Version: July 22, 2022

iPRA 2022

The 4th Workshop on Interpolation: From Proofs to Applications

Haifa, Israel, August 11, 2022

At FLoC 2022: Federated Logic Conference,
affiliated with IJCAR 2022, the 11th International Joint
Conference on Automated Reasoning

Book of Abstracts

Michael Benedikt · Philipp Rümmer · Christoph Wernhard (Eds.)

Editors

Michael Benedikt
University of Oxford
Oxford
UK

Philipp Rümmer
University of Regensburg
Regensburg
Germany

Christoph Wernhard
University of Potsdam
Potsdam
Germany

Preface

This volume contains the abstracts of the presentations at the 4th Workshop on Interpolation: From Proofs to Applications (iPRA 2022), held on August 11, 2022, in Haifa, Israel, at the Federated Logic Conference 2022 (FLoC 2022) and affiliated with the 11th International Joint Conference on Automated Reasoning (IJCAR 2022). It resumes – after some hiatus – the iPRA workshop series, initiated in 2013 by Laura Kovács and Georg Weissenbacher, with three previous issues, 2013 in Saint Petersburg, 2014 in Vienna, and 2015 in San Francisco.

Starting out from Craig's interpolation theorem for first-order logic, the existence and computation of interpolant formulas has many facets of theoretical and practical interest. Approaches include techniques from automated reasoning, proof theory and model theory. Notable application fields are verification, databases and knowledge representation. The program of iPRA 2022 was centered around four invited talks by leading researchers presenting recent work:

- Bahareh Afshari on *Interpolation and Completeness in the Modal Mu-Calculus*
- Alessandro Gianola on *Uniform Interpolants and Model Completions in Formal Verification of Infinite-State Systems*
- H. Jerome Keisler and Jeffrey M. Keisler on *Application of Interpolation in Networks*
- Kenneth L. McMillan on *Interpolants and Transformational Proof Systems*

These invited talks were complemented by five contributed presentations, based on the submissions for an open call for contributions.

We would like to thank all those involved for their enthusiasm and high-quality contributions, in particular, the invited speakers, the authors of submitted contributions, the members of the Program Committee, and the FLoC 2022 Workshop Chairs Shaull Almagor and Guillermo A. Pérez.

August 2022

Michael Benedikt
Philipp Rümmer
Christoph Wernhard

Organization

Program Committee Chairs and Organizers

Michael Benedikt	University of Oxford, UK
Philipp Rümmer	University of Regensburg, Germany
Christoph Wernhard	University of Potsdam, Germany

Program Committee

Maria Paola Bonacina	Università degli Studi di Verona, Italy
Silvio Ghilardi	Università degli Studi di Milano, Italy
Arie Gurfinkel	University of Waterloo, Canada
Rosalie Iemhoff	Utrecht University, The Netherlands
Laura Kovács	TU Wien, Austria
Pavel Pudlák	Czech Academy of Sciences, Czech Republic
Georg Weissenbacher	TU Wien, Austria
Frank Wolter	University of Liverpool, UK

Contents

Invited Talks

Contributed Presentations

Interpolation and Completeness in the Modal Mu-Calculus

Bahareh Afshari[1,2]

[1] Institute for Logic, Language and Computation, University of Amsterdam, Amsterdam, The Netherlands
[2] Department of Philosophy, Linguistics and Theory of Science, University of Gothenburg, Gothenburg, Sweden

From a proof-theoretic perspective, the idea that interpolation is tied to provability is a natural one. Thinking about Craig interpolation, if a 'nice' proof of a valid implication $\phi \to \psi$ is available, one may succeed in defining an interpolant by induction on the proof-tree, starting from leaves and proceeding to the implication at the root. This method has recently been applied even to fixed point logics admitting cyclic proofs [2, 6]. In contrast, for uniform interpolation, there is no single proof to work from but a collection of proofs to accommodate: a witness to each valid implication $\phi \to \psi$ where the vocabulary of ψ is constrained. Working over a set of prospective proofs and relying on the structural properties of sequent calculus is the essence of Pitts' seminal result on uniform interpolation for intuitionistic logic [5].

In this talk we will look at how Pitts' technique can be adapted to derive uniform interpolation for the propositional modal μ-calculus [3]. For this we introduce the notion of an interpolation template, a finite (cyclic) derivation tree in a sequent calculus based on the Jungteerapanich–Stirling annotated proof system [4, 7]. Uniform interpolants arise from encoding the structure of interpolation templates within the syntax of the μ-calculus. We will conclude with a somewhat surprising corollary of the interpolation method via cyclic proofs: a straightforward proof of completeness for Kozen's finitary axiomatisation of the μ-calculus [1].

References

1. Afshari B., Leigh, G.E.: From interpolation to completeness (unpublished)
2. Afshari B., Leigh, G.E.: Lyndon interpolation for modal mu-calculus. In Post- Proceedings: Özgün A., Zinova Y. (eds.) TbiLLC2019, Language, Logic, and Computation, vol. 13206, pp. 97–213, LNCS (2022)
3. Afshari B., Leigh, G.E., Turata, G.M: Uniform interpolation from cyclic proofs: The case of modal mu-calculus. In: Das A., Negri S (eds.) TABLEAUX 2021, Automated Reasoning with Analytic Tableaux and Related Methods - 30th International Conference, vol. 12842, pp. 335–353, Springer, Birmingham (2021)
4. Jungteerapanich, N.: A tableau system for the modal μ-calculus. In: Giese, M., Waaler, A. (eds.) TABLEAUX 2009. LNCS (LNAI), vol. 5607, pp. 220–234. Springer, Heidelberg (2009)
5. Pitts A.M: On an interpretation of second order quantification in first order intuitionistic propositional logic. Journal of Symbolic Logic, **57**(1), pp. 33–52 (1992)

6. Shamkanov, D.: Circular proofs for the Gödel-Löb provability logic. Math. Notes **96**, 575–585 (2014)
7. Stirling, C.: A tableau proof system with names for modal mu-calculus. In: Voronkov, A., Korovina, M.V. (eds.) HOWARD-60: A Festschrift on the Occasion of Howard Barringer's 60th Birthday, EPiC Series in Computing, vol. 42, pp. 306–318. EasyChair (2014)

Uniform Interpolants and Model Completions in Formal Verification of Infinite-State Systems

Alessandro Gianola[1]

Free University of Bozen-Bolzano, Bolzano, Italy
gianola@inf.unibz.it

1 Overview

In this paper, we report recent results on *uniform interpolation* and its connection with *model completions* [9,10], and how uniform interpolants can be used as a powerful tool in the context of *formal verification* of infinite-state systems [8,4,18,23]. These results originate from a rather surprising confluence of two well-established research fields: the one of *model-theoretic algebra* [34] in mathematical logic, where uniform interpolants and model completions were investigated for non-classical logics, and the one of *automated reasoning* and *Satisfiability Modulo Theories* (SMT) [2], where uniform interpolants provide a light form of quantifier elimination. We discuss how such apparently quite distant scientific paradigms can indeed cooperate in formal verification of infinite-state systems, in particular for applications to the so-called *data-aware processes* [6,4,17,23]: the last ones are systems where the control flow of a (business) process can interact with a data storage.

1.1 The Problem of Quantifiers in Verification

Verification of infinite-state systems requires to develop *symbolic* techniques in order to represent the transitions and the sets of states that are *reachable* through those transitions: indeed, contrary to what happens in traditional finite-state model checking [13], an explicit, exhaustive exploration of the state space is not possible, because of the presence of infinitely many states.

In the context of SMT-based model checking, several techniques have been successfully studied to attack this problem: for instance, there is a plethora of prominent methods that are based on *forward* reachability, such as K-induction [35], or on *backward* reachability [20]. Methods based on forward reachability symbolically explore the state space starting from the initial configurations, via an iterative computation of *direct images*, i.e., the set of states reachable through the transitions. In case of backward reachability methods, the final states are 'regressed' by iteratively computing their predecessors (i.e., the *pre-image*), until a fixpoint is reached or the initial state(s) are intersected. Specifically, in both cases, the main ingredient of the verification procedure is the computation of *(direct or pre-)images* of sets of states via transitions: the state space can be, in this way, explored symbolically. There exist various SMT-based model checkers implementing these methods (e.g., KIND 2 [11] or MCMT [21]).

In many declarative formalisms (e.g., [20,8]), sets of states are represented using *quantifier-free* logical formulae, called *state formulae*: the variables occurring in a state formula are those whose content characterizes the global state of the system and may change during its evolution. Starting from state formulae, the verification procedure symbolically computes *reachable* sets of states, which should be represented again as state formulae: indeed, images are intended to describe sets of states, and as such, they should be *quantifier-free*. However, reachable sets of states need to be manipulated, because in general are *not* natively described by state formulae: this is due to the fact that, when computing images, first-order (mostly, *existential*) quantifiers may be introduced by the transitions of the system, which are usually quantified formulae. For instance, this is the case of data-aware processes, when their transitions contain as guards existential queries over a relational database [6,4]. Consequently, these quantifiers appear in the computation of images, breaking the quantifier-free format of state formulae. A sort of *quantifier elimination* is then needed to represent reachable states as state formulae, so as to guarantee the regressability of the verification procedure.

Dealing with quantifiers is a genuine problem when employing verification frameworks based on SMT solving. For instance, declarative versions of backward reachability [19,20] require discharging to SMT solvers proof obligations that can be reduced to satisfiability tests for quantified formulae with a restricted syntactic shape. Although SMT solvers can natively reason only about the quantifier-free fragments of theories, first-order quantifiers can be in some cases handled by *instantiation* [20], whereas in others [1], where quantifiers range over specific data structures such as real-valued clocks involving *light versions* of arithmetic, proper quantifier elimination can be employed. In that context, the existential quantifiers to eliminate bind only arithmetic variables, and the corresponding quantifier elimination procedures are consequently the standard ones studied for arithmetic theories, such as Fourier-Motzkin and Cooper algorithms [14].

In general, the *precise* computation of the set of reachable states can be effectively performed via *proper* quantifier elimination (QE). However, quantifier elimination is not always possible in generic first-order theories, and, when available, is in many cases computationally intractable: for instance, this is the aforementioned case of arithmetical theories, for which QE is in general too expensive to be employed in practice. In order to cope with this problem, other methods for symbol elimination (e.g., predicate abstraction [25,26] or *ordinary interpolation* [29,30]) have been investigated, which do not compute precise images, but perform an *approximation* of the reachable states: this implies that images can contain 'spurious elements', i.e., states that are not properly reachable in one step. Nevertheless, these methods have been proved to be quite successful and computationally efficient. The main limitation is that they usually require refinement techniques to handle the spurious traces possibly produced by the approximation.

2 Model Completions and Uniform Interpolation

The aim of this paper is to remark how in significant cases, such as the one of data-aware processes verification, approximating methods can be abandoned in favor of *exact methods* that have both the merits of computing precise images and of remaining computationally tractable (i.e., in polytime). These exact methods are based on the use of *uniform interpolants*, which are a strong form of interpolants having strict relationships with quantifier elimination performed in richer theories called *model completions*. We clarify why model completions come to the picture and how they are related to uniform interpolation.

2.1 Model-Theoretic Algebra and Model Completions

Formally, a first-order theory T has *quantifier elimination* iff for every formula ϕ in the signature of T there is a quantifier-free formula ϕ' s.t. ϕ is T-equivalent to ϕ'. Using logical transformations, the problem of eliminating quantifiers from a generic first-order formula can be equivalently formulated as the one of eliminating *existential* quantifiers from *constraints*, i.e., conjunctions of literals.

Eliminating *existential quantifiers* from a formula has an interesting interpretation: it can be seen as the logical counterpart of finding *witnesses*, i.e., *solutions*, to suitable systems of equations and/or disequalities expressed in logical form. Model-theoretic algebra, starting from the pioneering work by Robinson [33,34], provides a powerful setting where to formulate this problem in algebraic terms, and where to solve it by exploiting tools from model theory.

