Circularity in Syntax and Semantics

Programme and Book of Abstracts

University of Gothenburg
20–22 November, 2019

UNIVERSITY OF GOTHENBURG

Programme

Wednesday, 20 November

9.00 *Registration*

9.30 **Mai Gehrke**, *Equations as a Tool for Studying Logic Fragments*

Coffee break

11.00 **Sebastian Enqvist and Valentin Goranko**, *A Temporal Logic for Concurrent Coalitional Strategies in Multi-player Games*

11.40 **Michał Walicki**, *Circularity in Graph Normal Form*

12.20 **Clemens Kupke, Johannes Marti and Yde Venema**, *Size Matters in the Modal Mu-calculus*

Lunch

14.30 **Mads Dam**, *On First-order μ-calculus as a Language Independent Program Verification Framework*

Coffee break

16.00 **Antti Kuusisto**, *A Turing-complete Extension of First-order Logic*

16.40 **Paulo Guilherme Santos and Reinhard Kahle**, *Yablo's Paradox Revisited*

17.20 **Mattias Granberg Olsson**, *Truth and Fix-points for Almost Negative Formulae*

19.00 *Public Lindström Lecture:* **Johan van Benthem**, *Logic and Agency: The Promises and The Challenges*

Reception

Thursday, 21 November

9.30 **Helle Hvid Hansen**, *Complete Proof Systems for Parikh's Game Logic*

Coffee break

11.00 **Daniyar Shamkanov**, *Non-well-founded Derivations in the Gödel-Löb Provability Logic*

11.40 **Sonia Marin**, *On Cut-elimination for Non-wellfounded Proofs: The Case of PDL*

12.20 **Malvin Gattinger and Yde Venema**, *Interpolation for PDL: An Open Problem?*

Conference photo & lunch

14.30 *Research Lindström Lecture:* **Johan van Benthem**, *Dynamic Logics of Modal Change*

Coffee break

16.00 **Anupam Das**, *Project Announcement: Structure vs Invariants in Proofs (StrIP)*

17.00 *Panel discussion*

19.00 *Conference dinner at Ågrenska villan (Högåsplatsen 2, 412 56 Göteborg)*

Friday, 22 November

9.30 **Paul-André Melliès**, *Higher-order Parity Automata*

Coffee break

11.00 **Sorin Stratulat**, *Efficient Validation of FOL_{ID} Cyclic Induction Reasoning*

11.40 **Abhishek De, Luc Pellissier and Alexis Saurin**, *Towards Circular Proof Nets*

12.20 **Daichi Hayashi**, *On Friedman-Sheard Theories for Recursive Realizability*

Lunch

14.30 **Amina Doumane**, *Bouncing Threads for Infinitary and Circular Proofs*

Coffee break

16.00 **Andreas Abel**, *Resolving the Circularity of Infinite Processes via Sized Coinductive Types*

16.40 **Paul Kindvall Gorbow and Graham E. Leigh**, *The Reflective Multiverse of Set Theory*

17.20 **Stepan Kuznetsov**, *Half a Way Towards Circular Proofs for Kleene Lattices*

Closing

Abstracts

Resolving the Circularity of Infinite Processes via Sized Coinductive Types

Andreas Abel

Department of Computer Science and Engineering
Chalmers and Gothenburg University
andreas.abel@cse.gu.se

The greatest fixed point of a monotone function on subsets of a totality can be constructed by iterating the function on the totality until it stabilizes to the fixed point. Dually, the least fixed point can be obtained by iteration starting with the empty set. Naming the approximation stages and making them available as types is the fundamental idea of *sized types* [15, 7, 13, 8, 1, 10, 2, 5, 11, 16], the extension of conventional type theories and systems by sized (co)inductive types, meaning (co)inductive types indexed by the approximation stage. The indices are drawn from an abstract type of sizes whose denotation are ordinals up to a certain closure ordinal.

Sized types have been anecdotally successful [4, 12] to model coinductive processes more directly as via the coinduction principle drawn from, e.g., the concept of terminal coalgebra in category theory. Sized types have become part of the type-theoretic proof assistant Agda [3, 6] and there is ongoing work to integrate them into the Coq proof assistant [9, 14, 17, 18].

In this talk, I will explain the principles behind sized coinductive types and their realization in the Agda proof assistant. Time permitting, I will allow myself some speculation about why coinduction can be naturally reduced to induction, but not vice versa.

[1] A. Abel. Termination checking with types. *RAIRO – Theoretical Informatics and Applications*, 38(4):277–319, 2004. URL http://www.edpsciences.org/articles/ita/abs/2004/04/ita0428NS/ita0428NS.html. Special Issue: Fixed Points in Computer Science (FICS'03).

[2] A. Abel. *Type-Based Termination.* Harland media, 2007. URL http://www.harland-media.de/buch.php?ISBN=978-3-938363-04-1. Dissertation, Department of Computer Science, Ludwig-Maximilians-University Munich, 2006, titled *A Polymorphic Lambda Calculus with Sized Higher-Order Types.*

[3] A. Abel. MiniAgda: Integrating sized and dependent types. In A. Bove, E. Komendantskaya, and M. Niqui, editors, *Wksh. on Partiality And Recursion in Interactive Theorem Provers, PAR 2010*, volume 43 of *Electr. Proc. in Theor. Comp. Sci.*, pages 14–28, 2010. URL http://dx.doi.org/10.4204/EPTCS.43.

[4] A. Abel and J. Chapman. Normalization by evaluation in the delay monad: A case study for coinduction via copatterns and sized types. In P. Levy and N. Krishnaswami, editors, *Proc. 5th Wksh. on Mathematically Structured Functional Programming, MSFP 2014*, volume 153 of *Electr. Proc. in Theor. Comp. Sci.*, pages 51–67, 2014. URL http://dx.doi.org/10.4204/EPTCS.153.4.

[5] A. Abel and B. Pientka. Well-founded recursion with copatterns and sized types. *J. Func. Program.*, 26:61, 2016. URL http://dx.doi.org/10.1017/S0956796816000022. ICFP 2013 special issue.

[6] A. Abel, A. Vezzosi, and T. Winterhalter. Normalization by evaluation for sized dependent types. *Proc. of the ACM on Program. Lang.*, 1(ICFP):33:1–33:30, 2017. URL http://doi.acm.org/10.1145/3110277.

[7] R. M. Amadio and S. Coupet-Grimal. Analysis of a guard condition in type theory (extended abstract). In M. Nivat, editor, *Proc. of the 1st Int. Conf. on Foundations of Software Science and Computation Structure, FoSSaCS'98*, volume 1378 of *Lect. Notes in Comput. Sci.*, pages 48–62. Springer, 1998. URL http://dx.doi.org/10.1007/BFb0053541.

[8] G. Barthe, M. J. Frade, E. Giménez, L. Pinto, and T. Uustalu. Type-based termination of recursive definitions. *Math. Struct. in Comput. Sci.*, 14(1):97–141, 2004. URL http://dx.doi.org/10.1017/S0960129503004122.

[9] G. Barthe, B. Grégoire, and F. Pastawski. CIC$\hat{}$: Type-based termination of recursive definitions in the Calculus of Inductive Constructions. In M. Hermann and A. Voronkov, editors, *Proc. of the 13th Int. Conf. on Logic for Programming, Artificial Intelligence, and Reasoning, LPAR 2006*, volume 4246 of *Lect. Notes in Comput. Sci.*, pages 257–271. Springer, 2006. URL https://doi.org/10.1007/11916277_18.

[10] F. Blanqui. A type-based termination criterion for dependently-typed higher-order rewrite systems. In V. van Oostrom, editor, *Rewriting Techniques and Applications, RTA 2004, Aachen, Germany*, volume 3091 of *Lect. Notes in Comput. Sci.*, pages 24–39. Springer, 2004. URL https://doi.org/10.1007/978-3-540-25979-4_2.

[11] F. Blanqui. Size-based termination of higher-order rewriting. *J. Func. Program.*, 28:e11, 2018. URL https://doi.org/10.1017/S0956796818000072.

[12] N. A. Danielsson. Up-to techniques using sized types. *Proc. of the ACM on Program. Lang.*, 2(POPL):43:1–43:28, 2018. URL https://doi.org/10.1145/3158131.

[13] E. Giménez. Structural recursive definitions in type theory. In K. G. Larsen, S. Skyum, and G. Winskel, editors, *Int. Colloquium on Automata, Languages and Programming (ICALP'98), Aalborg, Denmark*, volume 1443 of *Lect. Notes in Comput. Sci.*, pages 397–408. Springer, 1998. URL https://doi.org/10.1007/BFb0055070.

[14] B. Grégoire and J. L. Sacchini. On strong normalization of the calculus of constructions with type-based termination. In C. G. Fermüller and A. Voronkov, editors, *Proc. of the 17th Int. Conf. on Logic for Programming, Artificial Intelligence, and Reasoning, LPAR-17, 2010*, volume 6397 of *Lect. Notes in Comput. Sci.*, pages 333–347. Springer, 2010. URL http://dx.doi.org/10.1007/978-3-642-16242-8_24.

[15] J. Hughes, L. Pareto, and A. Sabry. Proving the correctness of reactive systems using sized types. In H.-J. Boehm and G. L. Steele Jr., editors, *Proc. of the 23rd ACM Symp. on Principles of Programming Languages, POPL'96*, pages 410–423. ACM Press, 1996. URL http://doi.acm.org/10.1145/237721.240882.

[16] U. D. Lago and C. Grellois. Probabilistic termination by monadic affine sized typing. *ACM Trans. on Program. Lang. and Syst.*, 41(2):10:1–10:65, 2019. URL https://doi.org/10.1145/3293605.

[17] J. L. Sacchini. Type-based productivity of stream definitions in the calculus of constructions. In *28th ACM/IEEE Symp. on Logic in Computer Science (LICS'13)*,

pages 233–242. IEEE Computer Soc. Press, 2013. URL http://dx.doi.org/10.1109/LICS.2013.29.

[18] J. L. Sacchini. Linear sized types in the calculus of constructions. In M. Codish and E. Sumii, editors, *Proc. of the 12th Int. Symp. on Functional and Logic Programming, FLOPS 2014*, volume 8475 of *Lect. Notes in Comput. Sci.*, pages 169–185. Springer, 2014. URL http://dx.doi.org/10.1007/978-3-319-07151-0_11.

Logic and Agency: The Promises and the Challenges
Public Lindström Lecture

Johan van Benthem
University of Amsterdam
j.vanbenthem@uva.nl

One face of logic is turned toward truth and eternal consequence, but another face is dynamic, looking toward activities of reasoning and information handling by agents. In this lecture, I will develop the dynamic perspective, showing how key aspects of rational agency fit in the agenda of logic, such as handling and integrating information from various sources, revising erroneous beliefs, and balancing information with preferences and goals. In this lecture, the vehicle for achieving this will be dynamic-epistemic logics for various sorts of update. These mesh eventually with richer logics of games and strategic interaction. After all, much of reasoning is a social multi-agent process: 'intelligence seldom comes alone'. I end this part by noting how the two faces of logic share the same methodology, and are in fact complementary.

However, 'more logic' is just one way to go in studying agency. I briefly discuss current challenges from the 'less logic' camp, where successful behavior is explained by dynamical systems with very simple agents, or from learning systems that may not have a logical formulation at all. I hope to show that logic retains a valuable role even in that stormy setting.

[1] J. van Benthem. *Logical Dynamics of Information and Interaction.* Cambridge University Press, 2011.
[2] J. van Benthem. *Logic in Games.* The MIT Press, 2014.
[3] J. van Benthem. Fanning the Flames of Reason. *Valedictory lecture University of Amsterdam*, 2015.

Dynamic Logics of Model Change

Research Lindström Lecture

Johan van Benthem
University of Amsterdam
j.vanbenthem@uva.nl

Information update and real-world action suggest a universe of changing models, where finding the valid dynamic laws involves logical languages with modalities for model change. I will discuss two families of such dynamic modal logics, stating some typical results, as well as open problems.

Dynamic-epistemic logics add dynamic superstructure to existing static logics, and tend to not increase complexity of satisfiability and model checking. Logics of graph change, arising e.g. in the study of games or of languages that change their own models under evaluation, tend to jump to higher complexities, becoming undecidable or worse. I will report some recent results that zoom in on the border line between the two kinds of system, and end with some issues about redesign and restoring decidability for more complex logics of action and information dynamics.

[1] J. van Benthem. *Logical Dynamics of Information and Interaction.* Cambridge University Press, 2011.
[2] J. van Benthem, Ch. Mierzewski, and F. Zaffora Blando. *The Modal Logic of Stepwise Removal.* Stanford University, 2019.
[3] J. van Benthem and F. Liu. *Logic and Design of Graph Games.* Tsinghua University, 2019.

On First-Order μ-Calculus as a Language-Independent Program Verification Framework

Mads Dam
KTH Royal Institute of Technology, Stockholm, Sweden
mfd@kth.se

The μ-calculus in its various instantiations (modal, relational, first-order) is of interest in a computer science context due to its ability to capture general combinations of inductive and co-inductive properties in an elegant and uniform fashion, allowing to express a rich collection of temporal properties [6]. Much interest in the μ-calculus has stemmed from the development of tableaux-based local model checking algorithms by Stirling and others in the late '80s. In model checking the focus is on verification of closed, i.e. fully determined systems p against some logical specification. For compositional reasoning model structure is essential: The p's in this case are terms in some programming or process specification language, and the goal is enable the verification of a property ϕ against some composite term $f(p_1, \ldots, p_n)$ to be decomposed into its constituent parts, $p_1, \ldots, p_n$. Closed system model checking by itself is not capable of supporting such compositional verification, however, and more tools are needed, not least to support concurrency. A number of authors began to address this problem in the late '80s/early '90s [19, 1, 16] in various ways, for instance by means of an auxiliary structural entailment relation

$$\phi_1, \ldots, \phi_n \vdash \phi \tag{1}$$

with the intended meaning "for all $i \in [1, ..., n]$, if $p_i \models \phi_i$ then $f(p_1, \ldots, p_n) \models \phi$". Observe that for f a binary operation, say $\cdot$, this auxiliary relation becomes familiar from Kripke-style models for linear and relevant logics, explored for concurrent systems first in [9].

The paper [10] is the first in a series of works examining various generalisations of (1) as a framework for compositional program verification. In [10] the focus is on single-sided sequents of the form $x_1 : \phi_1, \ldots, x_n : \phi_n \vdash p : \phi$ where ϕ and the ϕ_i are L_μ formulas and p a CCS term. Later [13] this is generalised to sequents of the shape $\Gamma \vdash \Delta$ where the elements of Γ (Δ) are correctness assertions of the shape $p : \phi$ and where p is a process term possibly involving free variables and ϕ an L_μ formula, and in [18] to the first-order μ-calculus, in this way developing a framework that can be instantiated to, in principle, any first-order programming language. In the presentation we review some of this work and its applications in light of later developments in the area.

We cover two problem areas:

1. The development of effective proof mechanisms based on circular reasoning.
2. Tool support, applications, and instantiations to different modelling and programming languages.

The key to building sound circular proof systems in [13] was to augment L_μ with an explicit treatment of ordinal variables. This use of ordinal variables is related to Stirling's use of unique naming to keep track of formula unfoldings [6]. Together

with an infinitary well-foundedness condition, this allowed to obtain soundness and completeness through reduction to Kozen's proof system. In [17] a very general finitary graph-based discharge condition is introduced similar to the tree-based condition introduced earlier in [13] and the discharge condition is reformulated in terms of tree automata. The graph-based discharge condition is able, under a suitable set of conditions, to discharge a leaf in the proof structure against any node in the proof graph, not only ancestors in a tree structure. This allows proofs to be more compactly represented.

In [18] a non-circular ("classical") proof system is obtained by replacing the discharge condition by conditions formalising well-founded induction. It is shown that these two proof systems have identical proof power. Brotherston and Simpson studied this question in the context of a cyclic version CLKID$^\omega$ of Martin-Löf's LKID calculus of inductive definitions and gave a partial answer [8]. This was fully resolved in the affirmative in the presence of arithmetic by Berardi and Tatsuta [5] and in the negative in general by the same authors in [4] where a sentence 2-Hydra is exhibited that is provable in CLKID$^\omega$ but not in LKID.

The proof system in [18] was used in the Erlang Verification Tool, EVT[1], [15] as a basis for an interactive theorem prover for the Erlang programming language. The similar tool CYCLIST by Brotherston and colleagues is reported in [7]. The difficulty in implementing theorem provers based on circular reasoning is that it requires at least a subset of the internal nodes in the putative proof graph under construction to be explicitly represented, leading to exponential space complexity in the worst case. This also makes it challenging to embed the circular proof system in existing, mature theorem provers such as HOL4, Isabelle, or Coq that only support trees and store only unreduced subgoals, leading both EVT and later CYCLIST to implement their theorem provers from scratch.

The promise of circular verification is that, if proof search succeeds, "messy" induction details can potentially be completely hidden from users. Indeed, it is not difficult to embed fully automatic proof search strategies, for instance for closed system model checking, in this manner as tactics in a tool such as EVT. On the other hand, the global nature of the discharge rule can make proof search highly challenging. We report on some case studies performed in EVT, including a study involving an Erlang-based distributed database manager [3].

Despite its name, the EVT tool is not strongly tied to the Erlang language. In fact, exploiting the FOL base it is quite easy to embed a standard operational semantics. For instance a simplified state transition rule for the Erlang process spawning construct might look like the following:

$$\frac{-}{(\mathsf{spawn}(p_1); p_2, q) \rightarrow (p_2, q)||(p_1, \varepsilon)}$$

The rule should be read as follows: A single process ready to spawn p_1, with continuation p_2 and input queue state q can in one step cause p_1 to be spawned with an empty input queue, running in parallel with p_2 with input queue q. This rule is easily represented in EVT as a base case in an inductive (least fixed point) definition of the transition relation $\rightarrow$. In a similar vein, many of the datatypes used

[1] Also called VCPT and VeriCode in later instances.

can be represented as inductive definitions as well. In this manner a number of instantiations of the EVT tool has been created, for languages such as CCS [12], the π-calculus [11, 2], and System F with subtyping (part of the POPLmark challenge, `https://www.seas.upenn.edu/ plclub/poplmark/`) [14]. The latter two, in particular, uses deep embeddings and are interesting as exercises in how far circular reasoning based on FOL can be taken as a framework for languages involving binding.

[1] Henrik Reif Andersen, Colin Stirling, and Glynn Winskel. A compositional proof system for the modal mu-calculus. In *Proceedings of the Ninth Annual Symposium on Logic in Computer Science (LICS '94), Paris, France, July 4-7, 1994*, pages 144–153, 1994.

[2] Erik Angelin. Implementing the pi-calculus in the VeriCode proof tool. Technical Report TRITA-CSC-E 2006:133, KTH Royal Institute of Technology, 2006.

[3] Thomas Arts and Mads Dam. Verifying a distributed database lookup manager written in erlang. In *FM'99 - Formal Methods, World Congress on Formal Methods in the Development of Computing Systems, Toulouse, France, September 20-24, 1999, Proceedings, Volume I*, pages 682–700, 1999.

