common in this area, we measured efficiency in terms of size of the tableaux produced, by having in mind, as a minimum requirement, that p-simulation of truth-tables must hold. Clearly, the use of a larger number of rules would help in this sense; in particular, we could add a closure rule for each unsatisfiable situation arising from the analysis of truth-tables, as illustrated in Section 3 and formalized in Section 4. This would in principle reduce—but asymptotically not in any significant way—the size of the closed tableaux built as in the proof of Theorem 2, since each unsatisfiable branch could be closed immediately. A further option would consist in allowing only elimination rules such that all the immediate subformulas are involved in the rule, either in the antecedent or in the succedent. As an example, the rule $(L_3.\supset 1)$ would not be allowed in the system of Section 3. The systems resulting from such approach allow for the p-simulation of the truth-table method (the procedure described in the proof of Theorem 2 can still be applied) and have the advantage of facilitating proof search, in the sense that for each formula in a tableau one needs to apply at most one elimination rule. A drawback of such systems is that they tend to require more uses of cut, e.g., the formula in the example of Figure 1 (see Section 3) would not have a linear closed tableau.

On what concerns readability and compactness of the system, we mainly tried to minimize the number of rules and the number of formulas per rule. With such goal in mind, further simplifications could be proposed. As an example, one can notice that the rule $(L_3.\supset 2)$ might be rewritten as

$$F:\varphi_1 \supset \varphi_2 \ \& \ T:\theta(\varphi_2) \quad \Longrightarrow \quad T:\varphi_1$$

since the other formulas in the succedent may be obtained by an application of $(L_3.\supset 1)$. By generalizing such simplifications, one would obtain a more compact system for which, however, the result of Theorem 2 would not hold. Finally, we note that the proof of Theorem 2 suggests a very simple decision procedure, which is enough for p-simulating truth-tables. However, in general there might be better heuristics for guiding the construction of a tableau. For example, the canonical procedure given in [8] for the KE system for classical logic coincides, in essence, with the procedure we adopted in the proof of Theorem 3.

As we have seen, the syntactic encoding of the truth-tabular semantics presupposed by our classic-like approach to cut-based tableaux generates in principle

a multiplication of the number of rules. Moreover, in the resulting tableau systems, rules contain a number of expressions in the antecedent which need to be simultaneously matched to the nodes of a given branch in order to be applied. Even though proof-complexity theorists do not in general take into account the costs implicit in the use of a deductive system with a large number of rules and with rules which require a lot of pattern-matching effort, and we have here done our study in accordance with that tradition, one might also think it wiser to measure such costs in calculating the efficiency of a given proof system.

Though our methods cannot be expected to apply to infinite-valued logics in general, it is predictable that they extend smoothly at least to those infinite-valued logics with a finite-valued non-deterministic semantics [1]. The possible connection between our approach and resolution-based sets-as-signs methods [11] is another interesting topic for future research.

Acknowledgments. The third author thanks the support of FCT and FEDER via the projects PESt-OE/EEI/LA0008/2011 and ComFormCrypt PTDC/EIA-CCO/113033/2009 of IT, and UTAustin/MAT/0057/2008 of IST, as well as of the PQDR and GeTFun initiatives of SGIQ. The authors are indebted to five anonymous referees for their careful reading of an earlier version of this paper.

References

1. Avron, A., Lev, I.: Non-deterministic multiple-valued structures. *Journ. of Logic and Computation* 15, 241–261 (2005)
2. Boolos, G.: Don't eliminate cut. *Journ. of Phil. Logic* 13, 373–378 (1984)
3. Caleiro, C., Carnielli, W., Coniglio, M.E., Marcos, J.: Two's company: The humbug of many logical values. In: Béziau, J.-Y. (ed.) *Log. Universalis*, pp. 169–189. Birkhäuser Verlag, Basel (2005)
4. Caleiro, C., Marcos, J.: Classic-Like Analytic Tableaux for Finite-Valued Logics. In: Ono, H., Kanazawa, M., de Queiroz, R. (eds.) *WoLLIC 2009*. LNCS, vol. 5514, pp. 268–280. Springer, Heidelberg (2009)
5. Caleiro, C., Marcos, J.: Many-valuedness meets bivalence: Using logical values in an effective way. *J. of Multiple-Valued Log. and Soft Comp.* 18 (2012)
6. D'Agostino, M.: Investigations into the complexity of some propositional calculi. PRG Techn. Monogr. 88. Oxford Univ., Computing Lab, Oxford (1990)
7. D'Agostino, M.: Are tableaux an improvement on truth-tables? Cut-free proofs and bivalence. *Journ. of Log., Lang., and Inform.* 1, 235–252 (1992)
8. D'Agostino, M.: Tableau methods for classical propositional logic. In: *Handbook of Tableau Methods*, pp. 45–123. Kluwer Academic Publishers (1999)
9. D'Agostino, M., Mondadori, M.: The taming of the cut: Classical refutations with analytic cut. *Journ. of Log. and Comp.* 4(3), 285–319 (1994)
10. Finger, M., Gabbay, D.: Cut and pay. *Journ. of Logic, Language and Information* 15, 195–218 (2006)
11. Hähnle, R.: *Automated Deduction in Multiple-Valued Logics*. International Series of Monographs on Computer Science, vol. 10. Oxford University Press (1994)
12. Marcos, J., Mendonça, D.: Towards fully automated axiom extraction for finite-valued logics. In: Carnielli, W., et al. (eds.) *The Many Sides of Logic*, pp. 425–440. College Publications, London (2009)

Author Index

- Abriola, Sergio 110
Adsul, Bharat 291
Ahrens, Benedikt 127
Areces, Carlos 142
Asperti, Andrea 1
- Baldi, Paolo 154
Baltag, Alexandru 168
Bezhanishvili, Nick 191
- Caleiro, Carlos 321
Callejas, Claudio 206
Callejas Bedregal, Benjamín René 206
Chakraborty, Supratik 291
Ciabattini, Agata 154
Coniglio, Marcelo E. 268
- Díaz-Caro, Alejandro 216
Diekert, Volker 70
- Farjudian, Amin 232
Fervari, Raul 142
Figueira, Santiago 110
Fokina, Ekaterina 26
Fouché, Willem L. 246
Friedman, Sy 26
- Hoffmann, Guillaume 142
- Kallmeyer, Laura 34
Kamath, Pritish 291
Kupke, Clemens 191
- Link, Sebastian 257
- Madan, Vivek 291
Marcos, João 206, 321
Menasché Schechter, Luis 306
Metcalf, George 56
Muscholl, Anca 70
- Nies, André 26
- Osswald, Rainer 34
- Panangaden, Prakash 191
Petit, Barbara 216
- Renne, Bryan 168
Ribeiro, Márcio M. 268
Ricciotti, Wilmer 1
Rossman, Benjamin 282
- Sankaran, Abhisekh 291
Schweikardt, Nicole 85
Selinger, Peter 88
Senno, Gabriel 110
Smets, Sonja 168
Spendier, Lara 154
- van Ditmarsch, Hans 89
Volpe, Marco 321