The central notion is that of *existentially closed model*. A quantifier-free formula with parameters in a model M is *solvable* if there is an extension M' of M where the formula is satisfied. A model M is *existentially closed* if any solvable quantifier-free formula already has a solution in M itself. This notion is not first-order definable in general. However, in significant cases, the class of existentially closed models of T are exactly the models of another first-order theory T^*, which in the literature is abstractly characterized as the *model companion* of T. Under suitable hypothesis (one of them being T a universal theory), model companions enjoy richer properties and become the so-called *model completions* [12].

Formally, a *universal* theory T has a *model completion* iff there is a stronger theory $T^* \supseteq T$ (still within the same signature Σ of T) such that: (i) every Σ-constraint that is satisfiable in a model of T is satisfiable in a model of T^*; (ii) T^* *eliminates quantifiers*. Other equivalent definitions are possible [12]: for instance, (i) is equivalent to the fact that T and T^* prove the same universal formulae or again to the fact that every model of T can be embedded into a model of T^*. The theory T^*, if it exists, is unique. In model completions, quantifier elimination holds, even in case it does not in the original theory T. The model companion/model completion of a theory identifies the class of those models where *all satisfiable existential statements can be satisfied*.

In declarative approaches to verification of infinite-state systems, the runs of a system that are 'analyzed' by image computation are identified with certain *definable paths* in the models of a suitable theory T: for instance, in the case

of data-aware processes the theory T formalizes the relational database that is queried and modified by the transition system [8].

We already remarked that, when performing these computations, first-order quantifiers are introduced and are needed to be eliminated, even in case of theories T not admitting quantifier elimination. Nevertheless, in many significant cases where QE is not available, model completions still exist. This is the case of useful theories such as the ones used for data-aware process verification [8,4].

For these reasons, model completions become the crucial tool to exploit: without loss of generality, one can *restrict the analysis to paths within existentially closed models*, thus taking profit from the properties enjoyed by the model completion T^*, first of all quantifier elimination. For instance, indeed, it is possible to prove, in the case of safety verification, that performing backward reachability for systems whose models live in T is *equivalent* to performing backward reachability for systems whose models live in T^* [8,9]: favorably, in T^* quantifiers can be eliminated, even when in T quantifier elimination is not available.

2.2 Uniform Interpolants and QE in Model Completions

We now give a general definition of uniform interpolants and we discuss the strict relationship between uniform interpolants and model completions. Let T be a logic or a theory and let L be a suitable fragment (propositional, first-order quantifier-free, etc.) of its language. Given an L-formula $\phi(\underline{x}, \underline{y})$ (here $\underline{x}, \underline{y}$ are the variables occurring in ϕ), a *(L-)uniform interpolant (UI)* of ϕ (w.r.t. $\underline{y}$) is an L-formula $\phi'(\underline{x})$ where only the $\underline{x}$ occur, satisfying the following two properties:

(i) $\phi(\underline{x}, \underline{y}) \vdash_T \phi'(\underline{x})$;

(ii) for any further L-formula $\psi(\underline{x}, \underline{z})$ such that $\phi(\underline{x}, \underline{y}) \vdash_T \psi(\underline{x}, \underline{z})$, we have $\phi'(\underline{x}) \vdash_T \psi(\underline{x}, \underline{z})$.

Notice that, for every pair of L-formulae $\phi(\underline{x}, \underline{y})$ and $\psi(\underline{x}, \underline{z})$ such that $\phi(\underline{x}, \underline{y}) \vdash_T \psi(\underline{x}, \underline{z})$, a uniform interpolant of ϕ is in particular an *ordinary (Craig) interpolant* for the pair (ϕ, ψ) [15]. Hence, whenever uniform interpolants exist, one can compute an ordinary interpolant for an entailment such as $\phi(\underline{x}, \underline{y}) \vdash_T \psi(\underline{x}, \underline{z})$ in a way that is *independent* of ψ, i.e., *uniformly*: indeed, the same ordinary interpolant $\phi'(\underline{x})$ can be used as interpolant for all entailments $\vdash_T \phi(\underline{x}, \underline{y}) \rightarrow \psi(\underline{x}, \underline{z})$, varying ψ. For this reason, such an ordinary interpolant is called *uniform*. A theory T admits (L-)uniform interpolation iff, for every L-formula $\phi(\underline{x}, \underline{y})$, there exists a ($L$-)uniform interpolant $\phi'(\underline{x})$ of ϕ w.r.t. $\underline{y}$.

Uniform interpolants were originally investigated in non-classical logics, since the pioneering work by Pitts [32]. Uniform interpolants are stronger than ordinary interpolants: even in case Craig interpolants exist, uniform interpolants may not exist. Hence, the existence of uniform interpolants is an exceptional phenomenon, but not so rare. Since the nineties, they have been extensively studied in a large literature (e.g., [22,28,31]). Recently, the automated reasoning community has developed an increasing interest in uniform interpolants, focusing on the case L is the *quantifier-free* fragment of some first-order theory T: from now on, we restrict our attention to this case. This interest is witnessed by various talks by Kapur in various conferences and workshops (FloC 2010,

ISCAS 2013-14, SCS 2017 [27]), as well as by Gulwani and Musuvathi in [24], where uniform interpolants are called *covers*, examples of UI computations were supplied and some algorithms were sketched. The usefulness of uniform interpolants in model checking was first stressed in that work: they can be employed to compute *exact* direct images of set of states. This interest has been confirmed and further motivated by data-aware process verification [6,9].

The first formal *proof* about the *existence* of uniform interpolants in $\mathcal{EUF}$ was published in [5,9], where also the following crucial result was proved: computing uniform interpolants in T is *equivalent* to eliminating quantifiers in its model completion T^*. Hence, instead of investigating paths in T^* and eliminating quantifiers in model completions, reachability can be performed in T itself by computing uniform interpolants there. In those papers, a UI algorithm for $\mathcal{EUF}$ relying on a constrained variant of the Superposition Calculus was proposed: in the restricted case that is used for verifying safety of data-aware process, it is also possible to provide a quadratic bound in time for UI computation. Two simpler UI algorithms are studied in [16].

A close relationship between model completion and uniform interpolation was deeply studied also in other contexts, like in propositional logic (see, for instance, [22]). It is well-known that most propositional calculi, via Lindenbaum constructions, can be algebraized: for instance, the algebraic counterpart of classical logic are Boolean algebras, and the algebraic counterpart of intuitionistic logic are Heyting algebras. Under suitable hypotheses, it turns out that a propositional logic has uniform interpolation (for the global consequence relation) if and only if the equational theory axiomatizing its corresponding variety of algebras has a model completion [22].

Another important question suggested by formal verification regards the UI transfer to combined theories: for instance, for data-aware processes it is natural to consider the combination of different theories, such as the ones accounting for the different types of data contained in the persistent storage [10,6]. The UI transfer problem is: *supposing that uniform interpolants exist in theories* T_1, T_2, *under which conditions do they exist also in the combined theory* $T_1 \cup T_2$*?* In [7,10] combined uniform interpolants are shown to exist in the disjoint-signatures convex case under the same hypothesis, i.e., the *equality interpolating condition* [36], guaranteeing the transfer of quantifier-free ordinary interpolation: for convex theories, equality interpolating is not only a sufficient condition, but also necessary, in the sense that is required for the minimal combination with $\mathcal{EUF}$.

A combined UI algorithm [7,10] can be designed by extensively using *Beth definability*, which is a property that is enjoyed by equality interpolating theories [3]. Indeed, the equivalence between implicit and explicit definability that is provided by Beth definability is exploited by the combined algorithm in the following way: the algorithm guesses the implicitly definable variables, then eliminates them via explicit definability, and finally uses the component-wise input UI algorithms to eliminate the remaining (not implicitly definable) variables. The identification and the elimination of the implicitly defined variables via explicitly defining terms is crucial for the correctness of the combined UI algorithm.

References

1. Alberti, F., Bruttomesso, R., Ghilardi, S., Ranise, S., Sharygina, N.: An extension of lazy abstraction with interpolation for programs with arrays. Formal Methods in System Design **45**(1), 63–109 (2014)
2. Barrett, C.W., Tinelli, C.: Satisfiability modulo theories. In: Handbook of Model Checking., pp. 305–343. Springer (2018)
3. Bruttomesso, R., Ghilardi, S., Ranise, S.: Quantifier-free interpolation in combinations of equality interpolating theories. ACM Transactions on Computational Logic **15**(1), 5:1–5:34 (2014)
4. Calvanese, D., Ghilardi, S., Gianola, A., Montali, M., Rivkin, A.: Formal modeling and SMT-based parameterized verification of data-aware BPMN. In: Proceeding of BPM 2019. LNCS, vol. 11675, pp. 157–175. Springer (2019)
5. Calvanese, D., Ghilardi, S., Gianola, A., Montali, M., Rivkin, A.: Model completeness, covers and superposition. In: Proceedings of CADE 2019. LNCS (LNAI), vol. 11716, pp. 142–160. Springer (2019)
6. Calvanese, D., Ghilardi, S., Gianola, A., Montali, M., Rivkin, A.: Verification of data-aware processes: Challenges and opportunities for automated reasoning. In: Proceedings of ARCADE 2019. vol. 311. EPTCS (2019)
7. Calvanese, D., Ghilardi, S., Gianola, A., Montali, M., Rivkin, A.: Combined Covers and Beth Definability. In: Proceedings of IJCAR 2020. LNCS (LNAI), vol. 12166, pp. 181–200. Springer (2020)
8. Calvanese, D., Ghilardi, S., Gianola, A., Montali, M., Rivkin, A.: SMT-based verification of data-aware processes: a model-theoretic approach. Mathematical Structures in Computer Science **30**(3), 271–313 (2020)
9. Calvanese, D., Ghilardi, S., Gianola, A., Montali, M., Rivkin, A.: Model completeness, uniform interpolants and superposition calculus. Journal of Automated Reasoning **65**(7), 941–969 (2021)
10. Calvanese, D., Ghilardi, S., Gianola, A., Montali, M., Rivkin, A.: Combination of uniform interpolants via Beth definability. Journal of Automated Reasoning **66**(3) (2022)
11. Champion, A., Mebsout, A., Sticksel, C., Tinelli, C.: The Kind 2 model checker. In: Proceedings of CAV 2016. LNCS, vol. 9780, pp. 510–517. Springer (2016)
12. Chang, C.C., Keisler, J.H.: Model Theory. North-Holland Publishing Co., Amsterdam-London, third edn. (1990)
13. Clarke, E.M., Grumberg, O., Peled, D.A.: Model checking. MIT Press (2001)
14. Cooper, D.C.: Theorem proving in arithmetic without multiplication. In: Machine Intelligence. vol. 7, pp. 91–100. Edinburgh University Press (1972)
15. Craig, W.: Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory. Journal of Symbolic Logic **22**, 269–285 (1957)
16. Ghilardi, S., Gianola, A., Kapur, D.: Uniform interpolants in EUF: Algorithms using DAG-representations. Logical Methods in Computer Science **18**(2) (2022)
17. Ghilardi, S., Gianola, A., Montali, M., Rivkin, A.: Delta-BPMN: A concrete language and verifier for Data-Aware BPMN. In: Proceedings of BPM 2021. LNCS, vol. 12875, pp. 179–196. Springer (2021)
18. Ghilardi, S., Gianola, A., Montali, M., Rivkin, A.: Petri net-based object-centric processes with read-only data. Information Systems **107** (2022)
19. Ghilardi, S., Nicolini, E., Ranise, S., Zucchelli, D.: Towards SMT model checking of array-based systems. In: Proceedings of IJCAR 2008. LNCS (LNAI), vol. 5195, pp. 67–82. Springer (2008)