[4] Stefano Berardi and Makoto Tatsuta. Classical system of martin-löf's inductive definitions is not equivalent to cyclic proof system. In *Foundations of Software Science and Computation Structures - 20th International Conference, FOSSACS 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings*, pages 301–317, 2017.

[5] Stefano Berardi and Makoto Tatsuta. Equivalence of inductive definitions and cyclic proofs under arithmetic. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, June 20-23, 2017*, pages 1–12, 2017.

[6] Julian C. Bradfield and Colin Stirling. Modal mu-calculi. In *Handbook of Modal Logic.*, pages 721–756. 2007.

[7] James Brotherston, Nikos Gorogiannis, and Rasmus Lerchedahl Petersen. A generic cyclic theorem prover. In *Programming Languages and Systems - 10th Asian Symposium, APLAS 2012, Kyoto, Japan, December 11-13, 2012. Proceedings*, pages 350–367, 2012.

[8] James Brotherston and Alex Simpson. Complete sequent calculi for induction and infinite descent. In *22nd IEEE Symposium on Logic in Computer Science (LICS 2007), 10-12 July 2007, Wroclaw, Poland, Proceedings*, pages 51–62, 2007.

[9] Mads Dam. Relevance logic and concurrent composition. In *Proceedings of the Third Annual Symposium on Logic in Computer Science (LICS '88), Edinburgh, Scotland, UK, July 5-8, 1988*, pages 178–185, 1988.

[10] Mads Dam. Proving properties of dynamic process networks. *Inf. Comput.*, 140(2):95–114, 1998.

[11] Mads Dam. Proof systems for pi-calculus logics. In *Logic for Concurrency and Synchronisation*, Trends in Logic, pages 145–214. Logica Library, Kluwer Academic Publishers, 2002.

[12] Mads Dam and Dilian Gurov. Compositional verification of CCS processes. In *Perspectives of System Informatics, Third International Andrei Ershov Memorial Conference, PSI'99, Akademgorodok, Novosibirsk, Russia, July 6-9, 1999, Proceedings*, pages 247–256, 1999.

[13] Mads Dam and Dilian Gurov. μ-calculus with explicit points and approximations. *J. Log. Comput.*, 12(2):255–269, 2002.

[14] Marco Diciolla. A solution of POPLMark challenge with vcpt. Technical Report TRITA-CSC-E 2010:112, KTH Royal Institute of Technology, 2010.

[15] Lars-Åke Fredlund, Dilian Gurov, Thomas Noll, Mads Dam, Thomas Arts, and Gennady Chugunov. A verification tool for ERLANG. *STTT*, 4(4):405–420, 2003.

[16] Dilian Gurov, Sergey Berezin, and Bruce M. Kapron. A modal mu-calculus and a proof system for value passing processes. *Electr. Notes Theor. Comput. Sci.*, 5:47, 1996.

[17] Christoph Sprenger and Mads Dam. On global induction mechanisms in a μ-calculus with explicit approximations. *ITA*, 37(4):365–391, 2003.

[18] Christoph Sprenger and Mads Dam. On the structure of inductive reasoning: Circular and tree-shaped proofs in the μ-calculus. In *Foundations of Software Science and Computational Structures, 6th International Conference, FOSSACS 2003 Held as Part of the Joint European Conference on Theory and Practice of Software, ETAPS 2003, Warsaw, Poland, April 7-11, 2003, Proceedings*, pages 425–440, 2003.

[19] Colin Stirling. A complete compositional model proof system for a subset of CCS. In *Automata, Languages and Programming, 12th Colloquium, Nafplion, Greece, July 15-19, 1985, Proceedings*, pages 475–486, 1985.

Structure vs. Invariants in Proofs (StrIP)

Project Announcement

Anupam Das
University of Birmingham
A.Das@bham.ac.uk

One of the biggest successes of non-wellfounded proof theory is an emerging correspondence between **proofs** and **automata**. This is reflected in many aspects of non-wellfounded systems: correctness reduces to universality of infinite word automata, while automaton-theoretic decidability algorithms are often reflected in completeness arguments. More generally one could imagine identifying the computational content of cyclic proofs with some automaton model. Relevant recent works in this direction include [1, 3, 4].

In this high-level talk I will propose a research project to make these correspondences formal. In particular, I will propose that *alternation* in automaton models is faithfully captured by *deep inference* reasoning on the proof theory side. This yields a richer class of cyclic systems where correctness criteria are more fine grained and proofs are richer. These correspondences are summarised in the following table:[1]

Model	Büchi condition	parity condition
non-deterministic	cyclic proofs (μ)	cyclic proofs (μ, ν)
deterministic	'normal' cyclic proofs (μ)	'normal' cyclic proofs (μ, ν)
alternating	deep cyclic proofs (μ)	deep cyclic proofs (μ, ν)

This talk is based on a recently funded grant proposal of the same name [2].

[1] James Brotherston. Cyclic proofs for first-order logic with inductive definitions. In *Proceedings of TABLEAUX 2005*, pages 78–92, 2005.

[2] Anupam Das. Structure vs. Invariants in Proofs. UKRI Future Leaders Fellowship proposal, 2018. http://www.anupamdas.com/strip.pdf.

[3] Anupam Das and Damien Pous. A cut-free cyclic proof system for Kleene algebra. In *Proceedings of 26th TABLEAUX*, pages 261–277, 2017.

[4] Amina Doumane. Constructive completeness for the linear-time μ-calculus. In *Proceedings of LICS 2017*, pages 1–12, 2017.

[1] The labels μ and ν indicate whether the underlying logic has operators for least and greatest fixed points, respectively.

On Cut-elimination for Non-wellfounded Proofs: The Case of PDL

Anupam Das[a], Rajeev Goré[b] and Sonia Marin[c]

[a]University of Birmingham
A.Das@bham.ac.uk
[b]Research School of Computer Science, Australian National University, Australia
Rajeev.Gore@anu.edu.au
[c]IT University of Copenhagen
sonm@itu.dk

The importance of non-wellfounded proofs is now established; they have been used, for example, to design proof systems for first-order logic with inductive definitions [4] , for proofs of program termination in separation logic [3], for modal logics with fixed-points [1], etc. However, the issue of cut-elimination for such non-wellfounded proofs has not yet found a satisfactory answer. The goal of this line of work is to provide a general toolbox for cut-elimination in non-wellfounded systems, as well as understanding the computational content of cyclic proofs.

We consider here *Propositional Dynamic Logic* (PDL) as a case study, since we observed that the various techniques for cut-elimination in non-wellfounded proofs developed so far [10, 5, 9, 2, 11, 6] all seem to fail to extend to the case of PDL. Indeed, it presents a non-trivial interaction of the two key issues that arise when considering cut-elimination for non-wellfounded proofs, namely cuts on *modalities* and cuts on *fixed points*.

The closest approach to ours is the one of [6], that provides a general *computational interpretation* of cyclic proofs to calculate finite invariants of cut-elimination. However, it only works in a *constructive* setting, so a key issue here is to extend this way of reducing infinitary cut-elimination to finitary cut-elimination to the classical setting of PDL, where we cannot exploit constructivity.

PDL Propositional dynamic logic (PDL) is a logic designed for reasoning about programs [8]. The set of *formulae*, written ϕ, ψ etc., and the set of *actions*, written e, f etc., are generated mutually recursively from atomic actions a, b etc. and propositional variables p, q etc. as follows:

$$\begin{array}{lrcl} \text{Actions:} & e & ::= & a \mid (e \cdot e) \mid e^* \mid \phi? \\ \text{Formulae:} & \phi & ::= & p \mid \overline{p} \mid \langle e \rangle \phi \mid [e]\,\phi \end{array}$$

General negation $\overline{\phi}$ of a formula ϕ is defined via De Morgan duality. Other propositional connectives $\wedge, \vee, \rightarrow$ and units can be recovered by defining them using tests (see axiomatisation).

PDL: Semantics We consider usual *relational structures* of the form (W, R), where W is a non-empty set of *worlds* and R associates to any atomic action a its interpretation as a binary relation $R_a \subseteq (W \times W)$. A *model* is a relational structure equipped with a *valuation* $V : \mathsf{Prop} \rightarrow W$.

The $\vDash$ and $\to$ relations are defined by mutual induction, as extension of V and R respectively, to give the meanings of compound actions and formulae.

$$
\begin{array}{lll}
\mathfrak{A}, x \vDash p & \text{iff} & x \in V(p). \\
\mathfrak{A}, x \vDash \overline{p} & \text{iff} & x \notin V(p). \\
\mathfrak{A}, x \vDash \langle e \rangle \phi & \text{iff} & \text{there exists } y \in W \text{ s.t. } x \xrightarrow{e} y \text{ and } \mathfrak{A}, y \vDash \phi. \\
\mathfrak{A}, x \vDash [e]\, \phi & \text{iff} & \text{for all } y \in W, \text{ if } x \xrightarrow{e} y, \text{ then } \mathfrak{A}, y \vDash \phi. \\
x \xrightarrow{a} y & \text{iff} & (x, y) \in R_a. \\
x \xrightarrow{e \cdot f} y & \text{iff} & \text{there exists } z \in W \text{ such that } x \xrightarrow{e} z \text{ and } z \xrightarrow{f} y. \\
x \xrightarrow{e^*} y & \text{iff} & \text{there exists } n \in \mathbb{N} \text{ and there exist } z_0, \ldots, z_n \in W \text{ such that} \\
 & & z_0 = x,\ z_n = y, \text{ and for all } 0 \leq i \leq n,\ z_i \xrightarrow{e} z_{i+1}. \\
x \xrightarrow{\phi?} y & \text{iff} & x = y \text{ and } \mathfrak{A}, x \vDash \phi.
\end{array}
$$

The *truth* of a formula is then evaluated in a model at a given world and ϕ is said to be *valid* if for any model $\mathfrak{A} = (W, R, V)$ and any world x in W, $\mathfrak{A}, x \vDash \phi$.

PDL: Axiomatisation The Hilbert calculus HPDL for PDL is given by any sound and complete Hilbert calculus system for classical propositional logic, the necessitation rule for any e, and the following axiom schemata:

$$
\mathsf{nec}\ \frac{\phi}{[e]\,\phi}
\qquad
\begin{array}{ll}
[e]\,(\phi \to \psi) \to ([e]\,\phi \to [e]\,\psi) & [e_0 e_1]\,\phi \leftrightarrow [e_0]\,[e_1]\,\phi \\
[e^*]\,\phi \leftrightarrow (\phi \vee [e]\,[e^*]\,\phi) & [\phi_0?]\,\phi_1 \leftrightarrow (\phi_0 \to \phi_1) \\
\multicolumn{2}{c}{[e^*]\,(\phi \to [e]\,\phi) \to (\phi \to [e^*]\,\phi)}
\end{array}
$$

PDL: Sequent calculus So far, proof-theoretic treatments of PDL which enjoy a syntactic cut-elimination rely on an "omega rule" with an infinite number of premises. We propose to use *non-wellfounded proofs* instead. (A similar non-wellfounded *labelled* system has been proposed independently in [7], but without cut-elimination, or even a cut-free completeness result.)

The sequent system LPDL for PDL is given by the following rules comprising identity, cut, atomic rule k_a for each action a, and program rules:

$$
\begin{array}{ccc}
\mathsf{id}\ \dfrac{}{\Gamma, p, \overline{p}} & \mathsf{cut}\ \dfrac{\Gamma, \overline{\phi} \quad \Delta, \phi}{\Gamma, \Delta} & \mathsf{k}_a\ \dfrac{\phi, \Gamma}{[a]\,\phi, \langle a \rangle \Gamma, \Delta} \\[2ex]
\langle \cdot \rangle\ \dfrac{\Gamma, \langle e_0 \rangle \langle e_1 \rangle \phi}{\Gamma, \langle e_0 e_1 \rangle \phi} & \langle * \rangle\ \dfrac{\Gamma, \phi, \langle e \rangle \langle e^* \rangle \phi}{\Gamma, \langle e^* \rangle \phi} & \langle ? \rangle\ \dfrac{\Gamma, \phi \quad \Gamma, \psi}{\Gamma, \langle \phi? \rangle \psi} \\[2ex]
[\cdot]\ \dfrac{\Gamma, [e_0]\,[e_1]\,\phi}{\Gamma, [e_0 e_1]\,\phi} & [*]\ \dfrac{\Gamma, \phi \quad \Gamma, [e]\,[e^*]\,\phi}{\Gamma, [e^*]\,\phi} & [?]\ \dfrac{\Gamma, \overline{\phi}, \psi}{\Gamma, [\phi?]\,\psi}
\end{array}
$$

A (non-wellfounded) *preproof* is an infinite binary tree where nodes are sequents and edges are rules such that, locally, each node is the conclusion of a rule instance for which its children are premisses.

Along an infinite branch of a preproof, a *trace* is a maximal path in the graph of (immediate) ancestry, restricted to that branch.

A (non-wellfounded) *proof* is a preproof where, along each infinite branch, there is a (infinite) trace that eventually hits a $[*]$ formula for which it is infinitely often principal and always follows the right premiss.

A *cyclic proof* is a proof that is *regular*, i.e. has only finitely many distinct subtrees.

Theorem 1 (Soundness and completeness with cut). *A formula ϕ is valid if and only if ϕ has a (non-wellfounded) proof with* cut.

Cut-elimination method To produce an infinite cut-free proof, we must show that we may produce proofs with arbitrarily large cut-free prefixes in a continuous manner. To highlight why issues may occur here, consider the following *key cut case* on fixed points:

$$\dfrac{\dfrac{\Gamma, \phi, \langle a\rangle\langle a^*\rangle\phi}{\Gamma, \langle a^*\rangle\phi}\langle *\rangle \quad \dfrac{\Gamma, \bar{\phi} \quad \Gamma, [a]\,[a^*]\,\bar{\phi}}{\Gamma, [a^*]\,\bar{\phi}}[*]}{\Gamma}\mathsf{cut} \quad \rightsquigarrow \quad \dfrac{\dfrac{\Gamma, \phi, \langle a\rangle\langle a^*\rangle\phi \quad \Gamma, \bar{\phi}}{\Gamma, \langle a\rangle\langle a^*\rangle\phi}\mathsf{cut} \quad \Gamma, [a]\,[a^*]\,\bar{\phi}}{\Gamma}\mathsf{cut}$$

where we can observe that (*i*) only cut-rules are produced and (*ii*) the application of the $[*]$ rule disappears.

Hence, the first fundamental issue is to ensure that the cut-reduction process is *productive* i.e. that it eventually produces non-cut steps and hence actually produces an infinite cut-free preproof in the limit, and not just diverges. We then also need to make sure that this limit remains a proof, i.e. that the *correctness* condition on the traces is preserved.

Thus, to be productive, our cut-elimination strategy proceeds according to the following independent three steps:

1. modal cut-elimination
   $\Rightarrow$ easy productivity as the k_a-rule is preserved by the following reduction:
$$\dfrac{\dfrac{\phi, \Gamma_1}{[a]\,\phi, \langle a\rangle\Gamma_1, \Delta_1}\mathsf{k}_a \quad \dfrac{\bar{\phi}, \Gamma_2, \psi}{\langle a\rangle\bar{\phi}, \langle a\rangle\Gamma_2, [a]\,\psi, \Delta_2}\mathsf{k}_a}{\langle a\rangle\Gamma_1, \Delta_1, \langle a\rangle\Gamma_2, [a]\,\psi, \Delta_2}\mathsf{cut} \quad \rightsquigarrow \quad \dfrac{\dfrac{\phi, \Gamma_1 \quad \bar{\phi}, \Gamma_2, \psi}{\Gamma_1, \Gamma_2, \psi}\mathsf{cut}}{\langle a\rangle\Gamma_1, \Delta_1, \langle a\rangle\Gamma_2, [a]\,\psi, \Delta_2}\mathsf{k}_a$$
2. "at world" fixed-point cut-elimination
   $\Rightarrow$ using a partial computational interpretation inspired by the *witness function method* and Kreisel's *no-counterexample* interpretation
3. atomic/propositional cut-elimination $\Rightarrow$ standard

At the end of this process, an external proof of preservation of correctness is also required, which completes the cut-elimination result.

Theorem 2 (Cut-elimination). *For any cyclic proof with cuts, there is a cut-free proof of the same theorem.*

[1] Bahareh Afshari and Graham E. Leigh. Cut-free completeness for modal mu-calculus. In *Proceedings of LICS 2017*, 2017.

[2] David Baelde, Amina Doumane, and Alexis Saurin. Infinitary proof theory: the multiplicative additive case. In *Proceedings of CSL 2016*, 2016.

[3] James Brotherston, Richard Bornat, and Cristiano Calcagno. Cyclic proofs of program termination in separation logic. *ACM SIGPLAN Notices*, 43(1):101–112, 2008.
[4] James Brotherston and Alex Simpson. Sequent calculi for induction and infinite descent. *Journal of Logic and Computation*, 21(6):1177–1216, 2010.
[5] Pierre Clairambault. Least and greatest fixpoints in game semantics. In *Proceedings of FOSSACS 2009*, 2009.
[6] Anupam Das and Damien Pous. Non-wellfounded proof theory for (Kleene+action)(algebras+lattices). In *Proceedings of CSL 2018*, 2018.
[7] Simon Docherty and Reuben Rowe. A non-wellfounded, labelled proof system for Propositional Dynamic Logic. In *Proceedings of TABLEAUX 2019*, 2019.
[8] Michael Fisher and Roy Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 1979.
[9] Jérôme Fortier and Luigi Santocanale. Cuts for circular proofs: semantics and cut-elimination. In *Proceedings of CSL 2013*, 2013.
[10] Grigori E. Mints. Finite investigations of transfinite derivations. *Journal of Soviet Mathematics*, 10(4):548–596, 1978.
[11] Yury Savateev and Daniyar Shamkanov. Cut-elimination for the weak modal Grzegorczyk logic via non-wellfounded proofs. In *Proceedings of WoLLIC 2019*, 2019.

Towards Circular Proof Nets

Abhishek De, Luc Pellissier and Alexis Saurin
IRIF – CNRS, Université de Paris and INRIA
ade@irif.fr
luc.pellissier@irif.fr
alexis.saurin@irif.fr

Abstract

We develop infinets, that are proof-nets for non-wellfounded linear proof theory (aka. μMALL$^\infty$), and discuss some of their properties and shortcomings.

Finitary vs. non-wellfounded proof theory for fixed point logics. Extensions and variants of the μ-calculus [11] are now recognized as a central theoretical tools for the study of inductive and coinductive statements and of corresponding computational behaviours. Among them, multiplicative, additive linear logic with fixed points, μMALL has been developed in the past ten years and provided first with a finitary sequent proof system [1] with induction and coinduction rules à la Park and more recently with infinitary proofs where proof objects are non-wellfounded derivation trees [3, 9, 10, 12]. This last study stands in a broader line of work, not restricted to linear logic, to develop non-wellfounded proof theory [7, 6, 4, 5, 13], with a particular interest for the fragment of regular derivation trees called circular (or cyclic) proofs.