20. Ghilardi, S., Ranise, S.: Backward reachability of array-based systems by SMT solving: Termination and invariant synthesis. Logical Methods in Computer Science **6**(4) (2010)
21. Ghilardi, S., Ranise, S.: MCMT: A model checker modulo theories. In: Proceedings of IJCAR 2010. LNCS (LNAI), vol. 6173, pp. 22–29. Springer (2010)
22. Ghilardi, S., Zawadowski, M.: Sheaves, games, and model completions, Trends in Logic—Studia Logica Library, vol. 14. Kluwer Academic Publishers, Dordrecht (2002)
23. Gianola, A.: SMT-based Safety Verification of Data-Aware Processes: Foundations and Applications. Ph.D. thesis, Free University of Bozen-Bolzano (2022)
24. Gulwani, S., Musuvathi, M.: Cover algorithms and their combination. In: Proceedings of ESOP 2008. LNCS, vol. 4960, pp. 193–207. Springer (2008)
25. Jhala, R., Majumdar, R.: Software model checking. ACM Comput. Surv. **41**(4), 21:1–21:54 (2009)
26. Jhala, R., Podelski, A., Rybalchenko, A.: Predicate abstraction for program verification. In: Handbook of Model Checking, pp. 447–491. Springer (2018)
27. Kapur, D.: Nonlinear polynomials, interpolants and invariant generation for system analysis. In: Proceedings of SC-Square 2017 (co-located with ISSAC). vol. 1974. CEUR Workshop Proceedings (2017)
28. Kowalski, T., Metcalfe, G.: Uniform interpolation and coherence. Annals of Pure and Applied Logic **170**(7), 825–841 (2019)
29. McMillan, K.L.: Interpolation and SAT-Based Model Checking. In: Proceedings of CAV 2003. LNCS, vol. 2725, pp. 1–13. Springer (2003)
30. McMillan, K.L.: Lazy Abstraction with Interpolants. In: Proceedings of CAV 2006. LNCS, vol. 4144, pp. 123–136. Springer (2006)
31. Metcalfe, G., Reggio, L.: Model completions for universal classes of algebras: necessary and sufficient conditions. Journal of Symbolic Logic pp. 1–34 (2022)
32. Pitts, A.M.: On an interpretation of second order quantification in first order intuitionistic propositional logic. Journal of Symbolic Logic **57**(1), 33–52 (1992)
33. Robinson, A.: On the metamathematics of algebra. Studies in Logic and the Foundations of Mathematics, North-Holland Publishing Co., Amsterdam (1951)
34. Robinson, A.: Introduction to model theory and to the metamathematics of algebra. Studies in logic and the foundations of mathematics, North-Holland (1963)
35. Sheeran, M., Singh, S., Stålmarck, G.: Checking safety properties using induction and a SAT-solver. In: Proceedings of FMCAD 2000. LNCS, vol. 1954, pp. 108–125. Springer (2000)
36. Yorsh, G., Musuvathi, M.: A combination method for generating interpolants. In: Proceedings of CADE 2005. LNCS, vol. 3632, pp. 353–368. Springer (2005)

Application of Interpolation in Networks

H. Jerome Keisler[1] and Jeffrey M. Keisler[2]

[1] University of Wisconsin, Madison, WI, U.S.A.
[2] University of Massachusetts, Boston, MA, U.S.A.

This lecture is about networks consisting of a directed graph in which each vertex is equipped with a language with the Craig interpolation property, along with a knowledge base (set of sentences) in that language. The knowledge bases grow over time according to some set of rules including the following: For any edge between two vertices, any sentence in the first knowledge base that is in the common language can be added to the second knowledge base (intuitively, the first vertex sends a message to the second). We survey a variety of results and problems that arise in this framework. A central question is: Under which conditions will a sentence be eventually provable from the knowledge base at a given vertex.

Literature

1. Keisler, H.J., Keisler, J.M.: Craig interpolation for networks of sentences. Annals of Pure and Applied Logic **163**(9), 1322–1344 (2012)
2. Keisler, H.J., Keisler, J.M.: Observing, reporting, and deciding in networks of sentences. Annals of Pure and Applied Logic **165**(3), 812–836 (2014)

Interpolants and Transformational Proof Systems

Kenneth L. McMillan

The University of Texas at Austin, Austin, TX, U.S.A.

Proof-based interpolation can be thought of as a proof transformation that localizes a proof by introducing a cut or lemma. The hope is that such a lemma will be generally useful, for example in synthesizing an inductive proof. Usually, we think of interpolants as a service provided by an automated prover to some higher-level reasoner such as a program verifier. In this talk, however, we will consider interpolation as a proof transformation to be applied during proof search. To motivate this idea, we will first consider CDCL as a transformational proof system, with conflict clauses generated by an interpolating transformation rule. Then we move from ground to quantified clauses. By adding one proof search rule and one interpolating transformation rule, we obtain a stratified CHC solver. Another transformation allows us to obtain more general conflict clauses using interpolation. The proof transformation view allows us to tightly integrate higher-level proof strategies with CDCL. This presents engineering challenges, but has the potential to produce a class of efficient solvers that can exploit the structure of problem instances.

First-Order Interpolation Derived from Propositional Interpolation*

Matthias Baaz[1] and Anela Lolic[2]

[1] Institute of Discrete Mathematics and Geometry, TU Wien
baaz@logic.at
[2] Institute of Logic and Computation, TU Wien
anela@logic.at

Ever since Craig's seminal paper on interpolation [3], interpolation properties have been recognized as important properties of logical systems. Recall that a logic L has *interpolation* if whenever $A \to B$ holds in L there exists a formula I in the common language of A and B such that $A \to I$ and $I \to B$ hold in L.

Propositional interpolation properties can be determined and classified with relative ease using the ground-breaking results of Maksimova cf. [6, 5, 4]. This approach is based on an algebraic analysis of the logic in question. In contrast first-order interpolation properties are notoriously hard to determine, even for logics where propositional interpolation is more or less obvious. For example it is unknown whether $\mathrm{G}^{\mathrm{QF}}_{[0,1]}$ (first-order infinitely-valued Gödel logic) interpolates (cf. [1]) and even for $\mathrm{MC}^{\mathrm{QF}}$, the logic of constant domain Kripke frames of three worlds with two top worlds (an extension of MC), interpolation proofs are very hard cf. Ono [8]. This situation is due to the lack of an adequate algebraization of non-classical first-order logics.

In this paper we present a proof theoretic methodology to reduce first-order interpolation to propositional interpolation:

$$\left.\begin{array}{r}\text{existence of suitable skolemizations} + \\ \text{existence of Herbrand expansions} + \\ \text{propositional interpolation}\end{array}\right\} \Rightarrow \begin{array}{c}\text{first-order} \\ \text{interpolation.}\end{array}$$

The construction of the first-order interpolant from the propositional interpolant follows this procedure:

1. Develop a validity equivalent skolemization replacing all strong quantifiers[3] in the valid formula $A \to B$ to obtain the valid formula $A_1 \to B_1$.
2. Construct a valid Herbrand expansion $A_2 \to B_2$ for $A_1 \to B_1$. Occurrences of $\exists x B(x)$ and $\forall x A(x)$ are replaced by suitable finite disjunctions $\bigvee B(t_i)$ and conjunctions $\bigwedge B(t_i)$, respectively.
3. Interpolate the propositionally valid formula $A_2 \to B_2$ with the propositional interpolant I^*:
$$A_2 \to I^* \quad \text{and} \quad I^* \to B_2$$
are propositionally valid.

* This abstract is based on the publication [2].

[3] Here we are dealing with quantifiers $\forall$ and $\exists$ such that $A(t) \to \exists x A(x)$ and $\forall x A(x) \to A(t)$ hold. This occurrence of quantifiers is called weak, the dual occurrence is called strong.

4. Reintroduce weak quantifiers to obtain valid formulas

$$A_1 \to I^* \quad \text{and} \quad I^* \to B_1.$$

5. Eliminate all function symbols and constants not in the common language of A_1 and B_1 by introducing suitable quantifiers in I^* (note that no Skolem functions are in the common language, therefore they are eliminated). Let I be the result.
6. I is an interpolant for $A_1 \to B_1$. $A_1 \to I$ and $I \to B_1$ are skolemizations of $A \to I$ and $I \to B$. Therefore I is an interpolant of $A \to B$.

We apply this methodology to lattice based finitely-valued logics and the weak quantifier and subprenex fragments of infinitely-valued first-order Gödel logic.

Note that finitely-valued first-order logics admit variants of Maehara's Lemma and therefore interpolate if all truth values are quantifier free definable [7]. For logics where not all truth-values are represented by quantifier-free formulas this argument does not hold, which explains the necessity of different interpolation arguments for e.g. $\mathrm{MC}^{\mathrm{QF}}$ (the result for $\mathrm{MC}^{\mathrm{QF}}$ is covered by our framework, cf. Example 1). We provide a decision algorithm for the interpolation property for lattice based finitely-valued logics.

Most results in interpolation are concerned with the question whether a given logic interpolates but not with the more general question, to check the minimal extensions with that property. Our framework allows for the calculation of the relevant first-order extensions, which is given by the calculation of the relevant propositional extensions. For classical logic we show in this way that the fragment with $\top, \wedge, \vee, \forall, \exists, \to$ interpolates.

Example 1. Finite propositional and constant-domain Kripke frames can be understood as lattice-based finitely valued logics: Consider upwards closed subsets $\Gamma \subseteq W$, W is the set of worlds, and order them by inclusion. A formula A is assigned the truth value Γ iff A is true at exactly the worlds in Γ. The constant-domain intuitionistic Kripke frame $\mathcal{K}$ in Fig. 1 is represented by the lattice

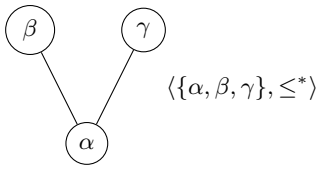

Fig. 1. Constant-domain intuitionistic Kripke frame $\mathcal{K}$.

$\mathcal{L}^{\to}(\{\binom{1}{1}\binom{1}{1}, \binom{1}{0}\binom{1}{1}, \binom{0}{0}\binom{1}{1}, \binom{1}{0}\binom{0}{0}, \binom{0}{0}\binom{0}{0}\}, \vee, \wedge, \to, \overline{\binom{0}{0}\binom{0}{0}})$ in Fig. 2. where

$$u \to v = \begin{cases} 1 & u \leq v \\ v & \text{else} \end{cases}$$

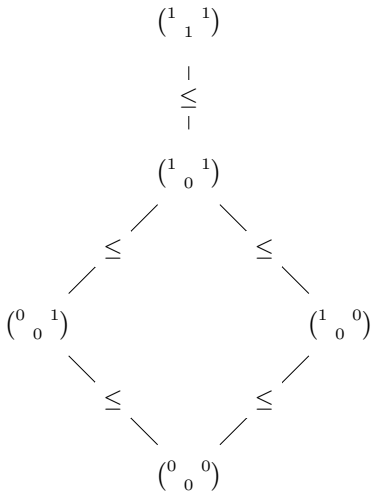

Fig. 2. The lattice.

$\mathrm{MC} = \mathbf{L}^0(\mathcal{L}^{\rightarrow})$ is the set of valid propositional sentences.

Propositional interpolation is easily demonstrated for MC, one of the seven intermediate logics which admit propositional interpolation [5]. Previous proofs for the interpolation of $\mathrm{MC}^{\mathrm{QF}}$, the first-order variant of MC, are quite involved, [8]. This interpolation result is a corollary of the main statement of this approach.

References

1. Aguilera, J.P., Baaz, M.: Ten problems in Gödel logic. Soft Computing **21**(1), 149–152 (2017)
2. Baaz, M., Lolic, A.: First-order interpolation derived from propositional interpolation. Theor. Comput. Sci. **837**, 209–222 (2020). https://doi.org/10.1016/j.tcs.2020.07.043, `https://doi.org/10.1016/j.tcs.2020.07.043`
3. Craig, W.: Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory. The Journal of Symbolic Logic **22**(03), 269–285 (1957)
4. Maksimova, L.: Intuitionistic logic and implicit definability. Annals of Pure and Applied Logic **105**(1-3), 83–102 (2000)
5. Maksimova, L.L.: Craig's theorem in superintuitionistic logics and amalgamable varieties of pseudo-Boolean algebras. Algebra and Logic **16**(6), 427–455 (1977)
6. Maksimova, L.L.: Interpolation properties of superintuitionistic logics. Studia Logica **38**(4), 419–428 (1979)
7. Miyama, T.: The interpolation theorem and Beth's theorem in many-valued logics. Mathematica Japonica **19**, 341–355 (1974)
8. Ono, H.: Model extension theorem and Craig's interpolation theorem for intermediate predicate logics. Reports on Mathematical Logic **15**, 41–58 (1983)

Interpolation and SAT-Based Model Checking Revisited: Adoption to Software Verification

Dirk Beyer, Nian-Ze Lee, and Philipp Wendler

LMU Munich, Germany

Interpolation-based model checking (McMillan, 2003) [3] is a formal-verification algorithm, which was originally devised to verify safety properties of finite-state transition systems. The algorithm is state-of-the-art in hardware model checking. It derives interpolants from unsatisfiable BMC queries, and collects them to construct an overapproximation of the set of reachable states. Unlike other formal-verification algorithms, such as k-induction or PDR, which have been extended to handle infinite-state systems and investigated for program analysis, McMillan's *interpolation-based model checking* algorithm from 2003 has not been used to verify programs. This work closes this significant, 19 years old gap in knowledge by adopting the algorithm to software verification. We implemented it in the framework CPAchecker, and evaluated the implementation against other state-of-the-art software-verification techniques over the largest publicly available benchmark suite of C safety-verification tasks. The evaluation demonstrates that interpolation-based model checking is competitive among other algorithms in terms of both the number of solved verification tasks and the run-time efficiency. Our results might have important implications for software verification, because researchers and developers now have a richer set of approaches to choose from.

The full paper is available on https://www.sosy-lab.org/research/cpa-imc and on arXiv [1].

Keywords: Software verification · Program analysis · Model checking · Interpolation · CPAchecker · SMT · SAT

Data-Availability Statement

To enhance the verifiability and transparency of the reported results, all used software, input programs, and raw experimental results are available in a supplemental research artifact [2]. For convenient browsing through the results, interactive tables are available at https://www.sosy-lab.org/research/cpa-imc. Current versions of CPAchecker are also available at https://cpachecker.sosy-lab.org.

References

1. Beyer, D., Lee, N.Z., Wendler, P.: Interpolation and SAT-based model checking revisited: Adoption to software verification. arXiV (2022)
2. Beyer, D., Lee, N.Z., Wendler, P.: Reproduction package for article 'Interpolation and SAT-based model checking revisited'. Zenodo (2022). https://doi.org/10.5281/zenodo.6700515
3. McMillan, K.L.: Interpolation and SAT-based model checking. In: Proc. CAV. pp. 1–13. LNCS 2725, Springer (2003). https://doi.org/10.1007/978-3-540-45069-6_1

Interpolation via Finitely Subdirectly Irreducible Algebras*

Wesley Fussner and George Metcalfe

Mathematical Institute, University of Bern, Switzerland
{wesley.fussner,george.metcalfe@unibe.ch}

Abstract. We present a number of algebraic results that facilitate the study of the deductive interpolation property and closely-related metalogical properties for algebraizable deductive systems. We prove that, when $\mathcal{V}$ is a congruence-distributive variety, $\mathcal{V}$ has the congruence extension property if and only if the class $\mathcal{V}_{\text{FSI}}$ of finitely subdirectly irreducible members of $\mathcal{V}$ has the congruence extension property. When a deductive system $\vdash$ is algebraized by $\mathcal{V}$, this provides a description of local deduction theorems in terms of algebraic models. Further, we prove that for a variety $\mathcal{V}$ with the congruence extension property such that $\mathcal{V}_{\text{FSI}}$ is closed under subalgebras, $\mathcal{V}$ has a one-sided amalgamation property (equivalently, since $\mathcal{V}$ is a variety, the amalgamation property) if and only if $\mathcal{V}_{\text{FSI}}$ has this property. When $\mathcal{V}$ is the algebraic counterpart of a deductive system $\vdash$, this yields a characterization of the deductive interpolation property in terms of algebraic semantics. We announce a similar result for the transferable injections property, and prove that possession of all these properties is decidable for finitely generated varieties satisfying certain conditions. Finally, as a case study, we describe the subvarieties of a notable variety of BL-algebras that have the amalgamation property.

Keywords: deductive interpolation property · amalgamation property · local deduction theorems · congruence extension property · finitely subdirectly irreducible.

This study develops an array of algebraic tools for investigating the deductive interpolation property, as well as several other closely-related metalogical properties. Our inquiry is rooted in the 'bridge theorems' of algebraic logic, which provide a means of toggling between metalogical properties of an algebraizable deductive system $\vdash$ (in the sense of [5]) and properties of the variety $\mathcal{V}$ of algebraic models corresponding to $\vdash$. When this variety $\mathcal{V}$ is tractable, such algebra-to-logic correspondences provide a powerful technique for obtaining interpolation theorems and similar results. Our main contribution consists of a number of transfer theorems that lift algebraic properties linked to deductive interpolation

* This research was supported by the Swiss National Science Foundation grant 200021_184693.

to a variety $\mathcal{V}$ from its subclass of finitely subdirectly irreducible members $\mathcal{V}_{\text{FSI}}$.[1] This gives a potent set of strategies for establishing these algebraic properties for varieties algebraizing deduction systems, for which the class of finitely subdirectly irreducibles is frequently well-behaved.[2] Further, our results yield that, under appropriate technical hypotheses, there is an effective procedure for deciding whether a deductive system $\vdash$ has the deductive interpolation property. Similar decidability results hold for the other metalogical properties we consider.

In order to express our contributions in more detail, we recall several definitions. A deductive system $\vdash$ has *the deductive interpolation property* if for all formulas φ, ψ such that $\varphi \vdash \psi$, there exists a formula σ such that

1. $\varphi \vdash \sigma$ and $\sigma \vdash \psi$, and
2. the variables of σ are among those appearing in both of φ and ψ.

Although this form of interpolation is given without regard to the language of $\vdash$, when an implication-like connection $\to$ is present in the language, the deductive interpolation property is strongly connected to the usual Craig interpolation property (see, e.g., [9]). This link is especially strong when $\vdash$ also has a *local deduction theorem*, i.e., when there exists a family $\{d_j(p,q) : j \in J\}$ of sets of formulas $d_j(p,q)$ in at most two variables such that for every set of formula $\Gamma \cup \{\varphi, \psi\}$ we have

$$\Gamma, \varphi \vdash \psi \iff \Gamma \vdash d_j(\varphi, \psi) \text{ for some } j \in J.$$

A familiar presentation of the local deduction theorem is obtained in the presence of an implication connective $\to$ and when $d_j(p,q) = \{p \to q\}$.

Both of the aforementioned properties have algebraic counterparts when $\vdash$ is algebraized by some variety $\mathcal{V}$. We say that an algebra $\mathbf{B}$ has the *congruence extension property* (or CEP for short) if for every subalgebra $\mathbf{A}$ of $\mathbf{B}$ and every congruence Θ of $\mathbf{A}$, there exists a congruence Ψ of $\mathbf{B}$ such that $\Psi \cap A^2 = \Theta$. When $\mathcal{K}$ is a class of similar algebras, we say that $\mathcal{K}$ has the CEP when each $\mathbf{A} \in \mathcal{K}$ does. Further, we say that a class $\mathcal{K}$ of similar algebras has the *amalgamation property* (or AP for short) if whenever $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathcal{K}$ and $\varphi_B \colon \mathbf{A} \to \mathbf{B}$, $\varphi_C \colon \mathbf{A} \to \mathbf{C}$ are embeddings, there exists $\mathbf{D} \in \mathcal{K}$ and embeddings $\psi_B \colon \mathbf{B} \to \mathbf{D}$, $\psi_C \colon \mathbf{C} \to \mathbf{D}$ such that $\psi_B \varphi_B = \psi_C \varphi_C$. The following well-known result provides a link between the concepts we have just introduced.

[1] Recall that an algebra $\mathbf{A}$ is *finitely subdirectly irreducible* if the least element of the congruence lattice of $\mathbf{A}$ is meet-irreducible, or, equivalently, if whenever $\mathbf{A}$ is isomorphic to a subdirect product of a finite, non-empty set of algebras, it is isomorphic to one of these algebras.

[2] Notably, if $\mathcal{V}$ has equationally definable principal meets, as is often the case for varieties algebraizing deductive systems, then $\mathcal{V}_{\text{FSI}}$ is a universal class [3, Theorem 1.5]. More concretely, if the members of $\mathcal{V}$ are semilinear residuated lattices, then $\mathcal{V}_{\text{FSI}}$ is exactly the class of totally ordered members of $\mathcal{V}$ [6].

Theorem 1. *Let $\vdash$ be a deductive system that is algebraized by a variety $\mathcal{V}$. Then:*

1. *([4]) The deductive system $\vdash$ has a local deduction theorem if and only if $\mathcal{V}$ has the congruence extension property.*
2. *([7]) When $\vdash$ has a local deduction theorem, $\vdash$ has the deductive interpolation property if and only if $\mathcal{V}$ has the amalgamation property.*

The following gives our first result. This theorem generalizes [8, Theorem 3.3], which announces a similar result dealing with subdirectly irreducibles (rather than finitely subdirectly irreducibles) under the additional hypothesis that the class of subdirectly irreducible algebras in $\mathcal{V}$ forms an elementary class.

Theorem 2. *Let $\mathcal{V}$ be a congruence-distributive variety. Then $\mathcal{V}$ has the* CEP *if and only if $\mathcal{V}_{FSI}$ has the* CEP.

Many of our results are best summarized by referencing commutative diagrams. If $\mathcal{K}$ is a class of similar algebras, we say that a *span* in $\mathcal{K}$ is a 5-tuple $\langle \mathbf{A}, \mathbf{B}, \mathbf{C}, \varphi_B, \varphi_C \rangle$ such that $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathcal{K}$ and $\varphi_B \colon \mathbf{A} \to \mathbf{B}$, $\varphi_C \colon \mathbf{A} \to \mathbf{C}$ are homomorphisms. We say that a span in $\mathcal{K}$ is *injective* if φ_B is an embedding, *doubly injective* if both φ_B and φ_C are embeddings, and *injective-surjective* if φ_B is an embedding and φ_C is surjective. The class $\mathcal{K}$ is said to have the *extension property* (or EP) if for any injective-surjective span $\langle \mathbf{A}, \mathbf{B}, \mathbf{C}, \varphi_B, \varphi_C \rangle$ in $\mathcal{K}$, there exist an algebra $\mathbf{D} \in \mathcal{K}$, a homomorphism $\psi_B \colon \mathbf{B} \to \mathbf{D}$, and an embedding $\psi_C \colon \mathbf{C} \to \mathbf{D}$ such that $\psi_B \varphi_B = \psi_C \varphi_C$ (i.e., the diagram in Figure 1(i) commutes). A variety $\mathcal{V}$ has the EP if and only if $\mathcal{V}$ has the CEP (see [2]), but this does not hold for arbitrary classes of algebras. However, under the additional hypothesis that $\mathcal{V}_{\text{FSI}}$ is closed under subalgebras, we may give the following stronger variation of Theorem 2.