While finitary μMALL proofs are amenable to standard proof theoretic study, they suffer from several drawbacks due to the coinduction inference rule: (i) cut cannot be eliminated and (ii) they do not have a subformula property. Shifting to the setting of non-wellfounded proofs – where (co)inductive rules are replaced with simple fixed point unfoldings and branches of the proof trees may be infinitely deep – allows to recover full (infinitary) cut-elimination together with a subformula property (wrt. Fisher-Ladner subformulas).

On the parallel nature of (validating) threads. Despite their good properties, non-wellfounded proofs are complex objects and in particular their soundness relies on a global validity condition expressed in the form of a thread condition ensuring that some coinductive property is infinitely unfolded along any infinite branch. This validity condition is not only useful for ensuring soundness of the logic but also to ensure productivity of cut-elimination. Technically, a thread is a sequence of formulas in an infinite branch which realize an infinite path in the Fisher-Ladner graph of a formula with some additional conditions. Therefore a thread essentially abstracts away the notion of sequents. This as a discrepancy between the sequential nature of sequent proofs and the parallel nature of threads is one of the key to the difficulty in proving cut-elimination in such non-wellfounded settings.

It is therefore natural to investigate whether one can free the proof objects from some of the above-mentioned sequentiality or, in linear logic terms, to design non-wellfounded proof-nets. The aim of the present abstract is to introduce proof nets for μMALL in the non-wellfounded setting.

Infinets. The first and third authors recently introduced infinitary proof nets, or infinets, in the multiplicative fragment [8]: an example of (a graphical representation of) a non-wellfounded proof structure is given in Figure 1. Those proof nets are defined according to Curien's style for proof nets which allows a very smooth extension to non-wellfounded branches. Infinets can be viewed as the result of forgetting the order of inference rules in a proof tree and retaining only the subformula ordering and invariants on axioms and infinite branches.

A correctness criterion characterizes sequentializability (ie those infinite proof structure which come from pre-proofs): it extends the usual Danos-Regnier criterion for MLL (connectivity and acyclicity of correction graphs) with two additional conditions to cope with specific phenomena due to non-wellfoundedness. For instance, the proof structure of Figure 1 satisfies the Danos-Regnier criterion but is not a infinet as it fails the other part of the criterion. Indeed, this proof structure cannot be sequentialized: it does not correspond to any non-wellfounded proof. As usual with proof nets, the correctness criterion ensures canonicity i.e. if two proofs are equivalent upto inference permutations then they have isomorphic infinets.

After presenting infinets, we plan to discuss some shortcomings of [8] which are the topics of our present work and present our current work for solving them:

- the original paper only treats proofs with finitely many cuts. While bouncing validity [2] should be naturally captured with infinets (and actually initially motivated the study of infinets), [8] cannot express it yet;
- the notion of finitely representable (circular, or cyclic) infinets is non-trivial and does not agree with that of circular proofs: some non-circular proofs have a simple circular infinet while some circular proofs, have no simple finite representation.

[1] David Baelde. Least and greatest fixed points in linear logic. *ACM Transactions on Computational Logic (TOCL)*, 13(1):2, 2012.

[2] David Baelde, Amina Doumane, Denis Kuperberg, and Alexis Saurin. Bouncing threads for infinitary and circular proofs. Submitted, 2019.

[3] David Baelde, Amina Doumane, and Alexis Saurin. Infinitary proof theory: the multiplicative additive case. In *25th EACSL Annual Conference on Computer Science Logic, CSL 2016*, volume 62 of *LIPIcs*, pages 42:1–42:17. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016.

[4] Stefano Berardi and Makoto Tatsuta. Classical system of martin-löf's inductive definitions is not equivalent to cyclic proof system. In FOSSACS 2017, Proceedings, volume 10203 of *Lecture Notes in Computer Science*, pages 301–317, 2017.

[5] Stefano Berardi and Makoto Tatsuta. Equivalence of inductive definitions and cyclic proofs under arithmetic. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017*, pages 1–12. IEEE Computer Society, 2017.

[6] James Brotherston and Alex Simpson. Sequent calculi for induction and infinite descent. *Journal of Logic and Computation*, 21(6):1177–1216, December 2011.

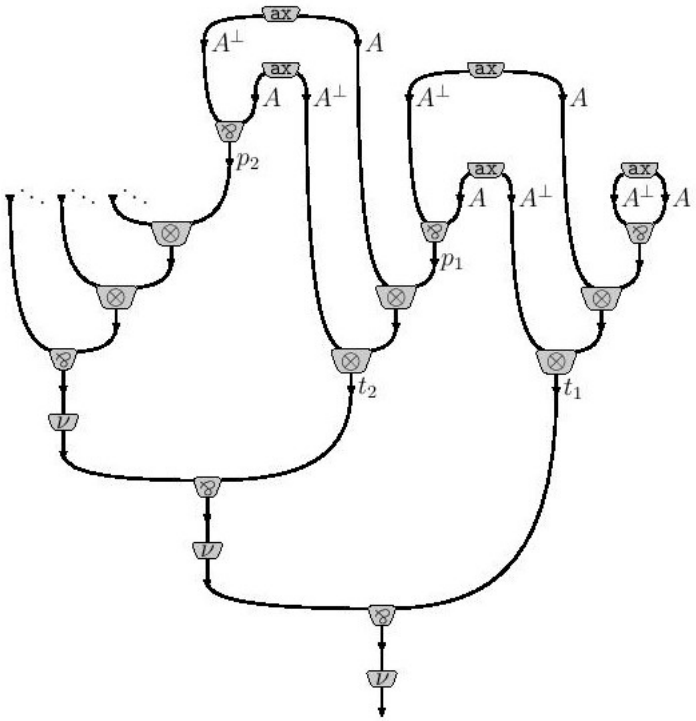


Figure 1: Example of a non-wellfounded proof structure of conclusion $\nu X.X \parr (A^\perp \otimes (A \otimes (A^\perp \parr A)))$.

[7] Christian Dax, Martin Hofmann, and Martin Lange. A proof system for the linear time μ-calculus. In *FSTTCS 2006, Proceedings*, pages 273–284, 2006.

[8] Abhishek De and Alexis Saurin. Infinets: The parallel syntax for non-wellfounded proof-theory. In *Automated Reasoning with Analytic Tableaux and Related Methods - 28th International Conference, TABLEAUX 2019, London, UK, September 3-5, 2019, Proceedings*, volume 11714 of *Lecture Notes in Computer Science*, pages 297–316. Springer, 2019.

[9] Amina Doumane. *On the infinitary proof theory of logics with fixed points. (Théorie de la démonstration infinitaire pour les logiques à points fixes).* PhD thesis, Paris Diderot University, France, 2017.

[10] Jérôme Fortier and Luigi Santocanale. Cuts for circular proofs: semantics and cut-elimination. In Simona Ronchi Della Rocca, editor, *Computer Science Logic 2013 (CSL 2013), CSL 2013*, volume 23 of *LIPIcs*, pages 248–262. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.

[11] Dexter Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[12] Luigi Santocanale. A calculus of circular proofs and its categorical semantics. In Mogens Nielsen and Uffe Engberg, editors, *Foundations of Software Science and Computation Structures*, volume 2303 of *Lecture Notes in Computer Science*, pages 357–371. Springer, 2002.

[13] Alex Simpson. Cyclic arithmetic is equivalent to peano arithmetic. In *FOSSACS 2017, Proceedings*, volume 10203 of *Lecture Notes in Computer Science*, pages 283–300, 2017.

Bouncing Threads for Infinitary and Circular Proofs

Amina Doumane
CNRS- ENS Lyon
amina.doumane@ens-lyon.fr

Infinitary (non-wellfounded) and circular proof systems have received much attention in recent years. Such proof systems allow non-wellfounded proof trees and impose some global validity condition in order to ensure soundness. Typically, it requires that every infinite branch is supported by some *thread* which traces some formula in a bottom-up manner and witnesses infinitely many progress points of a coinductive property.

Unfortunately, existing notions of validity impose a quite limited use of cuts in non-wellfounded proofs and many proofs that could be accepted as valid are rejected.

I will introduce a new validity condition for $\mu MALL^{\omega}$, the infinitary proof system for multiplicative additive linear logic with fixed points. This criterion generalizes the existing one, taking inspiration from Geometry of Interaction, enriching the structure of threads and relaxing their geometry: bouncing threads can leave the branch they validate and "bounce" (*i.e.* change direction, moving upward but also downward along proof branches) on axioms and cut rules.

As the usual "straight" criterion, the bouncing criterion enjoys cut elimination. This new validity condition is undecidable but can be decomposed into an infinite hierarchy of decidable conditions.

This is a joint work with David Baelde, Denis Kuperberg and Alexis Saurin.

A Temporal Logic for Concurrent Coalitional Strategies in Multi-player Games

Sebastian Enqvist and Valentin Goranko
Department of Philosophy, Stockholm University
{sebastian.enqvist, valentin.goranko}@philosophy.su.se

Abstract

We introduce a new simple yet expressive temporal logic for reasoning about concurrent game structures. We provide motivating examples, and discuss ongoing work on completeness, decidability and finite model property. The main underlying technical result is a translation into a variant of the modal μ-calculus.

Introduction and background

Game theory is one of the scientific disciplines where semantic and syntactic circularity play pivotal roles in several fundamental notions. For instance the key concept of *equilibrium*, being a best response to the other players' best responses, is a self-referential notion. *Infinite games* have a co-inductive flavor, and have gained attention in computer science as natural models of possibly non-terminating computations.

We consider multi-player concurrent games, in which possibly infinite sequences ('computations', 'plays') of transitions between configurations are carried out by several interacting agents acting on a common arena ('multi-player game structure'). Over the past few decades a variety of logics have been developed for formal specification and analysis of such games, extending familiar modal and temporal logics to enable modelling of the players' interaction. An early example is Parikh's dynamic logic of games [6] which extends propositional dynamic logic to a two-player setting, also adding a role-swapping program constructor. Later, Coalition Logic CL was introduced by Pauly [7] as a multi-agent multi-modal logic formalising reasoning about strategic abilities of coalitions of agents to guarantee the achievement of designated objectives regardless of the actions of the remaining agents. More precisely, CL features strategic operators of the type $[C]$, for any group ('coalition') of agents C and, for any formula ϕ, regarded as expressing the coalitional objective of C, $[C]\phi$ intuitively says that the coalition C has a collective action σ_C that guarantees the satisfaction of ϕ in every outcome state that can occur when the agents in C execute their actions in σ_C, regardless of the choice of actions of the agents that are not in C. Thus, CL can reason about unconditional powers of agents and coalitions to act unilaterally in pursuit of their goals.

Independently, and at about the same time, temporal extensions of CL, called "alternating-time temporal logics" ATL, ATL*, as well as "alternating-time μ-calculus", were introduced by Alur et al in [2] as multi-agent extensions of the standard computation-tree logics CTL and CTL* and the modal μ-calculus.

Even more expressive formalisms have been proposed since then, like "strategy logics" [5] or ATL/ ATL* extended with"strategy contexts" [1, 3]. All these logics involve modal and temporal operators with semantics based on fixed point constructions and other circular concepts mentioned above.

The logics CL and ATL/ATL* provide a natural, but rather restricted perspective: the agents in the proponent coalition are viewed as acting in full cooperation with each other but in complete opposition to all agents outside of the coalition, which are treated as adversaries. The strategic interaction in real life is much more complex, usually involving various patterns combining cooperation and competition. To capture these, more expressive and refined logical frameworks are needed. The recent work [4] took that issue up and proposed two expressive and versatile logical systems extending CL:

- the *Socially Friendly Coalition Logic (*SFCL*)*, enabling formal reasoning about strategic abilities of individuals and groups to ensure achievement of their private goals while allowing for cooperation with the entire society;
- the complementary, *Group Protecting Coalition Logic (*GPCL*)*, capturing reasoning about strategic abilities of the entire society to cooperate in order to ensure achievement of the societal goals, while simultaneously protecting the abilities of individuals and groups within the society to achieve their individual and group goals.

Main ideas and technical results

This abstract reports on an ongoing work, aiming at extending further the logic GPCL with temporal operators, thereby also extending ATL with the more sophisticated patterns of strategic reasoning provided by GPCL. The result is a new multi-agent temporal logic that retains much of the computational simplicity of both GPCL and ATL, yet it combines their features to enable reasoning about quite complex strategic interaction in concurrent game structures.

In particular, the logic that we propose is suited for reasoning about what we call *concurrent coalitional strategies*. While ATL can express statements about the power of groups of individual agents to force certain outcomes (coalitional powers), it lacks the capability to reason about how different coalitions can form simultaneously to independently achieve their objectives. This type of concurrency, we believe, is an important feature of multi-player strategic interaction. The standard solution concept of a Nash equilibrium can be seen as an instance of this notion, where several different coalitions are acting simultaneously to keep each individual player "in check" so they cannot deviate to obtain a higher payoff, which is required to ensure that the strategy profile is an equilibrium.

In this talk we will provide motivating examples, will introduce the logic ConStrL and will report on some recent technical results on it. Our main technical result is a (non-trivial) translation of ConStrL into a multi-agent variant of the modal μ-calculus. We show that this translation can be carried out using only a single recursion variable. This implies that methods for proving completeness of single-variable or *flat* μ-calculi [8, 9] can be adapted to ConStrL, using the completeness result for its next-time fragment GPCL obtained in [4]. In ongoing work in progress, we are using these methods to obtain a sound and complete axiomatic proof system for ConStrL, and show decidability and the finite model property for that logic.

[1] T. Ågotnes, V. Goranko, and W. Jamroga. Strategic commitment and release in logics for multi-agent systems (extended abstract). Technical Report IfI-08-01, Clausthal University of Technology, 2008.
[2] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, 2002.
[3] Thomas Brihaye, Arnaud Da Costa Lopes, François Laroussinie, and Nicolas Markey. ATL with strategy contexts and bounded memory. In S. Artëmov and A. Nerode, editors, *Proc. of LFCS'2009*, volume 5407 of *LNCS*, pages 92–106. Springer, 2009.
[4] Valentin Goranko and Sebastian Enqvist. Socially friendly and group protecting coalition logics. In *Proc. of AAMAS 2018*, pages 372–380, 2018.
[5] Fabio Mogavero, Aniello Murano, Giuseppe Perelli, and Moshe Y. Vardi. Reasoning about strategies: on the satisfiability problem. *Logical Methods in Computer Science*, 13(1), 2017.
[6] Rohit Parikh. The logic of games and its applications. *North-Holland Mathematics Studies*, 102:111–139, 1985.
[7] M. Pauly. A modal logic for coalitional power in games. *Journal of Logic and Computation*, 12(1):149–166, 2002.
[8] Luigi Santocanale and Yde Venema. Completeness for flat modal fixpoint logics. *Ann. Pure Appl. Logic*, 162(1):55–82, 2010.
[9] Lutz Schröder and Yde Venema. Flat coalgebraic fixed point logics. In Paul Gastin and François Laroussinie, editors, *CONCUR*, volume 6269 of *Lecture Notes in Computer Science*, pages 524–538. Springer, 2010.

Interpolation for PDL: An Open Problem?

Malvin Gattinger[a] and Yde Venema[b]

[a]Department of Artificial Intelligence; University of Groningen, The Netherlands
malvin@w4eg.de

[b]Institute for Logic, Language and Computation; Universiteit van Amsterdam, The Netherlands
y.venema@uva.nl

Propositional dynamic logic or PDL [3], introduced by Vaughan Pratt in the 1970s [7] is a relatively simple modal logic for reasoning about programs. Characteristic about the formalism is its two-sorted syntax, where the modalities of the language are given by an inductively defined collection of programs. PDL can be classified as a fixpoint logic because of the crucial iteration constructor on programs, with α^* denoting the (nondeterministic) program consisting of executing α an arbitrary but finite number of times.

Propositional dynamic logic has a reasonably well-developed meta-logical theory; for instance, it displays good computational behaviour, and has an elegant axiomatisation. However, the question whether the logic enjoys the (Craig-style) interpolation property, does not seem to have a simple answer. In the literature, at least three proofs have been proposed for the statement that PDL has interpolation indeed [6, 1, 4], but then all of these have also been reported to be wrong. In this talk we will first briefly review this history of (alleged) claims and refutations.

We will then focus on a paper by Daniel Leivant [6], which presented the first interpolation proof of PDL, unfortunately in a rather sketchy form. We will defend the proof against the criticism expressed by Marcus Kracht [5]. We will then present an overview of Leivant's proof; giving our own review, we will single out the parts that we have been able to verify [2] and those that need further scrutiny.

[1] Manfred Borzechowski. Tableau–Kalkül für PDL und Interpolation. Diplomarbeit, FU Berlin, 1988.

[2] Malvin Gattinger. Craig interpolation of PDL – a report on the proof by Daniel Leivant (1981). Technical report, 2014.

[3] David Harel, Dexter Kozen, and Jerzy Tiuryn. *Dynamic Logic*. The MIT Press, 2000.

[4] Tomasz Kowalski. PDL Has Interpolation. *J. Symb. Log.*, 67(3):933–946, 2002.

[5] Marcus Kracht. *Tools and Techniques in Modal Logic*. Number 142 in Studies in Logic. Elsevier, Amsterdam, 1999.

[6] Daniel Leivant. A Proof Theoretic Methodology for Propositional Dynamic Logic. In J. Díaz and I. Ramos, editors, *ICFPC*, volume 107 of *LNCS*, pages 356–373. Springer, 1981.

[7] Vaughan Pratt. Semantical considerations on Floyd-Hoare logic. In *Proc. 17th IEEE Symposium on Computer Science*, pages 109–121, 1976.

Equations as a Tool for Studying Logic Fragments

Mai Gehrke
Laboratoire J. A. Dieudonné, CNRS & Université Côte d"Azur
Mai.Gehrke@unice.fr

Topological methods, and Stone duality in particular, have played an important rôle in many parts of semantics, but applications in more algorithmic subjects are not so common. One of the few such applications is the use of profinite methods in automata theory, in particular so-called profinite equations for settling decision problems. The purpose of this talk is to explain what these equations are from a duality theoretic point of view, which may be applied to logic fragments in greater generality.

This project has been supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No.670624).

Complete Proof Systems for Parikh's Game Logic

Helle Hvid Hansen
Delft University of Technology
h.h.hansen@tudelft.nl

Game logic was introduced by Parikh in 1985 [8, 7] as a modal logic for reasoning about the outcomes that players can force in determined 2-player games such as those arising in mathematical economics and finite model theory.

Referring to the two players as *Angel* and *Demon*, a modal formula $\langle\gamma\rangle\phi$ should be read as, *"Angel has a strategy in the game γ to ensure an outcome in which ϕ holds"*. Syntactically, Parikh's game logic is an extension of Fischer & Ladner's propositional dynamic logic (PDL) with games now taking the place of programs. Complex games are composed from atomic games and constructors that denote sequential composition of games, as well as choice, iteration and test for Angel, and finally the dual operator which denotes swapping the roles of the two players. The strategic ability of Demon is thus only implicitly expressed through the dual operator. Semantically, the generalisation from 1-player games to 2-player games is obtained by moving from Kripke structures to monotone neighbourhood structures. Game logic is thus a non-normal, monotone modal logic.