Theorem 3. *Let $\mathcal{V}$ be a congruence-distributive variety such that $\mathcal{V}_{FSI}$ is closed under subalgebras. The following are equivalent:*

1. *$\mathcal{V}$ has the congruence extension property.*
2. *$\mathcal{V}$ has the extension property.*
3. *$\mathcal{V}_{FSI}$ has the congruence extension property.*
4. *$\mathcal{V}_{FSI}$ has the extension property.*

Now suppose that $\mathcal{K}$ and $\mathcal{K}'$ are two classes of algebras in a common signature and that $\langle \mathbf{A}, \mathbf{B}, \mathbf{C}, \varphi_B, \varphi_C \rangle$ is a doubly injective span in $\mathcal{K}$. An *amalgam in $\mathcal{K}'$* of this span is a triple $\langle \mathbf{D}, \psi_B, \psi_C \rangle$ where $\mathbf{D} \in \mathcal{K}'$ and $\psi_B \colon \mathbf{B} \to \mathbf{D}$ and $\psi_C \colon \mathbf{C} \to \mathbf{D}$ are embeddings such that $\psi_B \varphi_B = \psi_C \varphi_C$ (i.e., the diagram in Figure 1(ii) commutes). The class $\mathcal{K}$ has the amalgamation property, as previously defined, precisely when every doubly injective span in $\mathcal{K}$ has an amalgam in $\mathcal{K}$. Generalizing this notion, a class $\mathcal{K}$ is said to have the *one-sided amalgamation property* (or 1AP) if for any doubly injective span $\langle \mathbf{A}, \mathbf{B}, \mathbf{C}, \varphi_B, \varphi_C \rangle$ in $\mathcal{K}$, there exist a $\mathbf{D} \in \mathcal{K}$, a homomorphism $\psi_B \colon \mathbf{B} \to \mathbf{D}$, and an embedding $\psi_C \colon \mathbf{C} \to \mathbf{D}$ such that $\psi_B \varphi_B = \psi_C \varphi_C$ (i.e., the diagram in Figure 1(iii) commutes). The AP

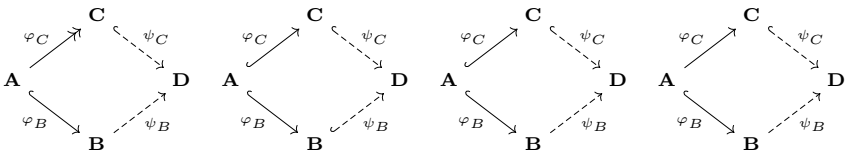

Fig. 1. Commutative diagrams for algebraic properties

and 1AP coincide when $\mathcal{K}$ is a variety, but this does not hold for arbitrary classes of algebras. Our second result is given as follows, and generalizes [12, Theorem 9].

Theorem 4. *Let $\mathcal{V}$ be a variety with the congruence extension property such that $\mathcal{V}_{FSI}$ is closed under subalgebras. The following are equivalent:*

1. *$\mathcal{V}$ has the amalgamation property.*
2. *$\mathcal{V}$ has the one-sided amalgamation property.*
3. *$\mathcal{V}_{FSI}$ has the one-sided amalgamation property.*
4. *Every doubly injective span of finitely generated algebras from $\mathcal{V}_{FSI}$ has an amalgam in $\mathcal{V}_{FSI} \times \mathcal{V}_{FSI}$.*
5. *Every doubly injective span of finitely generated algebras from $\mathcal{V}_{FSI}$ has an amalgam in $\mathcal{V}$.*

A class $\mathcal{K}$ of similar algebras is said to have the *transferable injections property* (or TIP for short) if for any injective span $\langle \mathbf{A}, \mathbf{B}, \mathbf{C}, \varphi_B, \varphi_C \rangle$ in $\mathcal{K}$, there exist an algebra $\mathbf{D} \in \mathcal{K}$, a homomorphism $\psi_B \colon \mathbf{B} \to \mathbf{D}$, and an embedding $\psi_C \colon \mathbf{C} \to \mathbf{D}$ such that $\psi_B \varphi_B = \psi_C \varphi_C$ (i.e., the diagram in Figure 1(iv) commutes). From [2, Lemma 1.7], a variety has the TIP if and only if it has both the CEP and AP. Building on the previously-announced results, we obtain the following similar theorem for the TIP.

Theorem 5. *Let $\mathcal{V}$ be a congruence-distributive variety such that $\mathcal{V}_{FSI}$ is closed under subalgebras. Then $\mathcal{V}$ has the transferable injections property if and only if $\mathcal{V}_{FSI}$ has the transferable injections property.*

Under appropriate hypotheses, the transfer theorems previously articulated can be used in conjunction with Jónsson's Lemma [11] to obtain decidability results for the algebraic properties we have discussed. In particular, we obtain:

Theorem 6. *Let $\mathcal{V}$ be a finitely generated congruence-distributive variety such that $\mathcal{V}_{FSI}$ is closed under subalgebras. Then there exist effective algorithms to decide if $\mathcal{V}$ has the congruence extension property, amalgamation property, or transferable injections property.*

Although Theorems 2-6 are phrased in universal-algebraic terms, we stress that, by Theorem 1, each of these results may be read as a semantic description

of the deductive interpolation property, local deduction theorem, or cognate metalogical property in the context of an algebraizable deductive system $\vdash$.

Finally, to illustrate the practical utility of our results, we conclude our discussion with a case study concerning several axiomatic extensions of Hájek's basic fuzzy logic [10]. These logical systems are algebraized by certain subvarieties of BL-algebras, which comprise classes of semilinear residuated lattices. In each of these varieties, the class of finitely subdirectly irreducible members is composed of linearly ordered BL-algebras, giving a class that is highly amenable to the methods we have articulated here. Using our previously-described results, we classify which of the subvarieties of BL-algebras under consideration enjoy the amalgamation property, consequently classifying which of the corresponding logical systems has the deductive interpolation property. This application provides fast and easy answers to questions of amalgamation and interpolation that are presently being actively considered in the literature (see [1]).

References

1. Aguzzoli, S., Bianchi, M.: Amalgamation property for varieties of BL-algebras generated by one chain with finitely many components. In: Proceedings of RAMiCS 2021. LNCS, vol. 13027, pp. 1–18. Springer (2021)
2. Bacsich, P.D.: Injectivity in model theory. Colloq. Math. **25**, 165–176 (1972)
3. Blok, W., Pigozzi, D.: A finite basis theorem for quasivarieties. Algebra Universalis **22**(1), 1–13 (1986)
4. Blok, W., Pigozzi, D.: Local deduction theorems in algebraic logic. In: Andréka, H., Monk, J., Nemeti, I. (eds.) Algebraic Logic, Colloquia Mathematica Societatis János Bolyai 54. pp. 75–109. Budapest, Hungary (1988)
5. Blok, W., Pigozzi, D.: Algebraizable Logics. No. 396 in Memoirs of the American Mathematical Society Volume 77, American Mathematical Society (1989)
6. Blount, K., Tsinakis, C.: The structure of residuated lattices. Internat. J. Algebra Comput. **13**(4), 437–461 (2003)
7. Czelakowski, J., Pigozzi, D.: Amalgamation and interpolation in abstract algebraic logic. In: Caicedo, X., Montenegro, C.H. (eds.) Models, Algebras, and Proofs, Lecture Notes in Pure and Applied Mathematics, vol. 203, pp. 187–265. Marcel Dekker, Inc. (1999)
8. Davey, B.: Weak injectivity and congruence extension in congruence-distributive equational classes. Canad. J. Math. **29**(3), 449–459 (1977)
9. Galatos, N., Ono, H.: Algebraization, parametrized local deduction theorem and interpolation for substructural logics over FL. Studia Logica **83**, 279–308
10. Hájek, P.: Metamathematics of Fuzzy Logic. Kluwer (1998)
11. Jónsson, B.: Algebras whose congruence lattices are distributive. Math. Scand. **21**, 110–121 (1968) (1967)
12. Metcalfe, G., Montagna, F., Tsinakis, C.: Amalgamation and interpolation in ordered algebras. J. Algebra **402**, 21–82 (2014)

When Iota Meets Lambda*

Andrzej Indrzejczak[1] and Michał Zawidzki ✉[2,1]

[1] Department of Logic, University of Lodz, 3/5 Lindleya St., 90-131 Łódź, Poland
[2] Department of Computer Science, University of Oxford, Oxford OX1 3QD, UK
andrzej.indrzejczak@filhist.uni.lodz.pl michal.zawidzki@cs.ox.ac.uk

1 Introduction

Definite descriptions (DD) are important information-conveying ingredients of every discourse, which have the form 'The so-and-so'. DD are widely discussed in linguistics and formal semantics, but their formal treatment in logic is surprisingly modest (see, e.g., Indrzejczak and Zawidzki [5] and the references therein). Although a variety of theories of DD are present on the market, usually a reductionist perspective by Russell is taken as the most prominent [6,8]. In contrast to Frege's view, he treated descriptions as a kind of incomplete signs and showed how to get rid of them by means of contextual definitions of the form:

$$\psi(\iota x\varphi(x)) \;:=\; \exists x(\forall y(\varphi(y) \leftrightarrow y = x) \wedge \psi(x)) \qquad (R)$$

However Russell's account has serious drawbacks. As it stands, (R) must be restricted to atomic ψ or it is necessary to add means for marking scope distinctions, as otherwise we can run into contradictions (see, e.g., Indrzejczak [4]).

It seems that we can avoid the problems triggered by the Russellian approach if we enrich the language with the lambda-operator and restrict the predication to descriptions to predicate abstracts of the form $\lambda x\varphi$, where φ is any formula. DD are used only as arguments of predicate abstracts, the modified version of (R) is (R_λ):

$$(\lambda x\psi)(\iota y\varphi) \leftrightarrow \exists x(\forall y(\varphi \leftrightarrow y = x) \wedge \psi), \qquad (R_\lambda)$$

where φ does not contain free occurrences of x. In this way we avoid problems with scope and inconsistency described above.

We provide a sound, complete, and cut-free tableau calculus TC_{R_λ} for the logic L_{R_λ} being a formalisation of a Russell-style theory of DD with the iota-operator used to construct DD, the lambda-operator forming predicate abstracts, and DD as genuine terms with a restricted right of residence. For this calculus we obtain a constructive proof of the Craig interpolation theorem and the Beth definability theorem as its usual consequence. This allows us to check whether an individual constant from a signature has a DD-counterpart under a given theory and can be replaced therewith.

* Both authors are supported by the National Science Centre, Poland (grant number: DEC-2017/25/B/HS1/01268). The second author is supported by the EPSRC projects OASIS (EP/S032347/1), AnaLOG (EP/P025943/1), and UK FIRES (EP/S019111/1), the SIRIUS Centre for Scalable Data Access, and Samsung Research UK.

2 Preliminaries

Syntax We consider formulas built in the standard first-order language with identity $\mathcal{L}$, where $\neg$, $\wedge$, and $\forall$ are taken as primitive expressions, augmented with the definite description operator $\imath$ and the abstraction operator λ.

A *signature* Σ for the language $\mathcal{L}$ is a triple $(\mathsf{PRED}, \mathsf{CONS}, \mathsf{ar})$, where PRED is a (possibly empty) set of predicates, CONS is a (possibly empty) set of individual constants, and $\mathsf{ar} : \mathsf{PRED} \longrightarrow \mathbb{N}_+$ is a function assigning a (positive) arity to each predicate.

Given a signature $\Sigma = (\mathsf{PRED}, \mathsf{CONS}, \mathsf{ar})$, a set of terms TERM and a set of formulas FOR over Σ (in the language of deduction) are defined simultaneously by the following context-free grammars:

$$\mathsf{TERM} \ni t ::= x \,|\, a \,|\, c \,|\, \imath x\varphi,$$
$$\mathsf{FOR} \ni \varphi ::= P(s_1, \ldots, s_n) \,|\, s_1 = s_2 \,|\, \neg\varphi \,|\, \varphi \wedge \varphi \,|\, \forall x\varphi \,|\, (\lambda x\psi)(t),$$

where $x \in \mathsf{VAR}$, $a \in \mathsf{PAR}$, $c \in \mathsf{CONS}$, $\varphi \in \mathsf{FOR}$, $s_1, \ldots, s_n \in \mathsf{VAR} \cup \mathsf{PAR} \cup \mathsf{CONS}$, $t \in \mathsf{TERM}$, and $P \in \mathsf{PRED}$ with $\mathsf{ar}(P) = n$. Henceforth, we will refer to the set $\mathsf{VAR} \cup \mathsf{PAR} \cup \mathsf{CONS}$ as TERM^-. $\varphi[s]$ indicates that s occurs freely in φ. $\varphi[s_1/s_2]$ is the result of a uniform substitution of s_1 with s_2 in φ, whereas $\varphi[s_1//s_2]$ is a result of replacing some occurrences of s_1 in φ with s_2. Note that this notation is restricted to $s_1, s_2 \in \mathsf{TERM}^-$ so we can make substitutions and replacements only using variables, parameters, or individual constants, but not DD.