Just as PDL can be translated into the (normal) modal μ-calculus [3], game logic can be translated into the monotone modal μ-calculus [7], and from there into normal modal μ-calculus [6]. However, unlike PDL and other fragments such as LTL and CTL*, game logic spans all levels of the alternation hierarchy of the normal modal μ-calculus [2]. Game logic is thus a highly expressive fragment of μ-calculus. Parikh proposed in his original paper a natural PDL-style Hilbert system which was easily proved to be sound, but its completeness remained an open problem.

In this talk, I will present recent work [4] in which we prove the completeness of Parikh's axiomatisation via a sequence of proof transformations that connects Parikh's Hilbert system with a complete proof system for the normal modal μ-calculus from [1]. This sequence of transformation goes via three intermediate proof systems, each of which is complete and of interest in its own right. The first, called G, is a cut-free sequent calculus for game logic with deep inference rules. The second, called CloG, is a circular, analytic sequent calculus for game logic in which formulas are annotated in order to keep track of fixpoint unfoldings. The third, called CloM, is a similar system for the monotone μ-calculus. The transformation of CloM-derivations into CloG-derivations relies on a novel translation from game logic into the monotone μ-calculus. This translation is truth- and validity-preserving, it commutes with fixpoint unfolding, and crucially, it reflects the order on fixpoint variables. Our approach builds on ideas and results by Afshari & Leigh [1] who developed cut-free sequent calculi for the normal μ-calculus.

At the end of the talk, if time permits, I will briefly discuss a coalgebraic generalisation of game logic [5].

[1] B. Afshari and G. Leigh. Cut-free completeness for modal mu-calculus. In *LICS 2017*, pages 1–12, 2017.

[2] D. Berwanger. Game Logic is strong enough for parity games. *Studia Logica*, 75(2):205–219, 2003.
[3] F. Carreiro and Y. Venema. PDL inside the μ-calculus: a syntactic and an automata-theoretic characterization. In R. Goré et alii, editor, *Advances in Modal Logic*, volume 10, pages 74–93, 2014.
[4] S. Enqvist, H. H. Hansen, C. Kupke, J. Marti, and Y. Venema. Completeness for game logic. In Patricia Bouyer, editor, *Thirty-Fourth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2019), Vancouver, Canada, 24-27 June, 2019*. IEEE, 2019.
[5] H. H. Hansen and C. Kupke. Weak completeness of coalgebraic dynamic logics. In *Fixed Points in Computer Science (FICS)*, volume 191 of *EPTCS*, pages 90–104, 2015.
[6] M. Kracht and F. Wolter. Normal monomodal logics can simulate all others. *Journal of Symbolic Logic*, 64(1):99–138, 1999.
[7] M. Pauly. *Logic for Social Software*. PhD thesis, University of Amsterdam, 2001.
[8] M. Pauly and R. Parikh. Game Logic: An overview. *Studia Logica*, 75(2):165–182, 2003.

On Friedman-Sheard Theories for Recursive Realizability

Daichi Hayashi

Hokkaido University, Sapporo, Japan
daichinhayashi0611@gmail.com

Introduction

According to the liar paradox, we cannot simultaneously retain many plausible principles for the unary truth predicate $\mathsf{T}(x)$. One famous solution by Friedman and Sheard ([1]) is to list several axioms or rules for a truth predicate, and then determine their consistent combinations. As a result, they specified nine theories A to I consisting of maximally consistent sets of truth principles. While their investigation concerns classical logic, we can also naturally think of the notion of truth in non-classical logic; as such, a comprehensive study by Leigh and Rathjen ([5]) gives nine intuitionistic counterparts A^i to I^i, the proof-theoretic strengths of which are later determined by Leigh ([4]).

Leigh and Rathjen's list of truth principles contain the axioms for logical connectives based on the normal Tarskian model theory, e.g. $\forall^\ulcorner A \vee B^\urcorner(\mathsf{T}^\ulcorner A \vee B^\urcorner \to \mathsf{T}^\ulcorner A^\urcorner \vee \mathsf{T}^\ulcorner B^\urcorner)$ informally stating that if (the name of) a sentence of the form $A \vee B$ is true then (the name of) either A or B is true. However, it is also possible to adopt evaluation clauses more suited for intuitionistic logic. In this paper, we reformulate Leigh and Rathjen's intuitionistic Friedman-Sheard theories using Kleene's (*recursive*) *realizability* interpretation ([3]).

In Kleene's realizability, which is inspired by the Brouwer-Heyting-Kolmogorov (BHK) interpretation, each sentence is asserted in association with specific information (*realization number*) about its particular construction (proof). For example, Kleene's realizability condition has the following clauses (see [3, 8]):

1. A number n is a realization for the sentence $A \vee B$ if and only if n is a pair $(0, (n)_1)$ and $(n)_1$ is a realization number for A, or n is a pair $((n)_0, (n)_1)$ with $(n)_0 \neq 0$ and $(n)_1$ is a realization number for B, where $(n)_0$ and $(n)_1$ are projection functions.
2. A number n is a realization of the sentence $A \to B$ if and only if n is a code of an effective procedure that converts each given realization number of A into that of B.

Based on this semantics, Nelson ([7]) provides an interpretation of Heyting Arithmetic (HA) that can be seen as corresponding to the fact that Peano Arithmetic (PA) is true in the classical standard model ([8]). Therefore, the use of realizability clauses seems natural in formalising intuitionistic truth. In particular, we focus on the realizability versions rD^i and rH^i of D^i and H^i, respectively, and obtain their proof-theoretic equivalences.

The author thanks the Japan Society for the Promotion of Science (JSPS), Core-to-Core Program (A. Advanced Research Networks) for supporting the research.

Results

Let $\mathcal{L}_{\mathrm{R}}$ be a language of HA, augmented with a new binary predicate symbol $x\mathrm{R}y$, informally meaning "x realizes y". We define the realizability predicate $\mathrm{R}(y)$ as $\exists x(x\mathrm{R}y)$. For the definition of an $\mathcal{L}_{\mathrm{R}}$-theory rD^i, we follow Halbach's reformulation of D in [2].

Definition 1. rD^i consists of HA and all axioms for realizability clauses. Here, we give two examples, $(\mathrm{R}_\vee)$ and $(\mathrm{R}_\to)$, corresponding to the above realizability clauses (for the notation see [5]):

$$(\mathrm{R}_\vee)\ \ \forall x\forall^\ulcorner A\vee B^\urcorner(x\mathrm{R}^\ulcorner A\vee B^\urcorner\leftrightarrow(((x)_0=0\wedge(x)_1\mathrm{R}^\ulcorner A^\urcorner)\vee((x)_0\neq 0\wedge(x)_1\mathrm{R}^\ulcorner B^\urcorner))),$$
$$(\mathrm{R}_\to)\ \ \forall x\forall^\ulcorner A\to B^\urcorner(x\mathrm{R}^\ulcorner A\to B^\urcorner\leftrightarrow(\forall y(y\mathrm{R}^\ulcorner A^\urcorner\to\{x\}(y)\downarrow\wedge\{x\}(y)\mathrm{R}^\ulcorner B^\urcorner))),$$

where $\{\}$ is the Kleene bracket and $\{x\}(y)\downarrow$ means that x has a value at y. The axioms for the other connectives are similarly defined according to Kleene's clauses.

Moreover, rD^i has the following two rules (R-Intro) and (R-Elim):

$$\frac{A(x)}{\mathrm{R}^\ulcorner A(\dot{x})^\urcorner}\ (\text{R-Intro}),\qquad \frac{\mathrm{R}^\ulcorner A(\dot{x}^\urcorner)}{A(x)}\ (\text{R-Elim}),$$

where $\ulcorner A(\dot{x})\urcorner$ is the result of substituting x for the free variable v in $\ulcorner A(v)\urcorner$.

Using a realizability interpretation, we can translate a subtheory of D^i ([4]) into rD^i. For the upper bound, we present a realizability version of Friedman-Sheard style revision semantics ([1]); in this model we have the soundness of rD^i. Similar to Leigh and Rathjen's theories ([6]), this proof can be formulated in ACA_0^{i+} (for a definition see [4]). Therefore, we obtain the following result:

Theorem 2. rD^i *and* D^i *have the same, negative arithmetical theorems.*

Definition 3. The theory rH^i is defined according to H^i in [5]. Here, we give two axioms as examples:

(R-Out) $\mathrm{R}^\ulcorner A(\dot{x})^\urcorner\to A(x)$ for each $\mathcal{L}_{\mathrm{R}}$-formula A;
(R-Rep) $\forall x\forall y\forall z(x\mathrm{R}y\to z\mathrm{R}^\ulcorner\dot{x}\mathrm{R}\dot{y}^\urcorner)$.

As for the proof-theoretic strength, the systems rH^i and H^i can mutually be translated via a realizability interpretation. Therefore, we have the following:

Theorem 4. rH^i *and* H^i *have the same, almost negative arithmetical theorems.*

Future Work

In the future, we expect to obtain realizability counterparts for intuitionistic truth theories other than D^i and H^i. In particular, Leigh and Rathjen ([5]) interpret truth predicates for E^i; F^i; G^i and I^i as intuitionistic provability. Thus, associating realization numbers with these proofs, we would have the upper bounds of rE^i, rF^i, rG^i and rI^i.

From the axiom $(\mathrm{R}_\to)$ of rD^i we have the weak completeness axiom $\forall^\ulcorner A^\urcorner(\neg\mathrm{R}^\ulcorner A^\urcorner\to\mathrm{R}^\ulcorner\neg A^\urcorner)$. However, as rD^i implies the realizability of a classically

refutable statement (cf. [8]), we cannot consistently add the strong completeness axiom (R-Comp): $\forall \ulcorner A \urcorner (\mathrm{R} \ulcorner A \urcorner \vee \mathrm{R} \ulcorner \neg A \urcorner)$ in the presence of (the uniform version of) the rule (R-Elim). This fact implies the existence of maximally consistent subtheories of $\mathsf{rD}^i + (\text{R-Comp})$ other than rD^i.

[1] Harvey Friedman and Michael Sheard. An axiomatic approach to self-referential truth. *Annals of Pure and Applied Logic*, 33:1–21, 1987.

[2] Volker Halbach. A system of complete and consistent truth. *Notre Dame Journal of Formal Logic*, 35(3):311–327, 1994.

[3] Stephen Cole Kleene. On the interpretation of intuitionistic number theory. *The journal of symbolic logic*, 10(4):109–124, 1945.

[4] Graham E Leigh. A proof-theoretic account of classical principles of truth. *Annals of Pure and Applied Logic*, 164(10):1009–1024, 2013.

[5] Graham E Leigh and Michael Rathjen. The friedman-sheard programme in intuitionistic logic. *The Journal of Symbolic Logic*, 77(3):777–806, 2012.

[6] Graham Emil Leigh and Michael Rathjen. An ordinal analysis for theories of self-referential truth. *Archive for Mathematical Logic*, 49(2):213–247, 2010.

[7] David Nelson. Recursive functions and intuitionistic number theory. *Transactions of the American Mathematical Society*, 61(2):307–368, 1947.

[8] A. Troelstra and D. van Dalen. *Constructivism in mathematics: An Introduction*. Elsevier Science, 1988.

The Reflective Multiverse of Set Theory

Paul Kindvall Gorbow and Graham E. Leigh
Department of Philosophy, Linguistics and Theory of Science, University of Gothenburg
paul.gorbow@gu.se
graham.leigh@gu.se

What is the significance of having proofs from the axioms of ZF? Three natural responses run as follows:

1. (Universe) The conclusions of proofs from ZF are true about sets.
2. (Multiverse) The conclusions of proofs from ZF are true in every universe satisfying ZF.
3. (Pragmatic formal foundations) The purpose of ZF is to facilitate implementation of "almost all of mathematics". Thus, proofs in ZF are only significant when they arise from the implementation of some mathematics.

The first response embodies a strong realist perspective of the notion of set, namely that there is a single, absolute, realm of sets whose facts-of-the-matter ground the truth or falsity of every set theoretic proposition. In contrast, the multiverse view holds that there exist a realm of universes, and each 'universe' is equipped with it's own notion of set and, possibly, with it's own conception of the multiverse.

A pragmatic reason for adopting the multiverse view is that set theorists actually work with conflicting axiomatic extensions of ZF, and that no methodology has been successful in deciding between these conflicting axioms. For instance, the methodology of large cardinals, which was initially considered to support the 'single universe' realist, does not decide important set-theoretic principles such as the continuum hypothesis. The multiverse view offers an explanation for this state of affairs.

The pragmatic formal foundations view restricts the scope of the formalist view on mathematics to just the foundation of mathematics. The idea is that the sole purpose of a foundational system, say ZF, is to embody a single unified framework for interpretation of "almost all of mathematics" by means of structures constructible in ZF. Thus, there is no need for the axioms of ZF to be true in any sense, and proofs in ZF only have indirect significance. This view is consistent with, and provides a foundation for, realism about the mathematics that gets interpreted, for example: The standard model of arithmetic can be constructed in ZF, and for this model ZF supplies a definition of truth, whereby the conclusions of proofs in PA come out as true, in line with a universe view of arithmetic.

This talk will argue that these three distinct positions relating proof and truth can coexist harmoniously. We present a novel mathematical construction, the Reflective Multiverse, which gives a formal account of the multiverse perspective, equipped with an untyped notion of truth-relative-to-a-universe allowing a universe to refer to both itself and other universes of the multiverse. The Reflective Multiverse draws on multiverse ideas of Gitman and Hamkins [2] as well as revision-semantical ideas from the axiomatic truth-theory of Friedman and Sheard [1].

The construction of the Reflective Multiverse takes place in the untyped compositional theory of truth obtained by iterating a reflection principle ω times over ZF. Here are some notable ingredients:

1. An untyped relation of truth-relative-to-a-universe, informally "$\mathcal{U}$ thinks that ϕ", obtained through a revision-semantical process that generalises the familiar set theoretic definition of $\mathcal{U} \models \phi$.
2. Typically, a universe will contain its own multiverse. So nested sentences such as '$\mathcal{U}$ thinks that "$\mathcal{V}$ thinks that ϕ" ' are expressible, where $\mathcal{V}$ is a universe in the multiverse of $\mathcal{U}$.
3. In a strong form of the Reflective Multiverse, constructed by iterating the Global Reflection principle, each universe has an isomorphic copy of itself in its multiverse.

There is an interplay between two types of circularity here: Firstly the untyped truth-relative-to-a-universe predicate can be applied to itself. Secondly, the Global Reflection principle employed results in the isomorphism property above, which gives rise to universes whose own reflection of the multiverse contains a copy of themselves. It turns out that this makes it possible to interpret a strong universe view within each universe of the multiverse.

In summary, the foundational background for the construction, the system obtained by iterating a reflection principle ω times over ZF, interprets a natural theory of the multiverse, which expresses a view of a rich multiverse. Moreover, in the strong form of the construction, the universe view is harmoniously positioned within this multiverse view. But what is the significance of proofs in this multiverse theory? We take it that this can neither be answered with the universe nor the multiverse view, and that it makes more sense to look at it as a pragmatic formal foundation.

[1] Harvey Friedman, Michael Sheard, An axiomatic approach to self-referential truth, *Annals of Pure and Applied Logic*, vol. 33 (1987), pp. 1–21.
[2] Victoria Gitman, Joel Hamkins, A natural model of the multiverse axioms, *Notre Dame Journal of Formal Logic*, vol. 51 (2010), no. 4, pp. 475–484.

Size Matters in the Modal μ-calculus

Clemens Kupke[a], Johannes Marti[b] and Yde Venema[b]
[a]Strathclyde University
clemens.kupke@strath.ac.uk
[b]ILLC, University of Amsterdam
johannes.marti@gmail.com
Y.Venema@uva.nl

Overview

Among the fixpoint logics used in computer science the modal μ-calculus offers high expressive power, while still having comparatively low computational complexity for common decision problems. When analyzing the complexity algorithms for the modal μ-calculus one can choose between at least three different measures for the size of a formula: First, its length, that is, the number of symbols in a representation of the formula as a string; second, its subformula size, that is, the number of distinct subformulas of the formula; and third, its closure size, that is, the number of distinct elements in its Fischer-Ladner closure.

We characterize classes of alternating tree automata that correspond to these different measures for the size of a formula in the μ-calculus, thus providing a framework in which algorithmic properties of formulas can be conveniently discussed. We call these alternating tree automata *parity formulas* as they are a direct generalization of the presentation of formulas as trees whose nodes are labelled with logical connectives.

We use this framework to discuss the existence of efficient guarded transformation, which is roughly the problem of rewriting a formula of the modal μ-calculus into an equivalent one in which every occurrence of a bound variable is separated from its binder by a modal operator.

Moreover, we discuss how the size of a formula behaves with respect to alphabetical equivalence, that is, the renaming of bound variables. Especially, we show how to efficiently obtain for every formula a minimal alphabetically equivalent formula, where we consider both minimality with respect to subformula size and minimality with respect to closure size.

Size matters

Our starting point are two, surprising, observations by Bruse, Friedmann and Lange from [1]:

1. Closure size can be exponentially more succinct than subformulas size. More precisely, there exist enumerations of formulas for which the number of distinct subformulas increases exponentially, whereas the size of the closure increases just linearly. Thus, there is a good reason to develop algorithms for the μ-calculus that are efficient in the closure size of the input.
2. The standard unfolding procedure to obtain a guarded equivalent to a given formula, which is widely believed to be polynomial, can cause an exponential blow-up. This is a problem because guarding is often used as an initial step in complexity results, for instance when constructing an equivalent automaton

for a given formula or when employing a tableaux system that only operates on guarded formulas.

Parity formulas

Our main contribution is to characterize the different size measures for formulas in the μ-calculus by means of structural constraints on parity formulas, which are defined as follows:

Definition 1. A *parity formula* $\mathcal{G} = (V, E, L, \Omega, v_I)$ is a tuple where

- (V, E) is a finite, directed graph;
- $L : V \to \mathsf{Lit} \cup \{\wedge, \vee, \Diamond, \Box, \epsilon\}$ is a labeling function assigning literals and logical operators to nodes;
- $\Omega : V \xrightarrow{\circ} \omega$ is a partial function assigning numbers, thought of as priorities, to some of the nodes in $\mathcal{G}$; and
- $v_I \in V$ is the initial node of $\mathcal{G}$;

such that

1. $|E[v]| = 0$ if $L(v) \in \mathtt{Lit}$, $|E[v]| = 1$ if $L(v) \in \{\Diamond, \Box, \epsilon\}$, and $|E[v]| \leq 2$ if $L(v) \in \{\wedge, \vee\}$;
2. every cycle of (V, E) contains at least one node on which Ω is defined.

The size of a parity formula is the number of elements in V. Note that, because the outdegree of (V, E) is bound by 2, the number of elements in V also linearly bounds the actual length of an encoding of the whole parity formula.

Parity formulas are essentially the same as Wilke's alternating tree automata [3] and there are direct translations between parity formulas and the hierarchical equations systems from [2].

Analogously to formulas in the modal μ-calculus and to alternating tree automata it is straight-forward to define a semantics for parity formulas over Kripke structures in terms of parity games.

One can obtain an equivalent parity formula for every formula in the μ-calculus by considering its syntactic tree as a tree with backedges. Conversely, if a parity formula is a tree with backedges, where the priorities are suitably constrained, then we can read off an equivalent formula in the μ-calculus, whose length is linear in the number of nodes of the given parity formula. In this sense measuring the size of a formula by its length corresponds to working with parity formulas whose underlying graph is a tree with backedges.

We obtain similar results for subformula size, by defining a class of *untwisted* parity formulas, which roughly means that the underlying graph is a directed acyclic graph with backedges that are well-behaved with respect to the priorities. The subformula graph of formula in the μ-calculus can be turned into such a untwisted parity formula, and, conversely, for every untwisted parity formula there is an equivalent formula in the μ-calculus, whose subformula size is linear in the size of the given parity formula.

Lastly, we show that an analogous correspondence exists between the graph structure on the closure of a formula in the μ-calculus and the class of parity formulas without any structural constraints.

Guarded transformation

Concerning the guarded transformations we have two main results:

First, we provide a procedure that transforms untwisted parity formulas into guarded parity formula of size quadratic in the size of the input, which is not necessarily untwisted. In terms of formulas in the μ-calculus this means that efficient guarded transformation is possible if one measures the input in terms of subformula size, but accepts the closure graph of a formula as output. Our result is reminiscent of a similar result for a restricted class of hierarchical equations systems from [2].

Second, we prove a limitative result concerning the difficulty of finding an efficient guarded transformation procedure, when the input is measured in closure size. Exploiting the connection between parity formulas and parity games, we show that if there was a polynomial algorithm that takes as input an arbitrary parity formula and produces an equivalent guarded parity formula then parity games could be solved in polynomial time. Although the latter is generally not believed to be impossible, it is has remained an open question for a long time.

Alphabetical variants

We also discuss how the three size measures behave with respect to alphabetical equivalence. Obviously, the length of a formula does not change when renaming bound variables. The same holds for closure size, but only if we identify alphabetically equivalent formulas in the closure set. Nevertheless, we show how for a given formula one can obtain an alphabetical variant such that on its closure alphabetical equivalence implies syntactic identity. For subformula size there is the additional complication that alphabetically equivalent formulas do not necessarily have the same subformulas, even when considering subformulas up-to alphabetical equivalence. For this case we show how to obtain an alphabetical variant of a given formula such that the number of its subformulas, counted up-to syntactic identity, is minimal among all alphabetical variants of the given formula.

[1] F. Bruse, O. Friedmann, and M. Lange. On guarded transformation in the modal μ-calculus. *Logic Journal of the IGPL*, 23(2):194–216, 2015.

[2] Helmut Seidl and Andreas Neumann. On guarding nested fixpoints. In Jörg Flum and Mario Rodríguez-Artalejo, editors, *Computer Science Logic, 13th International Workshop, CSL '99, 8th Annual Conference of the EACSL, Madrid, Spain, September 20-25, 1999, Proceedings*, volume 1683 of *Lecture Notes in Computer Science*, pages 484–498. Springer, 1999.

[3] T. Wilke. Alternating tree automata, parity games, and modal μ-calculus. *Bulletin of the Belgian Mathematical Society*, 8:359–391, 2001.

A Turing-Complete Extension of First-Order Logic

Antti Kuusisto
University of Helsinki, Finland
antti.kuusisto@tuni.fi

Abstract

We discuss a natural Turing-complete extension of first-order logic FO. The extension adds two new features to FO. The first one of these is the capacity to *add new points* to models and *new tuples* to relations. The second one is a construct that allows formulae to refer to themselves. We survey some of the main features of this logic and its extensions.

Introduction

A natural Turing-complete extension $\mathcal{L}_0$ of first-order logic FO can be obtained by extending FO by two new features: (1) operators that *add new points* to models and *new tuples* to relations, and (2) operators that enable formulae to *refer to themselves*. The self-referentiality operator of $\mathcal{L}_0$ is based on a construct that enables *looping* when formulae are evaluated using game-theoretic semantics.

Here we present the logic $\mathcal{L}_0$ formally and discuss its relation to Turing-computation. Indeed, it turns out that $\mathcal{L}_0$ captures the class RE in the sense of descriptive complexity theory. We also discuss how $\mathcal{L}_0$ relates to natural language and observe that all operators of $\mathcal{L}_0$ have very simple natural language correspondents.

The reason the logic $\mathcal{L}_0$ is particularly interesting lies in its simplicity and its *exact behavioural correspondence* with Turing machines. Furthermore, it provides a natural and particularly simple *unified perspective* on logic and computation. It is also worth noting that the new operators of $\mathcal{L}_0$ nicely capture two fundamental classes of constructors that are omnipresent in the everyday practice of mathematics (but missing from FO): scenarios where fresh points are added to investigated constructions (or fresh lines are drawn, et cetera) play a central role in geometry, and recursive looping operators are found everywhere in mathematical practice, often indicated with the help of the famous three dots (...).

The logic $\mathcal{L}_0$ was first introduced in [5]. Below we discuss the results of that article and also further results not covered there. Other systems that bear some degree of similarity to $\mathcal{L}_0$ include for example BGS logic [1] and abstract state machines [3, 2]. Logics that modify structures include, for example, sabotage modal logic and public announcements logics. Recursive looping operators are a common feature in logics in finite model theory and verification. However, while the approach in $\mathcal{L}_0$ bears a degree of similarity to the fixed point operators of the μ-calculus, $\mathcal{L}_0$ is not based on fixed points and no monotonicity requirements apply.

The Turing-complete logic $\mathcal{L}_0$

Let $\mathcal{L}_0$ denote the language that extends the syntax of first-order logic by the following formula construction rules:

1. $\varphi \mapsto \mathrm{I}x\,\varphi$
2. $\varphi \mapsto \mathrm{I}_{R(x_1,\ldots,x_n)}\,\varphi$

3. C_i is an atomic formula (for each $i \in \mathbb{N}$)
4. $\varphi \mapsto C_i\,\varphi$

Intuitively, a formula of type $\mathrm{I}x\,\varphi(x)$ states that it is possible to *insert* a fresh, isolated element u into the domain of the current model so that the resulting new model satisfies $\varphi(u)$. The fresh element u being *isolated* means that u is disconnected from the original model; the relations of the original model are not altered in any way by the operator $\mathrm{I}x$, so u does not become part of any relational tuple at the moment of insertion. (Note that we assume a purely relational signature for the sake of simplicity.)

A formula of type $\mathrm{I}_{R(x_1,\ldots,x_n)}\,\varphi(x_1,\ldots,x_n)$ states that it is possible to insert a tuple $(u_1,\ldots,u_n)$ into the relation R so that $\varphi(u_1,\ldots,u_n)$ holds in the obtained model. The tuple $(u_1,\ldots,u_n)$ is a sequence of elements in the original model, so this time the domain of the model is not altered. Instead, the n-ary relation R obtains a new tuple.

The new atomic formulae C_i can be regarded as *variables* ranging over formulae, so a formula C_i can be considered to be a *pointer* to (or the *name* of) some other formula. The formulae $C_i\,\varphi$ could intuitively be given the following reading: *the claim C_i, which states that φ, holds.* Thus the formula $C_i\,\varphi$ is both *naming* φ to be called C_i and *claming* that φ holds. Importantly, the formula φ can contain C_i as an atomic formula. This leads to self-reference. For example, the liar's paradox now corresponds to the formula $C_i \neg C_i$.

The logic $\mathcal{L}_0$ is based on game-theoretic semantics GTS which directly extends the standard GTS of FO. Recall that the GTS of FO is based on games played by the *verifier* and *falsifier*. In a game $G(\mathfrak{M},\varphi)$, the verier is trying to show (or verify) that $\mathfrak{M} \models \varphi$ and the falsifier is opposing this, i.e., the falsifier wishes to falsify the claim $\mathfrak{M} \models \varphi$. The players start from the *position* $(\mathfrak{M},\varphi)$ and work their way towards the subformulae of φ. See [6] for a detailed exposition of GTS for first-order logic and [4] for some of the founding ideas behind GTS.

To deal with the novel logic $\mathcal{L}_0$, the rules for the FO-game are extended as follows.

1. In a position $(\mathfrak{M},\mathrm{I}x\,\psi(x))$, the game is continued from a position $(\mathfrak{M}',\psi(u))$ where $\mathfrak{M}'$ is the model obtained by inserting a fresh isolated point u into the domain of $\mathfrak{M}$.
2. In a position $(\mathfrak{M},\mathrm{I}_{R(x_1,\ldots,x_n)}\,\psi(x_1,\ldots,x_n))$, the verifier chooses a tuple $(u_1,\ldots,u_n)$ of elements in $\mathfrak{M}$ and the game is continued from the position $(\mathfrak{M}',\psi(u_1,\ldots,u_n))$ where $\mathfrak{M}'$ is the model obtained from $\mathfrak{M}$ by inserting the tuple $(u_1,\ldots,u_n)$ into the relation R.
3. In a position $(\mathfrak{M},C_i\,\psi)$, the game simply moves to the position $(\mathfrak{M},\psi)$.
4. In an atomic position $(\mathfrak{M},C_i)$, the game moves to the position $(\mathfrak{M},C_i\,\psi)$. Here $C_i\,\psi$ is a subformula of the original formula φ that the semantic game began with. (For simplicity, we assume that at least one such subformula exists in φ. If there are many such subformulae, the verifier chooses which one to continue from.)

Just like the FO-game, the extended game ends only if an atomic position $(\mathfrak{M},R(u_1,\ldots,u_n))$ or $(\mathfrak{M},u=v)$ is encountered. The winner is then decided pre-

cisely as in the FO-game. Thus the extended game can go on forever, as for example the games for $C_i\, C_i$ and $C_i \neg C_i$ always will. In the case the play of the game indeed goes on forever, then that play is won by *neither* of the players. Note that Turing-machines exhibit precisely this kind of behaviour: they can

1. *halt in an accepting state* (corresponding to the verifier winning the semantic game play),
2. *halt in a rejecting state* (corresponding to the falsifier winning),
3. *diverge* (corresponding to neither of the players winning).

Indeed, there is a *precise* correspondence between the logic $\mathcal{L}_0$ and Turing machines. Let $\mathfrak{M} \models^+ \varphi$ (respectively, $\mathfrak{M} \models^- \varphi$) denote that the verifier (respectively, falsifier) has a winning strategy in the game starting from $(\mathfrak{M}, \varphi)$. Let $\mathrm{enc}(\mathfrak{M})$ denote the encoding of the *finite* model $\mathfrak{M}$ according to any standard encoding scheme. Then the following theorem shows that $\mathcal{L}_0$ corresponds to Turing machines so that not only acceptance and rejection but even divergence of Turing computation is captured in a precise and natural way.

Theorem. *For every Turing machine* TM*, there exists a formula $\varphi \in \mathcal{L}_0$ such that*

1. TM *accepts* $\mathrm{enc}(\mathfrak{M})$ *iff* $\mathfrak{M} \models^+ \varphi$
2. TM *rejects* $\mathrm{enc}(\mathfrak{M})$ *iff* $\mathfrak{M} \models^- \varphi$

Vice versa, for every $\varphi \in \mathcal{L}_0$, there is a TM *such that the above conditions hold.*

Technically this is a result in descriptive complexity theory showing that $\mathcal{L}_0$ captures the complexity class RE. We note that the theorem can easily be extended to a logic $\mathcal{L}$ that extends $\mathcal{L}_0$ by constructs that allow one to also delete domain elements and tuples from relations. Furthermore, while the result concerns finite models, it is possible to extend the result to deal with arbitrary models. The idea is to extend Turing machines to suitable hypercomputation models while allowing iteration in $\mathcal{L}_0$ to repeat beyond ω rounds.

Since $\mathcal{L}_0$ captures RE, it cannot be closed under negation. Thus $\neg$ is not the classical negation. However, $\mathcal{L}_0$ has a very natural translation into natural language. The key is to replace *truth* by *verification*. We read $\mathfrak{M} \models^+ \varphi$ as the claim that "*it is verifiable that* $T(\varphi)$" where T is the translation from $\mathcal{L}_0$ into natural language defined as follows. We let T map FO-atoms in the usual way to the corresponding natural language statements, so for example $T(x = y)$ simply reads "x equals y". The atoms C_i are read as they stand, so $T(C_i) = C_i$. The FO-quantifiers translate in the standard way, so $T(\exists x \varphi) =$ *there exists an* x *such that* $T(\varphi)$ and analogously for $\forall x$. Also $\vee$ and $\wedge$ translate in the standard way, so $T(\varphi \vee \psi) = T(\varphi)$ *or* $T(\psi)$ and analogously for $\wedge$. However, $T(\neg\psi) =$ *it is falsifiable that* $T(\psi)$. Thus negation now translates to the dual of verifiability. Concerning the insertion operators, we let $T(\mathrm{I}x\, \varphi) =$ *it is possible to insert a new element* x *such that* $T(\varphi)$. Similarly, we let $T(\mathrm{I}_{R(x_1,\ldots,x_n)}\, \varphi) =$ *it is possible to insert a tuple* $(x_1, \ldots, x_n)$ *into* R *such that* $T(\varphi)$. Finally, we let $T(C_i\, \varphi) =$ *it is possible verify the claim* C_i *which states that* $T(\varphi)$.

Thereby $\mathcal{L}_0$ can be seen as a *simple Turing-complete fragment of natural language*. Indeed, the simplicity of $\mathcal{L}_0$ is one of its main strengths. As typical computationally motivated logics translate into $\mathcal{L}_0$ more or less directly, it can be used as

a natural umbrella logic for studying complexities of logics. This can be advantageous, as the number of different logic formalisms is huge. Furthermore, $\mathcal{L}_0$ offers a top platform for descriptive complexity. Indeed, $\mathcal{L}_0$ can easily capture classes beoynd ELEMENTARY, while no k-th order logic can.

It is interesting to note that $\neg$ can be read as the classical negation on those fragments of $\mathcal{L}_0$ where the semantic games are determined (such as FO). Furthermore, adding a generalized quantifier to $\mathcal{L}_0$ corresponds precisely to adding a corresponding oracle to Turing machines.

[1] Andreas Blass, Yuri Gurevich and Saharon Shelah. Choiceless Polynomial Time. *Ann. Pure Appl. Logic* 100(1-3), (1999), pp. 141–187, doi:10.1016/S0168-0072(99)00005-6.

[2] Egon Börger and Robert F. Stärk. *Abstract State Machines. A Method for High-Level System Design and Analysis.* Springer, 2003, doi:10.1007/978-1-84882-736-3_3.

[3] Y. Gurevich. A new thesis. *American Mathematical Society Abstracts* 6(4), 1985, p. 317.

[4] J. Hintikka. *Logic, Language-Games and Information: Kantian Themes in the Philosophy of Logic.* Oxford: Clarendon Press, 1973.

[5] Antti Kuusisto. Some Turing-complete extensions of first-order logic. In: *The 5th International Symposium on Games, Automata, Logics and Formal Verification, 2014*, pp. 4–17, 2014, doi:10.4204/EPTCS.161.4.

[6] A. L. Mann, G. Sandu and M. Sevenster. *Independence-Friendly Logic - a Game-Theoretic Approach.* Cambridge University Press, 2011, doi:10.1017/CBO9780511981418.

Half a Way Towards Circular Proofs for Kleene Lattices

Stepan Kuznetsov
Steklov Mathematical Institute of the RAS, Moscow, Russia
sk@mi-ras.ru

In their recent paper [6], A. Das and D. Pous proposed a circular proof system for the (in)equational theory of Kleene algebras. This system was then used in order to obtain a modern, syntactic proof of completeness and decidability for this theory [5]. Essentially, constructing such a system became possible because Kleene algebras lie below a line which can be called the *incompleteness barrier*. For systems with inductive constructions, like natural numbers in arithmetic or the Kleene star, for example, there exist two possible axiomatisations. The stronger one uses ω-rules, and the weaker one includes some sort of induction principles. Circular systems are supposed to reflect some form of induction, while ω-rules can be equivalently represented by non-well-founded, but not necessarily circular, proofs.

The question is whether these two axiomatisations give the same theory, in other words, whether induction is complete w.r.t. ω-rules. Theories below this barrier (i.e., for which the answer is "yes") include, for example, the inequational theory of Kleene algebras [10] and Presburger arithmetic. Above this barrier lie Peano arithmetic [9, 17], action logic (the inequational theory of *residuated* Kleene algebras) [4, 7, 12], Horn theories of Kleene algebras [11] *etc.*

For theories above the incompleteness barrier, there is no known method of constructing cut-free circular proof systems. Indeed, known cut-elimination procedures [15, 16, 8, 6, 1] operate with non-well-founded proofs, and applying them to a circular one destroys circularity. Below the barrier, cut-free circularity becomes easier. For Kleene algebras, however, it is still not "out-of-the-box," and cut-free circular fragment of the most natural system [7] is incomplete. This issue was handled by Das and Pous [6] by introducing a more sophisticated calculus whose rules deeply operate in the succedent.

Notice that throughout this extended abstract we discuss only one approach to formalising inductive reasoning, namely, circular proofs. As noticed by one of the reviewers, it could be unclear that by this restriction we do not lose generality. Indeed, different versions of "inductive-style" calculi could yield different logics, see [13, 3]. However, for systems above the incompleteness barrier, this usually happens due to complexity reasons (e.g., Π^0_1-completeness for infinitary action logic). Thus, for these systems *no* complete finitary calculi are possible. Circular systems, however, usually yield somehow interesting fragments (as Peano arithmetic, for example). However, all known circular systems in this case essentially use cut.

Kleene algebras utilise three operations: $\cdot$ (product), $\vee$ (disjunction, or join), and * (Kleene star). In this talk, we focus on the extension of Kleene algebras with $\wedge$ (meet), which yields *Kleene lattices.* We also consider a modification of the original setting, namely, Kleene lattices without the unit, and with positive iteration, $^+$, instead of Kleene star (cf. [12]). We also include the zero constant, 0, which is the smallest element on the lattice, and at the same time the zero w.r.t. product. We

call such structures *positive Kleene lattices*. Such a lattice is called **-continuous*, if for any a we have $a^+ = \sup\{a^n \mid n \geq 1\}$.