Semantics Given a signature $\Sigma = (\mathsf{PRED}, \mathsf{CONS}, \mathsf{ar})$, a *model over* Σ is a structure $\mathcal{M} = (\mathcal{D}, \mathcal{I})$, where $\mathcal{D}$ is called a *domain* and $\mathcal{I}$ is an *interpretation*. For each predicate $P \in \mathsf{PRED}$, $\mathcal{I}(P) \subseteq \mathcal{D}^n$, where n is the arity of P. For each constant $c \in \mathsf{CONS}$, $\mathcal{I}(c) \in \mathcal{D}$. An *assignment* v is defined as a function mapping variables and parameters to elements of the domain. An *x-variant* v' of v agrees with v on all arguments, save, possibly, x. We will write v^x_o to denote the x-variant of v with $v^x_o(x) = o$.

Given a signature $\Sigma = (\mathsf{PRED}, \mathsf{CONS}, \mathsf{ar})$, sets TERM and FOR over Σ, a model $\mathcal{M} = (\mathcal{D}, \mathcal{I})$ over Σ, and an assignment v, let $\mathcal{I}_v(s)$ be $v(s)$ if $s \in \mathsf{VAR} \cup \mathsf{PAR}$, or $\mathcal{I}(s)$ if $s \in \mathsf{CONS}$. Then the notion of *satisfaction of a formula φ in $\mathcal{M}$ under v*, in symbols $M, v \models \varphi$, is defined inductively as follows:

$$\begin{aligned}
\mathcal{M}, v \models P(s_1, ..., s_n) &\text{ iff } \langle \mathcal{I}_v(s_1), \ldots, \mathcal{I}_v(s_n) \rangle \in \mathcal{I}(P) \\
\mathcal{M}, v \models s_1 = s_2 &\text{ iff } \mathcal{I}_v(s_1) = \mathcal{I}_v(s_2) \\
\mathcal{M}, v \models (\lambda x\psi)(s) &\text{ iff } \mathcal{M}, v^x_o \models \psi \text{ and } o = \mathcal{I}_v(s) \\
\mathcal{M}, v \models (\lambda x\psi)(\imath y\varphi) &\text{ iff there exists } o \in \mathcal{D} \text{ such that } \mathcal{M}, v^x_o \models \varphi[y/x], \\
&\qquad \mathcal{M}, v^x_o \models \psi, \text{ and for any } y\text{-variant } v' \text{ of } v^x_o, \text{ if} \\
&\qquad \mathcal{M}, v' \models \varphi, \text{ then } v'(y) = o \\
\mathcal{M}, v \models \neg\varphi &\text{ iff } \mathcal{M}, v \not\models \varphi, \\
\mathcal{M}, v \models \varphi \wedge \psi &\text{ iff } \mathcal{M}, v \models \varphi \text{ and } M, v \models \psi, \\
\mathcal{M}, v \models \forall x\varphi &\text{ iff } \mathcal{M}, v^x_o \models \varphi, \text{ for all } o \in \mathcal{D},
\end{aligned}$$

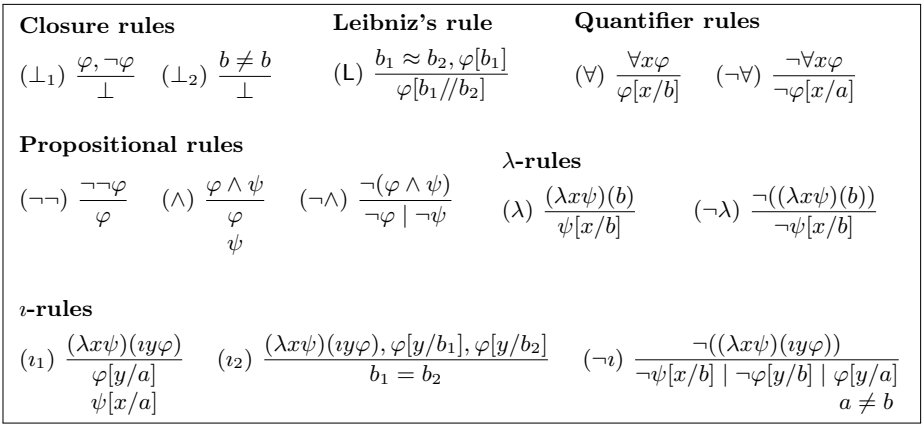

Fig. 1. Rules of the tableau calculus TC_{R_λ}

where $P \in \mathsf{PRED}$ with $\mathsf{ar}(P) = n$, $s, s_1, \ldots, s_n \in \mathsf{TERM}^-$, $x, y \in \mathsf{VAR}$, and $\varphi, \psi \in \mathsf{FOR}$, and x is not free in φ in condition for $(\lambda x\psi)(\iota y\varphi)$.

A formula φ over a signature Σ is called *satisfiable* if there exist a model $\mathcal{M}$ over Σ and a valuation v such that $\mathcal{M}, v \models \varphi$. φ is *valid*, in symbols $\models \varphi$, if, for all models $\mathcal{M}$ and valuations v, $\mathcal{M}, v \models \varphi$. In the remainder of the paper, instead of writing $\mathcal{M}, v \models \varphi_1, \ldots, \mathcal{M}, v \models \varphi_n$, we will write $\mathcal{M}, v \models \varphi_1, \ldots, \varphi_n$. Semantically we identify L_{R_λ} as the set of all valid formulas.

3 Calculus

In this section we focus on the construction of a tableau calculus for the logic of Russellian descriptions L_{R_λ}, henceforth abbreviated as TC_{R_λ}.

A *tableau* $\mathcal{T}$ generated by a calculus TC_{R_λ} is a *derivation tree* whose nodes are assigned formulas in the language of deduction. A *branch of* $\mathcal{T}$ is a simple path from the root to a leaf of $\mathcal{T}$. For simplicity, we will identify each branch $\mathcal{B}$ with the set of formulas assigned to nodes on $\mathcal{B}$.

In Figure 1 we present the rules constituting TC_{R_λ}. We transfer the notation from the previous section with the caveat that a denotes a parameter that is *fresh* on the branch, whereas b, b_1, b_2 denote parameters or constants already been present on the branch. Finally, $b_1 \approx b_2$ stands for either $b_1 = b_2$ or $b_2 = b_1$.

Theorem 1. *The tableau calculus* TC_{R_λ} *is sound and complete w.r.t. the semantics from Section 2.*

4 Interpolation

A nice feature of our calculus TC_{R_λ} is that it can be used to constructively show that the logic L_{R_λ} enjoys the Craig interpolation property. To this end we exploit

a technique introduced by Smullyan [7] and further adjusted to the tableaux setting by Fitting [2]. To take full advantage of this method we need to modify TC_{R_λ}, so that all the rules, save $(\bot_1)$, are single-premise rules. Consider the following two transformed rules: $(\mathsf{L}')\ \frac{\varphi[b_1]}{b_1 \neq b_2 | \varphi[b_1 // b_2]}$ and $(\imath_2')\ \frac{(\lambda x\psi)(\imath y\varphi)}{\neg\varphi[y/b_1] | \neg\varphi[y/b_2] | b_1 = b_2}$. Let TC'_{R_λ} be TC_{R_λ} with (L) and $(\imath_2)$ replaced with (L') and $(\imath_2')$, respectively. To show that TC_{R_λ} and TC'_{R_λ} are equivalent we will exploit the cut rule: $(\mathsf{cut})\ \frac{}{\varphi | \neg\varphi}$ and the following proposition.

Proposition 1. *The rule* (cut) *is admissible in* TC_{R_λ}.

Therefore we can apply (cut) safely in TC_{R_λ} to show derivability of other rules.

Lemma 1. *The tableau calculi* TC_{R_λ} *and* TC'_{R_λ} *are equivalent.*

Theorem 2. *The logic* L_{R_λ} *enjoys Craig's interpolation property.*

Proof (excerpt). The general scheme we exploit throughout the proof is:

> If $\chi_1, ..., \chi_k$ are interpolants for $\Gamma \cup \{\Psi_1\}, \ldots, \Gamma \cup \{\Psi_k\}$, then $\mathtt{I}(\chi_1, \ldots, \chi_k)$ is an interpolant for $\Gamma \cup \{\varphi\}$, where φ is the premise of the applied rule and $\Psi_1, \ldots, \Psi_k$ are all the (sets of) conclusions, and Γ is the set of all formulas on the branch above the premise.

Clearly, the specific rules for calculating interpolants are in two versions for each rule: the L-variant with L, or the R-variant with R assigned to the premise and conclusions (in the case of $(\bot_1)$ there are four combinations). Let $\gamma_1, ..., \gamma_n$ be the set of all formulas such that $\mathtt{L}\ \gamma_i \in \Gamma$, and let $\delta_1, ..., \delta_m$ be the set of all formulas such that $\mathtt{R}\ \delta_i \in \Gamma$. For each rule we are showing that for its L-variant:

> If, for every $i \leq k$, $\models \psi_i \wedge \gamma_1 \wedge \ldots \wedge \gamma_n \rightarrow \chi_i$ and $\models \chi_i \rightarrow \neg\delta_1 \vee \ldots \vee \neg\delta_m$, then $\models \varphi \wedge \gamma_1 \wedge \ldots \wedge \gamma_n \rightarrow I(\chi_1, \ldots, \chi_k)$.

and for the R-variant:

> If, for every $i \leq k$, $\models \gamma_1 \wedge \ldots \wedge \gamma_n \rightarrow \chi_i$ and $\models \chi_i \rightarrow \neg\delta_1 \vee \ldots \vee \neg\delta_m \vee \neg\psi_i$, then $\models I(\chi_1, \ldots, \chi_k) \rightarrow \neg\delta_1 \vee \ldots \vee \neg\delta_m \vee \neg\varphi$.

Below we state the principles for calculating interpolants for the specific rules of TC'_{R_λ}. For the remaining rules they can be found in [2].

$(\bot_2)$ $\bot$ is an interpolant for $\Gamma \cup \{\mathtt{X}\ b \neq b\}$, for $\mathtt{X} \in \{\mathtt{L}, \mathtt{R}\}$.

(L') If χ_1 is an interpolant for $\Gamma \cup \{\mathtt{L}\ b_1 \neq b_2\}$ and χ_2 is an interpolant for $\Gamma \cup \{\mathtt{L}\ \varphi[b_1 // b_2]\}$, then $\forall x(\chi_1 \vee \chi_2[b_2/x])$ is an interpolant for $\Gamma \cup \{\mathtt{L}\ \varphi[b_1]\}$.

(L') If χ_1 is an interpolant for $\Gamma \cup \{\mathtt{R}\ b_1 \neq b_2\}$ and χ_2 is an interpolant for $\Gamma \cup \{\mathtt{R}\ \varphi[b_1 // b_2]\}$, then $\exists x(\chi_1 \wedge \chi_2[b_2/x])$ is an interpolant for $\Gamma \cup \{\mathtt{R}\ \varphi[b_1]\}$.

(λ) If χ is an interpolant for $\Gamma \cup \{\mathtt{X}\ \psi[x/b]\}$, then χ is an interpolant for $\Gamma \cup \{\mathtt{X}\ (\lambda x\psi)(b)\}$, for $\mathtt{X} \in \{\mathtt{L}, \mathtt{R}\}$.