The inequational theory (algebraic logic) of positive Kleene lattices can be axiomatised by the following Gentzen-style sequent calculus with an ω-rule (antecedents should be always non-empty):

$$\frac{}{A \vdash A} \qquad \frac{\Gamma, A \cdot B, \Delta \vdash C}{\Gamma, A, B, \Delta \vdash C} \qquad \frac{\Gamma \vdash A \quad \Delta \vdash B}{\Gamma, \Delta \vdash A \cdot B} \qquad \frac{\Gamma \vdash A_i}{\Gamma \vdash A_1 \vee A_2} \qquad \frac{\Gamma, A_1, \Delta \vdash B \quad \Gamma, A_2, \Delta \vdash B}{\Gamma, A_1 \vee A_2, \Delta \vdash B}$$

$$\frac{\Gamma, A_i, \Delta \vdash B}{\Gamma, A_1 \wedge A_2, \Delta \vdash B} \qquad \frac{\Gamma \vdash A_1 \quad \Gamma \vdash A_2}{\Gamma \vdash A_1 \wedge A_2} \qquad \frac{\Gamma_1 \vdash A \quad \ldots \quad \Gamma_n \vdash A}{\Gamma_1, \ldots, \Gamma_n \vdash A^+} \qquad \frac{\left(\Gamma, A^n, \Delta \vdash B\right)_{n=1}^{\infty}}{\Gamma, A^+, \Delta \vdash B}$$

The cut rule is admissible, as shown by Palka [14] for an extension of this calculus with residuals. This system can be equivalently rewritten using non-well-founded proof trees [7, 12] instead of the ω-rule. In such a system, the rules for iteration are replaced by the following ones:

$$\frac{\Gamma, A, \Delta \vdash B \quad \Gamma, A^+, \Delta \vdash B}{\Gamma, A^+, \Delta \vdash B} \qquad \frac{\Gamma \vdash A \quad \Delta \vdash A^+}{\Gamma, \Delta \vdash A^+} \qquad \frac{\Gamma \vdash A}{\Gamma \vdash A^+}$$

These rules are finitely branching; however, now we allow infinite paths in the trees, with the following correctness condition: each such path should include infinitely many principal applications of the left rule for $^+$.

In this talk, we aim to discuss derivability of sequents of a specific form: in these sequents, $\wedge$ can occur in the succedent, but not in the antecedent. An example: $(a^6)^+ \vdash (a^2)^+ \wedge (a^3)^+$.

Let D be a formula and A be its designated subformula; then we write $D[A]$, and allow ourselves to replace A with another formula B, yielding $D[B]$ (notice that we replace only *one occurrence* of A). We claim that the following rules (we call them *deep* right rules) are admissible:

$$\frac{\Gamma \vdash D[A \vee (A \cdot A^+)]}{\Gamma \vdash D[A^+]} \qquad \frac{\Gamma \vdash D[(A \cdot C) \vee (B \cdot C)]}{\Gamma \vdash D[(A \vee B) \cdot C]} \qquad \frac{\Gamma \vdash D[0]}{\Gamma \vdash D[(p \cdot A) \wedge (q \cdot B)]}\ (p \neq q)$$

$$\frac{\Gamma \vdash D[(A \wedge C) \vee (B \wedge C)]}{\Gamma \vdash D[(A \vee B) \wedge C]} \qquad \frac{\Gamma \vdash D[A \cdot (B \cdot C)]}{\Gamma \vdash D[(A \cdot B) \cdot C]} \qquad \frac{\Gamma \vdash D[p \cdot (A \wedge B)]}{\Gamma \vdash D[(p \cdot A) \wedge (p \cdot B)]}$$

$$\frac{\Gamma \vdash D[0]}{\Gamma \vdash D[(p \cdot A) \wedge q]} \qquad \frac{\Gamma \vdash D[p]}{\Gamma \vdash D[p \wedge p]} \qquad \frac{\Gamma \vdash D[0]}{\Gamma \vdash D[p \wedge q]}\ (p \neq q) \qquad \frac{\Gamma \vdash D[0]}{\Gamma \vdash D[0 \cdot A]} \qquad \frac{\Gamma \vdash D[0]}{\Gamma \vdash D[A \cdot 0]}$$

$$\frac{\Gamma \vdash D[0]}{\Gamma \vdash D[p \wedge (q \cdot B)]} \qquad \frac{\Gamma \vdash D[0]}{\Gamma \vdash D[0 \wedge A]} \qquad \frac{\Gamma \vdash D[0]}{\Gamma \vdash D[A \wedge 0]} \qquad \frac{\Gamma \vdash D[A]}{\Gamma \vdash D[0 \vee A]} \qquad \frac{\Gamma \vdash D[A]}{\Gamma \vdash D[A \vee 0]}$$

Indeed, all of these rules are of the form $\frac{\Gamma \vdash D[A]}{\Gamma \vdash D[B]}$ and their admissibility is established by the following general scheme. First, we prove $A \vdash B$. This is checked explicitly for all of these rules. Notice that the 4th deep rule looks like the distributivity principle, which is not generally true in Kleene lattices. However, this "half" of distributivity, $(A \wedge C) \vee (B \wedge C) \vdash (A \vee B) \wedge C$, is indeed derivable (while the converse, $\dashv$, is not). Second, by monotonicity we get $D[A] \vdash D[B]$. Finally, the necessary rule is just the cut of $\Gamma \vdash D[A]$ and $D[A] \vdash D[B]$.

The more interesting fact, which is specific for the fragment without $\wedge$ in antecedents, is the invertibility of the deep rules.

Lemma 1. *For any of the deep rules, if its conclusion is derivable and* Γ *does not contain* $\wedge$*, then the premise of this rule is also derivable.*

This lemma allows normalizing formulae in the succedent in the spirit of normalizing extended regular expressions by Antimirov and Mosses [2]. Starting from an arbitrary succedent D, we apply the deep rule backwards, transforming D into a normal form. This normal form is a disjunction of formulae of the form $p_i \cdot A_i$, where p_i are variables, or just p_i.

For the left-hand side, we develop the leftmost derivation exactly as Das and Pous [6] do. This gives us, on each derivation branch, an antecedent of the form p, Γ, where p is a variable. Now we use the following lemma:

Lemma 2. *If* $p, \Gamma \vdash (p \cdot A_1) \vee \ldots \vee (p \cdot A_k) \vee (q_1 \cdot B_1) \vee \ldots \vee (q_m \cdot B_m)$ *is derivable, and* Γ *does not contain* $\wedge$*, then* $\Gamma \vdash A_1 \vee \ldots \vee A_k$ *is also derivable.*

The original sequent $p, \Gamma \vdash (p \cdot A_1) \vee \ldots \vee (p \cdot A_k) \vee (q_1 \cdot B_1) \vee \ldots \vee (q_m \cdot B_m)$ is derivable from $\Gamma \vdash A_1 \vee \ldots \vee A_k$ by cut with $p \cdot (A_1 \vee \ldots \vee A_k) \vdash (p \cdot A_1) \vee \ldots \vee (p \cdot A_k)$, which can be formulated as the following inference rule:

$$\frac{\Gamma \vdash A_1 \vee \ldots \vee A_k}{p, \Gamma \vdash (p \cdot A_1) \vee \ldots \vee (p \cdot A_k)}$$

and right rule for $\vee$ (weakening on the right).

We arrive at a leftmost proof, and, using the same arguments as Das and Pous [6], we show that it regular in the sense that it includes only a finite number of different subtrees. Thus, we have obtained a circular proof, but in a system with more rules (which include all deep rules and the rule for $\vee$ presented just above). We can also prove a correctness lemma, stating that each cycle in this proof includes an application of the left rule for $^+$. Now, since all the newly added rules can be modelled in the original calculus using cut, we have a correct proof (with cuts) in the old calculus.

Thus, we have constructed a sound, complete, and cut-free system with circular proofs, which is capable of deriving sequents of the form $\Gamma \vdash B$, where B is in the language of positive Kleene lattices ($\cdot$, $\vee$, $\wedge$, $^+$), and Γ additionally is not allowed to include $\wedge$. The question of extending this result to the whole inequational theory of positive Kleene lattices (i.e., with $\wedge$ also in Γ), as well as translating it to the usual Kleene star *, remains open.

[1] Bahareh Afshari and Graham E. Leigh. Cut-free completeness for modal mu-calculus. In *32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–12, 2017.

[2] Valentin M. Antimirov and Peter D. Mosses. Rewriting extended regular expressions. *Theoretical Computer Science*, 143:51–72, 1995.

[3] Stefano Berardi and Makoto Tatsuta. Classical system of Martin-Löf's inductive definitions is not equivalent to cyclic proofs. *Logical Methods in Computer Science*, 15(3), 2019.

[4] Wojciech Buszkowski. On action logic: equational theories of action algebras. *Journal of Logic and Computation*, 17(1):199–217, 2007.

[5] Anupam Das, Amina Doumane, and Damien Pous. Left-handed completeness for Kleene algebra, via cyclic proofs. In Gilles Barthe, Geoff Sutcliffe, and Margus Veanes, editors, *LPAR-22. 22nd International Conference on Logic for Programming, Artificial Intelligence and Reasoning, Awassa, Ethiopia, 16-21 November 2018*, volume 57 of *EPiC Series in Computing*, pages 271–289. EasyChair, 2018.

[6] Anupam Das and Damien Pous. A cut-free cyclic proof system for Kleene algebra. In Renate A. Schmidt and Cláudia Nalon, editors, *TABLEAUX 2017: Automated Reasoning with Analytic Tableaux and Related Methods*, volume 10501 of *Lecture Notes in Computer Science (Lecture Notes in Artificial Intelligence)*, pages 261–277. Springer, 2017.

[7] Anupam Das and Damien Pous. Non-wellfounded proof theory for (Kleene+action) (algebras+lattices). In Dan R. Ghica and Achim Jung, editors, *27th EACSL Annual Conference on Computer Science Logic, CSL 2018, September 4-7, 2018, Birmingham, UK*, volume 119 of *Leibniz International Proceedings in Informatics*, pages 19:1–19:18. Schloss Dagstuhl — Leibniz-Zentrum für Informatik, 2018.

[8] Jérôme Fortier and Luigi Santocanale. Cuts for circular proofs: semantics and cut-elimination. In Simona Ronchi Della Rocca, editor, *Computer Science Logic 2013 (CSL 2013)*, volume 23 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 248–262, Dagstuhl, Germany, 2013. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[9] Kurt Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I. *Monatshefte für Mathematik und Physik*, 38:173–198, 1931.

[10] Dexter Kozen. A completeness theorem for Kleene algebras and the algebra of regular events. *Information and Computation*, 110(2):366–390, 1994.

[11] Dexter Kozen. On the complexity of reasoning in Kleene algebra. *Information and Computation*, 179:152–162, 2002.

[12] Stepan Kuznetsov. The Lambek calculus with iteration: two variants. In Juliette Kennedy and Ruy de Queiroz, editors, *WoLLIC 2017: Logic, Language, Information, and Computation*, volume 10388 of *Lecture Notes in Computer Science*, pages 182–198. Springer, 2017.

[13] Stepan Kuznetsov. *-continuity vs. induction: divide and conquer. volume 12 of *Advances in Modal Logic*, pages 493–510, London, 2018. College Publications.

[14] Ewa Palka. An infinitary sequent system for the equational theory of *-continuous action lattices. *Fundamenta Informaticae*, 78(2):295–309, 2007.

[15] Daniyar Shamkanov. Circular proofs for the Gödel–Löb provability logic. *Mathematical Notes*, 96(4):575–585, 2014.

[16] Daniyar Shamkanov and Yury Savateev. Cut-elimination for the modal Grzegorczyk logic via non-well-founded proofs. In Juliette Kennedy and Ruy de Queiroz, editors, *Logic, Language, Information, and Computation, 24th International Workshop, WoLLIC 2017*, volume 10388 of *Lecture Notes in Computer Science*, pages 358–371. Springer, 2017.

[17] Alex Simpson. Cyclic arithmetic is equivalent to Peano arithmetic. In Javier Esparza and Andrzej S. Murawski, editors, *FoSSACS 2017: Foundations of Software Science and Computation Structures*, volume 10203 of *Lecture Notes in Computer Science*, pages 283–300. Springer, 2017.

Higher-order Parity Automata

Paul-André Melliès
Institut de Recherche en Informatique Fondamentale CNRS, Université Paris Diderot
mellies@irif.fr

In this introductory talk, I will explain how the familiar notion of parity tree automaton may be nicely extended to a higher-order notion of parity automaton which recognises infinitary lambda-terms instead of infinitary trees. One main result of the theory (LICS 2017) is that every simply-typed lambda-term extended with inductive and coinductive fixpoint operators Y_μ and Y_ν induces an infinitary lambda-term whose recognition by a higher-order parity automaton is decidable. The proof of the theorem relies on two entirely independent component, which I will examine in the talk: (1) the formulation of Scott models in the automata-friendly language of linear logic (2) a non-trivial and enlightening correspondence theorem between the accepting run-trees of a lambda-term and of its infinitary normal form.

Truth and Fix-points for Almost Negative Formulae

Mattias Granberg Olsson
University of Gothenburg, Gothenburg, Sweden
mattias.granberg.olsson@gu.se

In a series of papers [4, 1, 7, 2, 3] and using different methods, Buchholz, Arai and Rüede and Strahm proved that the theory $\widehat{\mathrm{ID}}_1^{\mathrm{i}}$ of fix-points for strictly positive operator forms in intuitionistic logic is conservative over Heyting Arithmetic (HA). This is in contrast to the classical situation, where already the theory $\widehat{\mathrm{ID}}_1(\Pi_2)$ of fix-points for strictly positive Π_2 operator forms proves the consistency of Peano Arithmetic.

This talk will outline an alternative proof of a substantial fragment of the intuitionistic conservativity result. We consider the "realizability translation" of $\widehat{\mathrm{ID}}_1^{\mathrm{i}}$, a theory we call $\underline{\mathrm{r}}\,\widehat{\mathrm{ID}}_1^{\mathrm{i}}$. Since the image of a formula under this "translation" is an *almost negative* formula, i.e. a formula without disjunction and where $\exists$ only occurs in front of primitive recursive term-equations, we study a theory $\widehat{\mathrm{ID}}_1^{\mathrm{i}}(\Lambda)$ of fix-points for strictly positive almost negative operator forms.

To this end we introduce an exhaustive hierarchy Λ_n of almost negative formulae, similar to the formula hierarchies introduced by e.g. Burr in [5, 6], having the following properties:

- If we replace a relation (other than $=$) in strictly positive position in a Λ_n-formula by a Λ_n-formula, the result is still Λ_n.
- The transformation of Λ_n into a canonical form preserves nestings of $\rightarrow$, and in particular strictly positive positions.

We also introduce a theory Th_n of type-free truth over HA for each level of the hierarchy, and a "limit" theory Th of type-free truth for all truth-positive almost negative sentences over (and incorporating) the relevant fragment of the disquotational truth theory $\mathrm{UTB}^{\mathrm{i}}$ of uniform Tarski-biconditionals. We show that each of these truth theories Th_n interprets the corresponding fix-point theory $\widehat{\mathrm{ID}}_1^{\mathrm{i}}(\Lambda_n)$ for strictly positive Λ_n operator forms, and that, similarly, the truth theory Th interprets $\widehat{\mathrm{ID}}_1^{\mathrm{i}}(\Lambda)$. Finally we show that each level of the formula-hierarchy defines its own truth-predicate already in HA, which in particular shows that both $\widehat{\mathrm{ID}}_1^{\mathrm{i}}(\Lambda)$ and the theory-hierarchies are conservative over HA.

This is joint work in progress with my supervisor Graham Leigh. Earlier versions of this talk were given at the Logic Colloquium 2019 and at Proof, Complexity and Computation 2019.

[1] Toshiyasu Arai. Some results on cut-elimination, provable well-orderings, induction and reflection. *Annals of Pure and Applied Logic*, 95(1–3):93–184, 1998.
[2] Toshiyasu Arai. Intuitionistic fixed point theories over Heyting arithmetic. In Solomon Feferman and Wilfried Sieg, editors, *Proofs, Categories and Computations. Essays in Honor of Grigori Mints*, volume 13 of *Tributes*, pages 1–14. College Publications, July 2010.
[3] Toshiyasu Arai. Quick cut-elimination for strictly positive cuts. *Annals of Pure and Applied Logic*, 162(10):807–815, 2011.

[4] Wilfried Buchholz. An intuitionistic fixed point theory. *Archive for Mathematical Logic*, 37(1):21–27, 1997.

[5] Wolfgang Burr. Fragments of Heyting Arithmetic. *The Journal of Symbolic Logic*, 65(3):1223–1240, September 2000.

[6] Wolfgang Burr. The intuitionistic arithmetical hierarchy. In Jan van Eijk, Vincent van Oostrom, and Albert Visser, editors, *Logic Colloquium '99: Proceedings of the Annual European Summer Meeting of the Association for Symbolic Logic, Held in Utrecht, Netherlands, August 1–6, 1999*, number 17 in Lecture Notes in Logic, pages 51–59. A K Peters, Ltd., 2004.

[7] Christian Rüede and Thomas Strahm. Intuitionistic fixed point theories for strictly positive operators. *Mathematical Logic Quaterly*, 48(2):195–202, 2002.

Yablo's Paradox Revisited

Paulo Guilherme Santos[a] and Reinhard Kahle[b]

[a]Centro de Matemática e Aplicações, FCT-NOVA, Caparica, Portugal
pgd.santos@campus.fct.unl.pt
[b]Universität Tübingen, Tübingen, Germany
kahle@mat.uc.pt

Abstract

Stephen Yablo proposed a paradox which, according to his evaluation, does not depend on self-reference. In this note, we render Yablo's paradox, first, by use of an order relation, showing that it does not rely on arithmetic. Secondly, we formalize it in Linear Temporal Logic. This formalization uncovers the underlying logical structure of Yablo's paradox.

Introduction

In [9], Stephen Yablo presented the following paradox: "Imagine an infinite sequence of sentences $S_1, S_2, S_3, \ldots$, each to the effect that every subsequent sentence is untrue:

(S_1) for all $k > 1$, S_k is untrue,
(S_2) for all $k > 2$, S_k is untrue,
(S_3) for all $k > 3$, S_k is untrue, ..."

This is a paradox, in the sense that none of the S_n could be true nor false without provoking a contradition. Yablo's concern with this paradox is, that—according to his understand—it does not involve self-reference. The given formulation of the paradox is problematic, first of all, because of the use of the dots. Frege heavily critizised such dots in his polemics against Thomae [3, §§125ff], a critizism which was also appreciated by Hilbert. In the 1993 notation, one can characterize S_n as a sentence fulfilling the following equivalence:

$$S_n \leftrightarrow (\forall k > n.\, \neg S_k). \tag{Y}$$

Even without dots, these equivalences require a "function" $S\,.$ —or, more adequately, in λ notation: $\lambda x.S_x$—which is clearly common to all sentences, and which could be regarded as a self-referential feature, somehow of higher type character. Thus, with a temporal reading of the indices, one could read S_n as "I will remain false from the next moment on." The aim of this note, however, is not to decide the alleged non-self-referential nature of Yablo's paradox, but rather to uncover its underlying logical structure.

This work was funded by the FCT-project Hilbert's 24$^{\text{th}}$Problem: PTDC/MHC-FIL/2583/2014.

An Order Theory to Express Yablo's Paradox

Several authors have studied Yablo's Paradox in the realm of arithmetic to discuss its self-referential nature, see, for instance, [8], [7], [6], and [1]. We will show that Yablo's Paradox does not depend on arithmetic, but just on the availability of a linear order. The order theory, we are going to present, is similar to the "scheme (B)" of [6] and to the "theory **Y**" of [5].

Definition 1. Let $\mathcal{T}$ be a first-order theory with a binary relation symbol $<$ and the following two axioms:

(AxT1) $\forall x.\forall y.\forall z.\, x < y \wedge y < z \rightarrow x < z$;
(AxT2) $\forall x.\exists y.\, x < y$.

Now we include a formal version of (Y) in the theory $\mathcal{T}$.

Definition 2. Let $\mathcal{Y}$ be the theory $\mathcal{T}$ with a new unary relation symbol S and the following axiom:

(AxY) $\forall x.\, (S(x) \leftrightarrow (\forall k > x.\, \neg S(k)))$.

Of course, (AxY) is just the formal counterpart of (Y).

Theorem 3. *The theory $\mathcal{Y}$ is inconsistent.*

The previous result can be found in [5] and in [6]. $\mathcal{Y}$ is not only suitable to represent Yablo's Paradox, it is also minimal in the sense of the following theorem.

Theorem 4. *The theories $\mathcal{Y} - (\text{AxT1})$, $\mathcal{Y} - (\text{AxT2})$, and $\mathcal{Y} - (\text{AxY})$ are consistent.*

Thus, $\mathcal{Y}$ is a (minimal) theory to formalize Yablo's Paradox. We do not need any arithmetics but jsut some elementary properties for the order relation of $\langle \mathbb{N}, <_{\mathbb{N}} \rangle$.

Yablo's Paradox in Linear Temporal Logic

In accordance with a reading of the order in temporal terms, temporal logic seems to be even more suitable framework to formalize Yablo's Paradox. We will consider Linear Temporal Logic (LTL) as given in [4]. LTL is a propositional logic with the temporal operators X, G, and F, with the following meanings:

X : "In the next moment it will be the case that …"
G : "It will always be the case that …"
F : "It will at some time be the case that …"

In [2, pg. 21] the semantics and some properties of LTL are presented. In the following, V is a set of propositional variables.

Definition 5. [2, pg. 21] A *temporal* (or *Kripke*) *structure* for V is an infinite sequence $K = \langle \eta_i \rangle_{i \in \mathbb{N}}$, where for each $i \in \mathbb{N}$, $\eta_i : V \rightarrow \{0, 1\}$. For each K and $i \in \mathbb{N}$ we define $K_i(\varphi)$, where φ is a LTL formula, inductively by:

1.) For each $v \in V$, $K_i(v) = \eta_i(v)$;

2.) $K_i(\bot) = 0$;

3.) $K_i(\varphi \to \psi) = 1$ if, and only if, $K_i(\varphi) = 0$ or $K_i(\psi) = 1$;

4.) $K_i(X\varphi) = K_{i+1}(\varphi)$;

5.) $K_i(G\varphi) = 1$ if, and only if, for every $j \geq i$, $K_j(\varphi) = 1$;

6.) $K_i(F\varphi) = 1$ if, and only if, there is $j \geq i$ such that $K_j(\varphi) = 1$.

Observe that, in LTL, the future includes the present. Validity and logical equivalence in LTL are defined in the usual way [2, pg. 22]. Now we move to interpret Yablo's Paradox in LTL. The idea behind what we are going to do is the following: we are going to consider the sentences $S_0, S_1, \ldots$ from Yablo's Paradox as a unique sentence whose truth value is changing in time, this means that we are going to abstract from the particularities of Yablo's Paradox and consider the sentences $S_0, S_1, \ldots$ as a whole. More precisely, we are going to consider a formula S whose truth value changes in time according to condition (Y) from before. As we are considering a sentence S that changes according to (Y), then we are considering each S_i as being $K_i(S)$ (for every temporal structure K). Hence, we have the following: $K_i(S) = 1 \iff \forall j > i.\, K_j(S) = 0$. That is, $K_i(S) = 1 \iff \forall j \geq i + 1.\, K_j(S) = 0$. The previous equivalence in LTL (quantified universally in the temporal structure K) is the formula:

$$S \leftrightarrow XG\neg S. \tag{TY}$$

So, Yablo's Paradox in Temporal Logic is, intuitively, expressed by "This sentence is true if in the next moment it is false in the future" or, if one does not consider that the future includes the present, by "This sentence is true if it is false in the future". One can derive a contradiction in a very similar way as before by thinking (intuitively) about the previous sentences in the natural language. The next result formalises that intuitive approach of time in LTL—if one considers (TY) in LTL one can derive a contradiction, i.e., LTL + (TY) is inconsistent.

Theorem 6. *If the formula $S \leftrightarrow XG\neg S$ is satisfiability in LTL, then one can derive $\bot$ in LTL.*

The concept of time that we are using in LTL is a natural one, replacing the order relation $<$. Thus, we are not adding a new concept to our formulation of Yablo's Paradox, instead we are simply giving a name to some properties of the considered relation.

Within LTL, however, the paradoxical nature of (TY) is clearly due to self-reference. But clearly this suffices to consider Yablo's Paradox self-referential, since to consider a given sentence self-referential there ought to be a formulation of it that conveys non-trivial information about itself, it does not mean that all formulations do so.

[1] R.T. Cook. *The Yablo Paradox: An Essay on Circularity*. OUP Oxford, 2014.

[2] Stephan Merz Fred Kroger. *Temporal Logic and State Systems*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2008.

[3] Gottlob Frege. *Grundgesetze der Arithmetik, begriffschriftlich abgeleitet*, volume 2. Hermann Pohle, Jena, 1903.

[4] Valentin Goranko and Antony Galton. Temporal logic. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, winter 2015 edition, 2015.
[5] Volker Halbach and Shuoying Zhang. Yablo without Gödel. *Analysis*, 77(1):53–59, 2017.
[6] Jeffrey Ketland. Bueno and colyvan on yablo's paradox. *Analysis*, 64(2):165–172, 2004.
[7] Lavinia María Picollo. Yablo's paradox in second-order languages: Consistency and unsatisfiability. *Studia Logica*, 101(3):601–617, Jun 2013.
[8] Graham Priest. Yablo's paradox. *Analysis*, 57(4):236–242, 1997.
[9] Stephen Yablo. Paradox without self–reference. *Analysis*, 53(4):251–252, 1993.

Non-well-founded Derivations in the Gödel-Löb Provability Logic

Daniyar S. Shamkanov
Steklov Mathematical Institute of Russian Academy of Sciences, Gubkina str. 8, 119991, Moscow, Russia
National Research University Higher School of Economics, Moscow, Russia
daniyar.shamkanov@gmail.com

The Gödel-Löb provability logic GL is a modal logic, which can be characterized by strictly partially ordered Kripke frames without infinite ascending chains. This logic is sound and complete with respect to its arithmetical interpretation, where the modal connective $\Box$ corresponds to the standard formal provability predicate in Peano arithmetic [15].

Several years ago a proof-theoretic presentation for the Gödel-Löb provability logic GL in the form of a sequent calculus allowing non-well-founded proofs was given in [10, 5]. In this talk I would like to consider Hilbert-style non-well-founded derivations in GL and discuss different forms of semantics for GL with the obtained derivability relation.

A *non-well-founded derivation, or ∞-derivation, in* GL is a (possibly infinite) tree whose nodes are marked by modal formulas and that is constructed according to the rules (mp) and (nec):

$$\mathsf{mp}\ \frac{A \qquad A \to B}{B}, \qquad \mathsf{nec}\ \frac{A}{\Box A}.$$

In addition, any infinite branch in an ∞-derivation must contain infinitely many applications of the rule (nec). Below is an example of an ∞-derivation:

$$\mathsf{mp}\ \frac{\mathsf{nec}\ \dfrac{\mathsf{mp}\ \dfrac{\mathsf{nec}\ \dfrac{\mathsf{mp}\ \dfrac{\mathsf{nec}\ \dfrac{\vdots}{\Box p_3}}{p_2}\qquad \Box p_3 \to p_2}{\Box p_2}}{p_1}\qquad \Box p_2 \to p_1}{\Box p_1}\qquad \Box p_1 \to p_0}{p_0}.$$

In terms of abstract algebraic logic, the algebraic counterpart of the Gödel-Löb provability logic GL is the variety of so called Magari algebras [6, 14], where a *Magari algebra* $\mathcal{A} = (Y, \wedge, \vee, \to, 0, 1, \Box)$ is a Boolean algebra $(Y, \wedge, \vee, \to, 0, 1)$ together with a unary map $\Box\colon Y \to Y$ satisfying the identities:

$$\Box 1 = 1, \quad \Box(x \wedge y) = \Box x \wedge \Box y, \quad \Box(\Box x \to x) = \Box x.$$

The syntactic consequence relation given by ∞-derivations in GL defines a generalized quasivariety of certain Magari algebras, which I call Pakhomov-Walsh-founded, or $\Box$-founded. A Magari algebra is *Pakhomov-Walsh-founded*[1]*, or $\Box$-founded,* if, for every sequence $(a_i)_{i\in\mathbb{N}}$ of elements of this algebra such that $\Box a_{i+1} \leqslant$

[1]This notion was inspired by a paper of F. Pakhomov and J. Walsh [8].

a_i, one has $a_0 = 1$. There is an equivalent definition of this notion in terms of the binary relation $\prec_{\mathcal{A}}$ on a Magari algebra $\mathcal{A}$:

$$a \prec_{\mathcal{A}} b \Longleftrightarrow b \leqslant \Diamond a,$$

where $\Diamond a = \neg\Box\neg a$. Notice that, for any Magari algebra $\mathcal{A}$, the relation $\prec_{\mathcal{A}}$ is a strict partial order on $\mathcal{A} \setminus \{0\}$. A Magari algebra $\mathcal{A}$ is $\Box$-*founded* if and only if the partial order $\prec_{\mathcal{A}}$ on $\mathcal{A} \setminus \{0\}$ is well-founded. It is the case that the Gödel-Löb provability logic with non-well-founded derivations is strongly sound and complete for its algebraic interpretation over $\Box$-founded Magari algebras.

Neighbourhood semantics is a form of semantics which is less abstract than algebraic one. It is a generalization of Kripke semantics independently developed by D. Scott and R. Montague in [9] and [7]. A neighbourhood frame can be defined as a pair $(X, \Box)$, where X is a set and $\Box$ is an unary operation in $\mathcal{P}(X)$. The Gödel-Löb provability logic GL is compact for its neighbourhood interpretation, which immediately implies that GL is strongly neighbourhood complete (see [12, 2]). However, this completeness result holds for the case of the so-called local semantic consequence relation. Recall that, over neighbourhood GL-models, a formula A is a local semantic consequent of Γ if for any neighbourhood GL-model $\mathcal{M}$ and any world x of $\mathcal{M}$

$$(\forall B \in \Gamma\ \mathcal{M}, x \vDash B) \Rightarrow \mathcal{M}, x \vDash A.$$

A formula A is a global semantic consequent of Γ if for any neighbourhood GL-model $\mathcal{M}$

$$(\forall B \in \Gamma\ \mathcal{M} \vDash B) \Rightarrow \mathcal{M} \vDash A.$$

In the paper [11], I established that GL enriched with non-well-founded derivations is strongly neighbourhood complete in the case of the global semantic consequence relation. In other words, there is an ∞-derivation of a formula A from the set of assumptions Γ if and only if the formula A is a global semantic consequent of Γ over neighbourhood GL-frames. An algebraic generalization of this result states that a Magari algebra is $\Box$-founded if and only if it is embeddable into an atomic complete Magari algebra.

There is another more concrete form of semantics for the Gödel-Löb provability logic with non-well-founded derivations. This system appears to be countably complete for its neighbourhood interpretation over certain tree-like neighbourhood GL-frames resembling Kripke frames. Notice that there is a close connection between neighbourhood GL-frames and scattered topological spaces. H. Simmons [13] and L. Esakia [4] showed that any scattered topological space could be considered as a neighbourhood GL-frame and any neighbourhood GL-frame could be obtained from the uniquely defined scattered topological space. Thus, scattered topological spaces and neighbourhood GL-frames are essentially the same notions. Any tree-like neighbourhood GL-frame mentioned above is homeomorphic to a countable ordinal with the interval topology. It follows that the Gödel-Löb provability logic with non-well-founded derivations is countably complete for a slightly modified neighbourhood interpretation over the first uncountable ordinal equipped with the interval topology. This latter observation resembles the result of M. Abashidze [1] and A. Blass [3] that GL is locally complete for its neighbourhood interpretation over the ordinal ω^ω (see also [2]).

[1] M. Abashidze. Ordinal completeness of the Gödel-Löb modal system. In Russian. In: *Intensional logics and the logical structure of theories.* Tbilisi: Metsniereba, 1985, pp. 49-73.
[2] J.P. Aguilera and D. Fernndez-Duque. Strong Completeness of Provability Logic for Ordinal Spaces. In: *The Journal of Symbolic Logic* 82.1 (2 2017), pp. 608-638.
[3] A. Blass. Infinitary combinatorics and modal logic. In: *The Journal of Symbolic Logic* 55.2 (1990), pp. 761-778.
[4] L. Esakia. Diagonal constructions, Löb's formula and Cantor's scattered space. In Russian. In: *Studies in Logic and Semantics* 132.3 (1981), pp. 128-143.
[5] R. Iemhoff. Reasoning in circles. In: *Liber Amicorum Alberti. A Tribute to Albert Visser.* Ed. by Jan van Eijck et al. London: College Publications, 2016, pp. 165-178.
[6] R. Magari. The diagonalizable algebras (the algebraization of the theories which express Theor. II). In: *Bollettino dell'Unione Matematica Italiana.* 4th ser. 12 (1975), pp. 117-125.
[7] R. Montague. Universal Grammar. In: *Theoria* 36 (3 1970), pp. 373-398.
[8] F. Pakhomov and J. Walsh. *Reflection Ranks and Ordinal Analysis.* 2018. url: arXiv: 1805.02095.
[9] D. Scott. Advice in modal logic. In: *Philosophical problems in Logic.* Ed. by K. Lambert. Reidel, 1970.
[10] D. Shamkanov. Circular proofs for the Gödel-Löb provability logic. In: *Mathematical Notes* 96.3 (2014), pp. 575-585.
[11] D. Shamkanov. Global neighbourhood completeness of the Gödel-Löb provability logic. In: *Logic, Language, Information, and Computation. 24th International Workshop, WoL- LIC 2017 (London, UK, July 18-21, 2017).* Ed. by Juliette Kennedy and Ruy de Queiroz. Lecture Notes in Computer Science 103888. Springer, 2017, 358-371.
[12] V. Shehtman. On neighbourhood semantics thirty years later. In: *We Will Show Them! Essays in Honour of Dov Gabbay.* Ed. by S. Artemov et al. Vol. 2. London: College Publications, 2005, pp. 663-692.
[13] H. Simmons. Topological aspects of suitable theories. In: *Proceedings of the Edinburgh Mathematical Society* 19.4 (1975), pp. 383-391.
[14] C. Smoryński. *Self-Reference and Modal Logic.* New York: Springer, 1985.
[15] R. Solovay. Provability Interpretations of Modal Logic. In: *Israel Journal of Mathematics* 25 (1976), pp. 287-304.

Efficient Validation of FOL_{ID} Cyclic Induction Reasoning

Sorin Stratulat
Université de Lorraine, CNRS, LORIA, F-57000 Metz, France,
sorin.stratulat@univ-lorraine.fr

Abstract

Checking the soundness of the cyclic induction reasoning for first-order logic with inductive definitions (FOL_{ID}) is decidable but the standard checking method is based on a doubly exponential complement operation for Büchi automata. We present a polynomial method 'semi-deciding' this problem; its most expensive steps recall the comparisons with multiset path orderings. In practice, it has been integrated in the Cyclist prover and successfully checked all the proofs included in its distribution.

FOL_{ID} cyclic proofs may also be hard to certify. Our method helps to represent the cyclic induction reasoning as being well-founded, where the ordering constraints are automatically built from the analysis of the proofs. Hence, it creates a bridge between the two induction reasoning methods and opens the perspective to use the certification methods adapted for well-founded induction proofs.

Introduction. Cyclic pre-proofs for the classical first-order logic with inductive predicates (FOL_{ID}) have been extensively studied in [1, 2, 4]. They are finite sequent-based derivations where some terminal nodes, called *buds*, are labelled with sequents already occurring in the derivation, called *companions*. Bud-companion (BC) relations, graphically represented as *back-links*, are described by an induction function attached to the derivation, such that only one companion is assigned to each bud, but a node can be the companion of one or several buds. The pre-proofs can be viewed as digraphs whose cycles, if any, are introduced by the BC-relations.

It is easy to build unsound pre-proofs, for example by creating a BC-relation between the nodes labelled by the sequents from a stuttering step. The classical soundness criterion is the *global trace condition*. Firstly, the paths are annotated by traces built from inductive antecedent atoms (IAAs) found on the lhs of the sequents in the path, then it is shown that for every infinite path p in the cyclic derivation of a false sequent, there is some trace following p such that all successive steps starting from some point are decreasing and certain steps occurring infinitely often are strictly decreasing w.r.t. some semantic ordering. We say that a *progress point* happens in the trace when a step is strictly decreasing. A *proof* is a pre-proof if every infinite path has an infinitely progressing trace starting from some point.

The standard checking method [2] of the global trace condition is decidable but based on a doubly exponential complement operation for Büchi automata [5]. It has been implemented in the Cyclist prover [3] and experiments showed that the soundness checking can take up to 44% of the proof time. On the other hand, a less costly, polynomial-time, checking method has been presented in [7, 9].[1] The pre-proof to be checked is firstly normalized into a digraph consisting of a set of

[1] [6] tackles a similar question, although from a more theoretical viewpoint.

derivation trees to which is attached an extended induction function. The resulting digraph counts among its roots the companions and the root of the pre-proof to be checked. The normalized pre-proof is a proof if every strongly connected component (SCC) of the digraph satisfies some ordering constraints, similar to those used for certifying cyclic Noetherian induction proofs [8].

Implementation. The method has been implemented in Cyclist. Cyclist builds the pre-proofs using a depth-first search strategy that aims at closing open nodes as quickly as possible. Whenever a new cycle is built, model-checking techniques provided by an external model checker are called to validate it. If the validation result is negative, the prover backtracks by trying to find another way to build new cycles. Hence, it may happen that the model checker be called several times during the construction of a pre-proof.

To each root r from the digraph $\mathcal{P}$ of a normalized pre-proof tree-set, the method attaches a measure $\mathcal{M}(r)$ consisting of a multiset of IAAs of the sequent labelling r, denoted by $S(r)$. One of the challenges is to find the good measures such that the ordering constraints be satisfied. A procedure for computing these measure values is given by Algorithm 1.