$(\neg\lambda)$ If χ is an interpolant for $\Gamma \cup \{\mathtt{X}\ \neg\psi[x/b]\}$, then χ is an interpolant for $\Gamma \cup \{\mathtt{X}\ \neg((\lambda x\psi)(b))\}$, for $\mathtt{X} \in \{\mathtt{L}, \mathtt{R}\}$.

$(\imath_1)$ If χ is an interpolant for $\Gamma \cup \{\mathtt{X}\ \psi[x/a], \mathtt{X}\ \varphi[x/a]\}$, then χ is an interpolant for $\Gamma \cup \{\mathtt{X}\ (\lambda x\psi)(\imath y\varphi)\}$, for $\mathtt{X} \in \{\mathtt{L}, \mathtt{R}\}$.

(ι_2') If χ_1 is an interpolant for $\Gamma \cup \{\mathtt{L}\ \neg\varphi[y/b_1]\}$, χ_2 is an interpolant for $\Gamma \cup \{\mathtt{L}\ \neg\varphi[y/b_2]\}$ and χ_3 is an interpolant for $\Gamma \cup \{\mathtt{L}\ b_1 = b_2\}$, then $\forall x \forall y(\chi_1 \vee \chi_2 \vee \chi_3[b_1/x, b_2/y])$ is an interpolant for $\Gamma \cup \{\mathtt{L}\ (\lambda x\psi)(\iota y\varphi)\}$.

(ι_2') If χ_1 is an interpolant for $\Gamma \cup \{\mathtt{R}\ \neg\varphi[y/b_1]\}$, χ_2 is an interpolant for $\Gamma \cup \{\mathtt{R}\ \neg\varphi[y/b_2]\}$ and χ_3 is an interpolant for $\Gamma \cup \{\mathtt{R}\ b_1 = b_2\}$, then $\exists x \exists y(\chi_1 \wedge \chi_2 \wedge \chi_3[b_1/x, b_2/y])$ is an interpolant for $\Gamma \cup \{\mathtt{R}\ (\lambda x\psi)(\iota y\varphi)\}$.

($\neg\iota$) If χ_1 is an interpolant for $\Gamma \cup \{\mathtt{L}\ \neg\psi[y/b]\}$, χ_2 is an interpolant for $\Gamma \cup \{\mathtt{L}\ \neg\varphi[y/b]\}$ and χ_3 is an interpolant for $\Gamma \cup \{\mathtt{L}\ \varphi[y/a], \mathtt{L}\ a \neq b\}$, then $\forall x(\chi_1 \vee \chi_2 \vee \chi_3[b/x])$ is an interpolant for $\Gamma \cup \{\mathtt{L}\ \neg((\lambda x\psi)(\iota y\varphi))\}$.

($\neg\iota$) If χ_1 is an interpolant for $\Gamma \cup \{\mathtt{R}\ \neg\psi[y/b]\}$, χ_2 is an interpolant for $\Gamma \cup \{\mathtt{R}\ \neg\varphi[y/b]\}$ and χ_3 is an interpolant for $\Gamma \cup \{\mathtt{R}\ \varphi[y/a], \mathtt{L}\ a \neq b\}$, then $\exists x(\chi_1 \wedge \chi_2 \wedge \chi_3[b/x])$ is an interpolant for $\Gamma \cup \{\mathtt{R}\ \neg((\lambda x\psi)(\iota y\varphi))\}$ □

As a consequence of Theorem 2 (see [3]) we get:

Theorem 3. *The logic L_{R_λ} enjoys Beth's definability property.*

Knowing that L_{R_λ} satisfies Beth's definability property results in a straightforward method of determining, for any signature $\Sigma = (\mathsf{PRED}, \mathsf{CONS}, \mathsf{ar})$, any theory Th over Σ and any constant $c \in \mathsf{CONS}$, whether c can be defined by a formula ψ over $\Sigma' = (\mathsf{PRED}, \mathsf{CONS} \setminus \{c\}, \mathsf{ar})$ under Th (see the work by Artale et al. [1]). In other words, we can decide whether there exists a formula that can form a definite description satisfied by the object that c denotes. Indeed, it suffices to check for implicit definability of c, that is, check if a tableau with the following formula at the root is closed:

$$\bigwedge(\mathsf{Th} \cup \mathsf{Th}') \wedge c \neq c',$$

where Th' is defined as Th with a fresh constant c' put in place of c. Since, by Theorem 3, implicit definability implies explicit definability, we can replace such a constant with a definite description $\iota x\psi$, where ψ is a definiens of c, whenever such a replacement results in a syntactically correct expression.

References

1. Artale, A., Mazzullo, A., Ozaki, A., Wolter, F.: On free description logics with definite descriptions. In: Bienvenu, M., Lakemeyer, G., Erdem, E. (eds.) Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning, KR 2021, Online event, November 3-12, 2021. pp. 63–73 (2021). https://doi.org/10.24963/kr.2021/7
2. Fitting, M.: First-Order Logic and Automated Theorem Proving, Graduate Texts in Computer Science, vol. 277. Springer-Verlag, New York (1996). https://doi.org/10.1007/978-1-4612-2360-3
3. Hermes, H.: Introduction to Mathematical Logic. Universitext, Springer-Verlag, Berlin, Heidelberg (1973). https://doi.org/10.1007/978-3-642-87132-0

4. Indrzejczak, A.: Russellian definite description theory—A proof-theoretic approach. The Review of Symbolic Logic **online first**, 1–26 (2021). https://doi.org/10.1017/S1755020321000289
5. Indrzejczak, A., Zawidzki, M.: Tableaux for free logics with descriptions. In: Proceedings of the 30th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods. pp. 56–73. TABLEAUX'21, Springer-Verlag, Berlin, Heidelberg (2021). https://doi.org/10.1007/978-3-030-86059-2_4
6. Russell, B.: On Denoting. Mind **XIV**(4), 479–493 (1905). https://doi.org/10.1093/mind/XIV.4.479
7. Smullyan, R.M.: First-Order Logic. Ergebnisse der Mathematik und ihrer Grenzgebiete, Springer Verlag, Berlin, Heidelberg (1968). https://doi.org/10.1007/978-3-642-86718-7
8. Whitehead, A.N., Russell, B.: Principia Mathematica, vol. I. Cambridge University Press, Cambridge (1910). https://doi.org/10.1017/CBO9780511623585

Interpolants and Interference

Adrián Rebola-Pardo[1,2⋆]

[1] Vienna University of Technology, Austria
[2] Johannes Kepler University, Austria
arebolap@forsyte.tuwien.ac.at

Abstract. Interference-based proof systems are necessary for proof generation in inprocessing SAT solvers. Unfortunately, their structure and semantic invariants are incompatible with standard recursive interpolant systems. Recent research on interference connects it with inference in more expressive logics. This circumvents some of the roadblocks in the quest for interpolant extraction from interference-based proofs, but also raises questions about the applicability of unfeasibility results in this setting.

1 SAT solving, interpolation and inprocessing

The execution of *conflict-driven clause learning* (CDCL) SAT solvers [35,25] matches closely a resolution refutation in ways that are well understood: each learnt clause can be derived from the working CNF formula via trivial resolution [3,28]. CDCL solvers exploit this by generating a resolution refutation when given an unsatisfiable input formula; this resolution proof can be used to generate a Robinson interpolant [6] using one of the many available methods [17,19,29,34,9,7]. These interpolants are extremely useful for industrial techniques in model checking [24,41,42].

In order to make up for the traditionally poor performance of CDCL solvers on instances involving parity constraints or symmetries, several *inprocessing techniques* have been introduced [22,10,37,2,4]. Unfortunately, generating resolution proofs for these formula transformations can be very complicated. Gaussian elimination [37,36], for instance, derives inferences that generally require exponentially-sized proofs to derive in resolution [39]; symmetry breaking [1,2] introduces predicates that are not logical consequences of the input formula, so they cannot even be derived by resolution [40].

Most SAT solvers generate their unsatisfiability proofs instead in the *delete resolution asymmetric tautology* (DRAT) proof system [43,14], which enables short proofs when inprocessing is used [23,12,26]. DRAT generalizes the *extended resolution* (ER) proof system, and is polynomially equivalent to it [21,18]. Hence, DRAT lacks feasible interpolation under cryptographic assumptions [20,33].

⋆ Supported by FWF through the LogiCS doctoral program W1255-N23, by WWTF through grants VRG11-005 and ICT15-103, and by Microsoft Research through its PhD Scholarship Programme.

Unfeasible interpolation is not by itself a caveat: if the overhead of interpolant generation is lower than the overhead of disabling inprocessing in SAT solvers, a recursive interpolation system might still be desirable in practice. Furthermore, our research on the logical underpinnings of DRAT suggests that some changes on the definition of an interpolant might be natural in this setting, and most importantly, compatible with model checking applications. Our goal with this talk is to receive expert feedback on this line of research.

2 Interference-based Proofs

DRAT and ER belong to a class of propositional proof systems generically called *interference-based* within the SAT solving community [13]. As opposed to the idea of *infering* a clause, i.e. inserting in the working formula F a clause C entailed by F, interference only requires that F and $F \wedge C$ are equisatisfiable. The consequences of this change are far-reaching:

- Whereas inference-based proofs preserve models (i.e. any model of the premises is a model of the conclusion), interference-based proofs only preserve the *existence* of a model; in particular, any model of the premises needs not be a model of the conclusion [11].
- Most inference-based proof systems are defined through inference rules of the form $A_1, \ldots, A_n \vdash C$, where $A_1, \ldots, A_n$ are required to occur in the formula F accumulated throughout the proof; critically, the presence or absence of other clauses is irrelevant. In interference-based proofs, clauses are derived through rules that involve the whole formula F. In other words, interference does not only depend on the presence of some clauses, but also *on the absence* of some clauses [31].
- As a consequence of the previous difference, interference rules are non-monotonic, i.e. a proof step deriving C from F might become incorrect when F is enlarged. This is reflected on the absence of the tree structure, otherwise typical in inference-based proofs, in interference-based proofs. Rather, the latter take the form of lists of clause introductions and deletions, incrementally inserting or removing clauses from the premise formula [30].

There are many examples of interference-based proof systems, including ER [21,33], DRAT [43], DPR [16], DSR [5] and WSR [30]. For this paper we focus on *delete subsumption-redundancy* (DSR), which generalizes DRAT.

A clause C is called a *reverse unit propagation* (RUP) over a CNF formula F whenever C can be derived from F through a trivial resolution derivation [3,28]; for the purposes of this paper, it suffices to consider this as a form of chained resolution. In that case, C is logically implied by F. Furthermore, we call a substitution σ *atomic* whenever σ maps variables to literals, $\top$ or $\bot$. Then, a clause C is a *substitution redundancy* (SR) clause [5] over F upon an atomic substitution σ whenever the following conditions hold:

1. The clause $\sigma(C)$ is a tautology.

2. For each clause $D \in F$, the clause $C \vee \sigma(D)$ is either a tautology or a RUP clause over F.

Under these conditions, F and $F \wedge C$ is equisatisfiable.

The *DSR proof system* allows the inconditional deletion of clauses, and the introduction of clauses C so far as they are either RUP clauses over the accumulated CNF formula F, or SR clauses over F upon a specified atomic substitution σ. For a proof system operating on clauses, DSR is very powerful. Being polynomially equivalent to ER [18], no exponential lower bounds are known, Short, intuitive refutations for complex formulas are relatively easy to produce [26,12,23,8], albeit further improvements on this are possible using WSR [30]. Furthermore, for equisatisfiable formulas F and G, there always exist DSR derivations of G from F [27].

3 Recursive Interpolation Systems

The departure of interference from the more common tree-shaped, monotonic, entailment-based proof framework has some consequences for the development of a recursive interpolation system. While it is possible to transform interference-based proofs into resolution [32] (of course, at an exponential cost), generating interpolants directly from the former seems contrived at the very least.