Algorithm 1 GenOrd($\mathcal{P}$): to each root r of $\mathcal{P}$ is attached a measure $\mathcal{M}(r)$

for all root r **do**
  $\mathcal{M}(r) := \emptyset$
end for
for all rb-path $r \rightarrow b$ from a non-singleton SSC **do**
  if there is a trace between an IAA A of $S(b)$ and an IAA A' of $S(r)$ **then**
    add A to $\mathcal{M}(rc)$ and A' to $\mathcal{M}(r)$, where rc is the companion of b
  end if
end for

Firstly, the measures attached to each root are empty sets. Then, for each root-bud (rb) path from a cycle, denoted by $r \rightarrow b$, and for every trace along $r \rightarrow b$, leading some IAA of $S(r)$ to another IAA of $S(b)$, we add the corresponding IAAs to the measures of r and the companion of b, respectively. Since the number of rb-paths is finite, Algorithm 1 terminates.

Algorithm 1 may compute measure values that do not pass the comparison test for some non-singleton SCCs that are validated by the model checker. For this case, we considered an improvement consisting of the incremental addition of IAAs from a root sequent that are not yet in the measure value of the corresponding root r. Since the validating orderings are multiset extensions of multiset path orderings, such an addition does not affect the comparison value along the rp-paths starting from r. On the other hand, it may affect the comparison tests for the rp-paths ending in the companions of r. This may duplicate some IAAs from the value measure of the roots from the rp-paths leading to these companions. The duplicated IAAs have to be processed as any incrementally added IAA, and so on, until no changes are performed any more.

Table 1 illustrates some statistics about the proofs of the conjectures considered in Table 1 from [3], checked with the standard as well as our improved method. The

IAAs are indexed in Cyclist to facilitate the construction of traces; the way they are indexed influence how the pre-proofs are built. The column labelled 'Time-E' is the proof time measured in milliseconds with our method. Similarly, the 'Time' column displays the proof time using the standard method, while 'SC%' shows the percentage of time taken to check soundness using the model checker. 'Depth' shows the depth of the proof, 'Nodes' the number of nodes in the proof, and 'Bckl.' the number of back-links in the proof. The last column shows the number of calls to the model checker as (calls on unsound proof)/(total calls) when our method is not used. The proofs have been performed on a MacBook Pro featuring a 2,7 GHz Intel Core i7 processor and 16 GB of memory. We can notice that, by using our method, the execution time is reduced by a factor going from 1.43 to 5.

Theorem	**Time-E**	Time	SC%	Depth	Nodes	Bckl.	Uns./All
$O_1 x \vdash Nx$	**2**	7	61	2	9	1	0/1
$E_1 x \vee O_2 x \vdash Nx$	**4**	11	63	3	19	2	0/4
$E_1 x \vee O_1 x \vdash Nx$	**2**	9	77	2	13	2	2/5
$N_1 x \vdash Ox \vee Ex$	**3**	7	52	2	8	1	0/1
$N_1 x \wedge N_2 y \vdash Q(x, y)$	**297**	425	40	4	19	3	168/181
$N_1 x \vdash Add(x, 0, x)$	**1**	5	76	1	7	1	0/1
$N_1 x \wedge N_2 y \wedge Add_3(x, y, z) \vdash Nz$	**8**	14	38	2	8	1	4/5
$N_1 x \wedge N_2 y \wedge Add_3(x, y, z) \vdash Add(x, sy, sz)$	**15**	22	32	2	14	1	9/10
$N_1 x \wedge N_2 y \vdash R(x, y)$	**266**	484	48	4	35	5	149/170

Table 1: Statistics for Cyclist proofs checked with the standard and our methods.

Even when using the improved version of Algorithm 1, the method may propose measure values that do not pass the comparison tests. Indeed, this was happened once, while proving $N_1 x \wedge N_2 y \vdash R(x, y)$. Hopefully, the prover backtracked and finally found the same proof as that built using the model checker. The source code of the implementation can be downloaded at https://members.loria.fr/SStratulat/files/e-cyclist.zip

[1] J. Brotherston. Cyclic proofs for first-order logic with inductive definitions. In *Proceedings of TABLEAUX-14*, volume 3702 of *LNAI*, pages 78–92. Springer-Verlag, 2005.

[2] J. Brotherston. *Sequent Calculus Proof Systems for Inductive Definitions*. PhD thesis, University of Edinburgh, November 2006.

[3] J. Brotherston, N. Gorogiannis, and R. L. Petersen. A generic cyclic theorem prover. In *APLAS-10 (10th Asian Symposium on Programming Languages and Systems)*, volume 7705 of *LNCS*, pages 350–367. Springer, 2012.

[4] J. Brotherston and A. Simpson. Sequent calculi for induction and infinite descent. *Journal of Logic and Computation*, 21(6):1177–1216, 2011.

[5] M. Michel. Complementation is more difficult with automata on infinite words. Technical report, CNET, 1988.

[6] R. Nollet, A. Saurin, and C. Tasson. Pspace-completeness of a thread criterion for circular proofs in linear logic with least and greatest fixed points. In S. Cer-

rito and A. Popescu, editors, *TABLEAUX'2019*, pages 317–334. Springer International Publishing, 2019.

[7] S. Stratulat. Cyclic proofs with ordering constraints. In R. A. Schmidt and C. Nalon, editors, *TABLEAUX 2017 (26th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods)*, volume 10501 of *LNAI*, pages 311–327. Springer, 2017.

[8] S. Stratulat. Mechanically certifying formula-based Noetherian induction reasoning. *Journal of Symbolic Computation*, 80, Part 1:209–249, 2017.

[9] S. Stratulat. Validating back-links of $\mathrm{FOL}_{\mathrm{ID}}$ cyclic pre-proofs. In S. Berardi and S. van Bakel, editors, *CL&C'18 (Seventh International Workshop on Classical Logic and Computation)*, number 281 in EPTCS, pages 39–53, 2018.

Circularity in Graph Normal Form

Michał Walicki
Department of Informatics, University of Bergen, Norway
michal@ii.uib.no

Graph Normal Form for propositional logic was introduced in [1] and applied in [6] to analysis of semantic paradoxes. Here, we begin by noting its generalization as a normal form also for first-order logic. Given a language and a set D, by $\mathbb{T}_D$ we denote the free term algebra over D, and by $\mathrm{AT}(D)$ the atomic formulas applying predicate symbols to elements of D. The usual atoms are thus $\mathrm{AT}(\mathbb{T}_X)$, for some set of variables X.

Definition 1. A formula is in GNF if it is an equivalence where

- the left side is an atom, $LS \in \mathrm{AT}(\mathbb{T}_X)$,
- the right side, RS, is a (universally quantified) conjunction of negated atoms,
- all free variables of RS occur in LS, $\mathcal{V}(RS) \subseteq \mathcal{V}(LS)$.

A theory, i.e., a set of formulas, Γ is in GNF if each formula of Γ is in GNF.

Fact 2. *For each theory Γ, there is a theory $GNF(\Gamma)$ in GNF with*

$$Mod(\Gamma) \leftrightarrows Mod(GNF(\Gamma)).$$

($\leftrightarrows$ denotes injections both ways.) The following example conveys the main idea of constructing *GNF*, based on a form of Morleyzation.

Example 3. In a PNF formula, we replace all $\exists$ by $\neg\forall\neg$, and then introduce fresh predicates for each subformula starting with a negation followed by a block of universal quantifiers, $\neg\forall x_1 ... \forall x_n ...$ This yields a definitional extension GNF^-. *GNF* is obtained by adding one more fresh predicate symbol and equivalence GNF^+, shown in the bottom line.

$$\begin{array}{rl} & \phi = \forall x \exists y P x y \\ GNF^-(\phi): & Az \leftrightarrow \forall x \neg S x \\ & Sx \leftrightarrow \forall y \neg P x y \\ \\ GNF^+(\phi): & A'z \leftrightarrow \neg Az \wedge \neg A'z \end{array} \qquad \begin{array}{rl} & \psi = \exists y \forall x P x y \\ GNF^-(\psi): & Nz \leftrightarrow \neg Rz \\ & Rz \leftrightarrow \forall y \neg \underline{Q} y \\ & \underline{Q} y \leftrightarrow \forall x \neg \overline{P} x y \\ & \overline{P} x y \leftrightarrow \neg P x y \\ GNF^+(\psi): & N'z \leftrightarrow \neg Nz \wedge \neg N'z \end{array}$$

Propositional combinations of atoms, occurring possibly in the RSs of the final equivalences of GNF^-, can be brought to the required format, typically, introducing additional predicate symbols.

A GNF theory can be represented by a graph, which here means a digraph, namely, a pair $G = (V_G, A_G)$ with vertices V_G and edges $A_G \subseteq V_G \times V_G$. For a vertex $x \in V_G$, we denote $A_G(x) = \{y \in V_G \mid (x,y) \in A_G\}$ and $A_G^-(y) = \{x \in V_G \mid (x,y) \in A_G\}$. This notation is extended to sets, e.g., $A_G(X) = \bigcup_{v \in X} A_G(v)$, etc.

Our approach can be applied to full FOL with equality, but to stay within 3 pages, we restrict it to FOL without equality or function symbols, FOL^-, and assume that

each predicate symbol B occurs at most once in LS of some formula of a theory, to which we refer as B's definition. Letting B_i range over predicate symbols from the language of a GNF theory Γ, a formula from Γ is represented schematically by the pattern (where some B_i may be identical to B_0):

$$B_0x \leftrightarrow \forall y\big(\neg B_1xy \wedge \ldots \wedge \neg B_nxy\big). \tag{4}$$

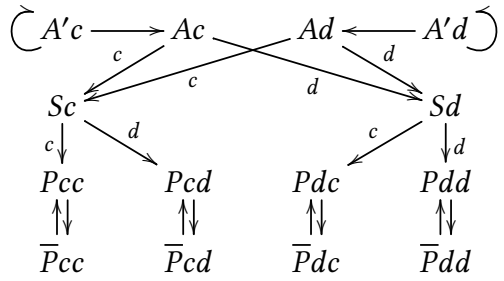
Definition 5. For a FOL$^-$ theory Γ in GNF and set D, the graph $G = \mathcal{G}_D(\Gamma)$ is given by:

1. $V_G = \textsc{At}(D) \cup \overline{\textsc{At}}(D)$, with $\overline{\textsc{At}}(D)$ given in point 3.
2. For each axiom (4) of Γ, each vertex $B_0d \in \textsc{At}(D)$ instantiating B_0x in its LS has the outgoing edges to $A_G(B_0d) = \{B_idc \mid 1 \leq i \leq n, c \in D^{arity(B_id)}\}$.
3. For each predicate symbol B without any definition in Γ, we add a fresh symbol $\overline{B}$ and the 2-cycle $Bd \leftrightarrows \overline{B}d$ for each $d \in D^{arity(B)}$. Vertices $\overline{B}d$ form $\overline{\textsc{At}}(D)$.

A nonempty D is a domain of a FOL-structure interpreting the language of the theory, and $Gr(\Gamma)$ denotes the class of graphs $\mathcal{G}_D(\Gamma)$, for all nonempty sets D.

Example 6. For $\phi = \forall x \exists y Pxy$ with $\Gamma = GNF(\phi)$ from Example 3, repeated to the left, and for the set $D = \{c, d\}$, Definition 5 yields the graph $\mathcal{G}_D(\phi)$ to the right:

$$\begin{aligned} A'z &\leftrightarrow \neg Az \wedge \neg A'z \\ Az &\leftrightarrow \forall x \neg Sx \\ Sx &\leftrightarrow \forall y \neg Pxy \end{aligned}$$



The subgraph induced by $\{Ac, Ad, Sc, Sd\}$ and all Pxy vertices corresponds to $GNF^-(\phi)$, with vertices $\overline{P}xy$ and their 2-cycles to Pxy added according to Definition 5.3. The top vertices $A'c, A'd$ with their edges arise from the final formula $GNF^+(\phi)$.

For any D and $d \in D$, each vertex Ad in $\mathcal{G}_D(\Gamma)$ has edges to $|D|$ copies of a subgraph with a source Sx and edges to Pxy, for all $x, y \in D$. Each pair Pxy, $\overline{P}xy$ forms a 2-cycle.

The graph $\mathcal{G}_D(\Gamma)$ captures (up to isomorphism) all models of a GNF theory Γ over the set D. These models stand namely in bijection to *kernels* of $\mathcal{G}_D(\Gamma)$. A *kernel* (*solution*, introduced in [5]) of a graph G is a subset $K \subseteq V_G$ which is *independent*, i.e., $A_G^-(K) \subseteq V_G \setminus K$, and *absorbs its complement*, i.e., $A_G^-(K) \supseteq V_G \setminus K$, in short, such that $A_G^-(K) = V_G \setminus K$. Equivalently, it is an assignment $\alpha \in \{1, 0\}^{V_G}$ such that $\forall x \in V_G : \alpha(x) = 1 \Leftrightarrow (\forall y \in A_G(x) : \alpha(y) = 0)$. (Vertices assigned 1 form a kernel.) $Ker(G)$ denotes the set of all kernels of a graph G.

Example 7. A model over $\{c, d\}$ of the formula $\forall x \exists y Pxy$ must satisfy either Pcc or Pcd, and either Pdd or Pdc. These determine exactly the kernels of $\mathcal{G}_{\{c,d\}}(\Gamma)$

from Example 6. For every kernel $K \cap \{A'c, A'd\} = \emptyset$, hence $\{Ac, Ad\} \subseteq K$, forcing $\{Sc, Sd\} \cap K = \emptyset$. $Sc \notin K$ requires $\{Pcc, Pcd\} \cap K \neq \emptyset$, while $Sd \notin K$ requires $\{Pdc, Pdd\} \cap K \neq \emptyset$.

Vertices of $\mathcal{G}_D(\Gamma)$ contain all D instances of atomic formulas. An assignment $v \in D^{\mathcal{V}(\phi)}$ to free variables $\mathcal{V}(\phi)$ of a formula ϕ, along with a kernel K, determine a valuation of all atoms over $\mathcal{V}(\phi)$, with atoms in K assigned 1. Satisfaction of arbitrary formulas is defined over this basis in the standartd way. Denoting $GMod(\Gamma) = \bigcup\{\{G\} \times Ker(G) \mid G \in Gr(\Gamma)\}$, we have the following correspondence (extending to full FOL).

Fact 8. *In FOL⁻:*

1. *For every GNF theory* Γ : $Mod(\Gamma) \leftrightarrows GMod(\Gamma)$.
2. *For every theory* T : $Mod(T) \leftrightarrows GMod(GNF(T))$.

Consistency of GNF theories can be thus analysed through the properties of their graphs. The notion of circularity, in particular, of a circular definition in a theory Γ is reflected precisely by circles in graphs $Gr(\Gamma)$. Acyclicity of any graph in $Gr(\phi)$, from Example 6, reflects intuitive and informal non-circularity of ϕ from Example 3. According to the classical theorem of Richardson, [3], a finite graph (or, more generally, a graph without infinite simple paths, or without infinite out-degree) has a kernel. Odd cycles appear thus as an exact representation of vicious logical circularity which, in this limited context (of finite branching or no infinite simple paths), is the only source of inconsistency.

Transforming a theory into GNF, although simple, may be worthwhile only in special cases. There is, however, a natural context where such a transformation is relatively straightforward, namely, extensions of theories with new predicates. An extension of a given theory Γ, with predicates defined by formulas Δ, can be formulated as $GNF^-(\Delta)$ extending Γ. Existence and uniquenss of corresponding model expansions, in particular, when Δ involves circularity, can be analysed using graphs of $GNF^-(\Delta)$.

Example 9. Extending the language of PA with predicate T defined by

(Δ) $\quad Tx \leftrightarrow \neg T(sx)$

yields, over the standard model $\mathbb{N}$ of PA, the acyclic graph $T0 \rightarrow T1 \rightarrow T2 \rightarrow$ ..., with two obvious kernels, applying T to all even or to all odd numbers. The occurrence of T on both sides of (Δ) does not lead to any circularity over the standard model.

Interpreting (Δ) over naturals modulo 2, yields a circular graph:

$$T[0] \rightleftarrows T[1].$$

Interpreting (Δ) over naturals modulo 3, yields a 3-cycle:

$$T[0] \longrightarrow T[1]. \longrightarrow T[2]. \text{ (back to } T[0]\text{)}$$

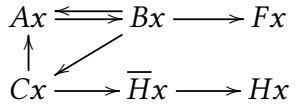
In the two first cases, the graphs have kernels by Richardson's theorem, so these models can be extended with an interpretation of (Δ). Nonexistence of such an extension in the third case follows by the trivial analysis of its graph. It is, in fact, a special case of a general fact that a finite graph without even cycles, where each vertex has an outgoing edge, has no kernel [8].

Example 10. A theory Γ with predicates F, H is extended with a predicate A given by

$$(\Delta) \quad Ax \leftrightarrow Fx \vee (Hx \wedge \neg Ax) \vee Ax.$$

$GNF^-(\Delta)$ is then the following theory, with the abbreviated graph:

$$\begin{aligned} GNF^-(\Delta) \quad & Ax \leftrightarrow \neg Bx \\ & Bx \leftrightarrow \neg Fx \wedge \neg Cx \wedge \neg Ax \\ & Cx \leftrightarrow \neg\overline{H}x \wedge \neg Ax \\ & \overline{H}x \leftrightarrow \neg Hx \end{aligned}$$



The induced subgraph $\{Ax, Bx, Cx\}$ has a kernel for each assignment to Fx and Hx. Thus, each model of any theory over F, H (not involving A) has an expansion to a model of (Δ), or else, the definition (Δ) has a fixed point over every such model. In particular, extension (Δ) is conservative.

The examples illustrate generality of the approach, which allows to analyse existence of fixed points, that is, models of GNF theories, using results from kernel theory. For instance, existence of fixed points for definitions without negative occurrences of the defined predicate becomes a special case of Richardson's theorem, for graphs satisfying its restrictions, because absence of negative occurrences in a defining formula implies absence of odd cycles in its graph. Kernel theory identifies various conditions (admitting also odd cycles, i.e., negative occurrences, as in Example 10) which can be applied in more general situations. An overview of kernel results for finite graphs can be found in [2], while few results for infinite ones in [4, 7].

[1] M. Bezem, C. Grabmayer, and M. Walicki. Expressive power of digraph solvability. *Annals of Pure and Applied Logic*, 163(3):200–212, 2012.

[2] E. Boros and V. Gurvich. Perfect graphs, kernels and cooperative games. *Discrete Mathematics*, 306:2336–2354, 2006.

[3] M. Richardson. Solutions of irreflexive relations. *The Annals of Mathematics, Second Series*, 58(3):573–590, 1953.

[4] R. Rojas-Monroy and J. I. Villarreal-Valdés. Kernels in infinite digraphs. *AKCE International Journal of Graphs and Combinatorics*, 7(1):103–111, 2010.

[5] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944 (1947).

[6] M. Walicki. Resolving infinitary paradoxes. *The Journal of Symbolic Logic*, 82(2):709–723, 2017.

[7] M. Walicki. Kernels of digraphs with finitely many ends. *Discrete Mathematics*, 342:473–486, 2019.

[8] M. Walicki and S. Dyrkolbotn. Finding kernels or solving SAT. *Journal of Discrete Algorithms*, 10:146–164, 2012.