Typical recursive interpolation systems work by generating a *partial interpolant* for each node in a refutation tree, with the partial interpolant for the final contradiction being the interpolant. A semantic invariant generalizing the definition of an interpolant is maintained for partial interpolants by the generation rules. Let us take as an example a simple interpolation system for resolution proofs inspired by [17,7]. For a clause C derived from the partition $A \wedge B$, we generate the partial interpolant $P(C)$ as follows:

- If C is a premise in A, then $P(C) = \bot$; if C is a premise in B, then $P(C) = \top$.
- If C is derived as the resolvent of D_1 and D_2 upon a pivot literal x with $x \in D_1$ and $\overline{x} \in D_2$, then $P(C)$ is:

$$\begin{array}{rl} P(D_1) \vee P(D_2) & \text{if } x \in \mathrm{Var}(A) \setminus \mathrm{Var}(B) \\ (x \wedge P(D_1)) \vee (\overline{x} \wedge P(D_2)) & \text{if } x \in \mathrm{Var}(A) \cap \mathrm{Var}(B) \\ P(D_1) \wedge P(D_2) & \text{if } x \in \mathrm{Var}(B) \setminus \mathrm{Var}(A) \end{array}$$

where we have denoted by F the set of variables occurring in F. Partial interpolants generated in this way maintain the following semantic invariant:

1. A implies $P(C) \vee C|_{\mathrm{Var}(A)}$
2. B implies $\neg P(C) \vee C|_{\mathrm{Var}(B)}$
3. $\mathrm{Var}(P(C)) \subseteq \mathrm{Var}(A) \cap \mathrm{Var}(B)$

where $C|_V$ for a clause C and a set of variables V denotes the clause constructed by removing all literals x and $\overline{x}$ with $x \in V$ from C.

While modifying this interpolation system to fit inference-based proof systems other than resolution is not difficult, doing so for interference-based proofs is. To start with, interference precludes the expression of proofs as trees. There still exists some proof structure (namely, the list of introductions and deletions), yet a much weaker one. However, whereas inferences are entailment-based (e.g. the resolvent of D_1 and D_2 is a logical consequence of D_1 and D_2), interference proof steps only preserve satisfiability. Proving that the invariant above is maintained uses very strongly that inferences are truth-preserving, which we cannot guarantee for interference.

4 Mutation Semantics

Our research has however shown that we can see interference-based proofs as inference-based proofs on an extension of propositional logic we call *mutation logic* [31,30]. This extension employs a *conditional mutation* operator ∇. Given two formulas φ and ψ and an atomic substitution σ, the formula $\nabla(\sigma \coloneqq \psi).\varphi$ is satisfied by a model m whenever: 1. If m satisfies ψ, then m satisfies $\sigma(\varphi)$, and 2. If m falsifies ψ, then m satisfies φ.

The intuition behind ∇ is that φ will be evaluated under a model that is obtained from m by *mutating* it through σ whenever the condition ψ holds, and by keeping m as is otherwise. The ∇ operator is also relatively well-behaved: ∇ distributes over $\neg$, $\vee$, $\wedge$; if C implies D, then $\nabla\varepsilon.C$ implies $\nabla\varepsilon.D$.

Mutation logic has some interesting complexity properties [31,30]. Its version without restrictions, which we call MPL, can be linearly encoded as a logically equivalent propositional circuit; this means that the satisfiability problem for MPL is NP-complete [38], and models can be easily obtained using a SAT solver. In our research, we consider *cubic mutation rules* of the form $\sigma \coloneqq \neg C$ where C is a clause; and the *uniform mutation CNF* (UMCNF) fragment of formulas $\nabla\varepsilon_1 \dots \nabla\varepsilon_n.F$ where the ε_i are cubic rules and F is a CNF formula. We showed that all MPL formulas have a linearly-sized, equivalent UMCNF formula.

Mutation logic is extremely useful in understanding the logical underpinnings of interference. In particular, one can show that, if C is an SR clause over F upon σ, then the formula F entails $\nabla(\sigma \coloneqq \neg C).(F \wedge C)$ [31]. Two things are worth noting at this point. Firstly, here we have entailment as opposed to satisfiability-equivalence; in fact, the rule $(\sigma \coloneqq \neg C)$ tells us exactly how to construct a model of $F \wedge C$ given *any* model of F. Secondly, one can reinterpret SR introduction as reasoning without loss of generality [30], which explains why DSR can introduce symmetry breaking predicates [12].

Equipped with mutation logic, an inference-based proof system over *mutation clauses* that closely matches inference-based proof systems over clauses can be constructed. Mutation clauses present as $\nabla\varepsilon_1 \dots \nabla\varepsilon_n.C$, where the ε_i are cubic rules and C is a clause. On the one hand, this yields a much better understanding of the reasoning process behind the definition of DSR, immediately

pointing to novel extensions [30]. On the other hand, this proof system reattains tree-shaped, truth-preserving, monotonic inference rules that simulate the reasoning performed by interference. We do not present this proof system here; the interested reader can find details about the *mutation-resolution* proof system in [31].

5 Interpolation for Mutation Logic

Mutation logic might provide a path forward in our quest for interpolant generation in SAT solvers with inprocessing. If we had a recursive interpolation system for the mutation-resolution proof system, generating an MPL interpolant, then we could use this interpolant in model checking applications via the aforementioned linear transformation into a propositional circuit.

There remain some roadblocks in constructing such an interpolation system, nevertheless. Proof complexity results have shown that, assuming that RSA is secure, the *extended Frege* (EF) proof system lacks feasible interpolation [20]. EF is itself polynomially equivalent to ER [33], hence also to the whole family of interference-based proof systems [18,15,5], so this (probably) precludes feasible interpolation for mutation-resolution as well.

A question, however, raises when extending the definition of an interpolant to MPL: what does it mean for x to be a variable of an MPL formula φ? Naïvely, we could simply interpret that x *syntactically* occurs in φ. However, in the formula

$$\nabla(\{y \mapsto \top\} \coloneqq x).\nabla(\{y \mapsto \bot\} \coloneqq \overline{x}).y$$

the variable y plays no role at all and can be replaced by any other variable (including x itself!). On the other hand, a definition of the variables of a formula that somehow considers y a "bound" variable does not seem to be very useful either, since in the following formula y *is* relevant:

$$\nabla(\{y \mapsto \bot\} \coloneqq \overline{x}).y$$

An alternative is to take a *semantic* definition of the variables of a formula: x is a variable of φ whenever there exist models m_1 and m_2 differing only on x which assign different truth values to φ. A variation of interpolants, which we call *semantic interpolants*, can then be defined by substituting semantic variables for syntactic variables. This might, in fact, be a reasonable change even in propositional logic: with this definition, if P_1 is equivalent to P_2, then P_1 is an interpolant of $A \wedge B$ iff P_2 is.

With this, we motivate some questions with a direct influence in this line of research, which we hope to discuss during iPRA 2022:

- Does the unfeasible interpolation result for EF from [20] still hold for semantic interpolants?
- Are there other variations on the definition of an interpolant that might be useful for model checking, and in particular for the extraction of interpolants from interference-based proofs?
- Do these insights shed any light over the connection between cryptographic assumptions and feasible interpolation?

References

1. Aloul, F.A., Ramani, A., Markov, I.L., Sakallah, K.A.: Solving difficult instances of boolean satisfiability in the presence of symmetry. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **22**(9), 1117–1137 (2003). https://doi.org/10.1109/TCAD.2003.816218, https://doi.org/10.1109/TCAD.2003.816218
2. Aloul, F.A., Sakallah, K.A., Markov, I.L.: Efficient symmetry breaking for boolean satisfiability. IEEE Trans. Computers **55**(5), 549–558 (2006). https://doi.org/10.1109/TC.2006.75, https://doi.org/10.1109/TC.2006.75
3. Beame, P., Kautz, H.A., Sabharwal, A.: Towards understanding and harnessing the potential of clause learning. J. Artif. Intell. Res. **22**, 319–351 (2004). https://doi.org/10.1613/jair.1410, https://doi.org/10.1613/jair.1410
4. Biere, A., Järvisalo, M., Kiesl, B.: Preprocessing in SAT solving. In: Biere, A., Heule, M., van Maaren, H., Walsh, T. (eds.) Handbook of Satisfiability, Frontiers in Artificial Intelligence and Applications, vol. 336, pp. 391–435. IOS Press (2021). https://doi.org/10.3233/FAIA200992
5. Buss, S., Thapen, N.: DRAT proofs, propagation redundancy, and extended resolution. In: Janota, M., Lynce, I. (eds.) Theory and Applications of Satisfiability Testing - SAT 2019 - 22nd International Conference, SAT 2019, Lisbon, Portugal, July 9-12, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11628, pp. 71–89. Springer (2019). https://doi.org/10.1007/978-3-030-24258-9_5, https://doi.org/10.1007/978-3-030-24258-9_5
6. Craig, W.: Linear reasoning. A new form of the herbrand-gentzen theorem. J. Symb. Log. **22**(3), 250–268 (1957). https://doi.org/10.2307/2963593, https://doi.org/10.2307/2963593
7. D'Silva, V., Kroening, D., Purandare, M., Weissenbacher, G.: Interpolant strength. In: Barthe, G., Hermenegildo, M.V. (eds.) Verification, Model Checking, and Abstract Interpretation, 11th International Conference, VMCAI 2010, Madrid, Spain, January 17-19, 2010. Proceedings. Lecture Notes in Computer Science, vol. 5944, pp. 129–145. Springer (2010). https://doi.org/10.1007/978-3-642-11319-2_12, https://doi.org/10.1007/978-3-642-11319-2_12
8. Gelder, A.V.: Producing and verifying extremely large propositional refutations - have your cake and eat it too. Ann. Math. Artif. Intell. **65**(4), 329–372 (2012). https://doi.org/10.1007/s10472-012-9322-x, https://doi.org/10.1007/s10472-012-9322-x
9. Gurfinkel, A., Vizel, Y.: Druping for interpolates. In: Formal Methods in Computer-Aided Design, FMCAD 2014, Lausanne, Switzerland, October 21-24, 2014. pp. 99–106. IEEE (2014). https://doi.org/10.1109/FMCAD.2014.6987601, https://doi.org/10.1109/FMCAD.2014.6987601
10. Heule, M., Järvisalo, M., Biere, A.: Revisiting hyper binary resolution. In: Gomes, C.P., Sellmann, M. (eds.) Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 10th International Conference, CPAIOR 2013, Yorktown Heights, NY, USA, May 18-22, 2013. Proceedings. Lecture Notes in Computer Science, vol. 7874, pp. 77–93. Springer (2013). https://doi.org/10.1007/978-3-642-38171-3_6, https://doi.org/10.1007/978-3-642-38171-3_6
11. Heule, M., Järvisalo, M., Lonsing, F., Seidl, M., Biere, A.: Clause elimination for SAT and QSAT. J. Artif. Intell. Res. **53**, 127–168 (2015). https://doi.org/10.1613/jair.4694, https://doi.org/10.1613/jair.4694

12. Heule, M., Jr., W.A.H., Wetzler, N.: Expressing symmetry breaking in DRAT proofs. In: Felty, A.P., Middeldorp, A. (eds.) Automated Deduction - CADE-25 - 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings. Lecture Notes in Computer Science, vol. 9195, pp. 591–606. Springer (2015). https://doi.org/10.1007/978-3-319-21401-6_40, https://doi.org/10.1007/978-3-319-21401-6_40
13. Heule, M., Kiesl, B.: The potential of interference-based proof systems. In: Reger, G., Traytel, D. (eds.) ARCADE 2017, 1st International Workshop on Automated Reasoning: Challenges, Applications, Directions, Exemplary Achievements, Gothenburg, Sweden, 6th August 2017. pp. 51–54. EPiC Series in Computing, EasyChair (2017), https://easychair.org/publications/paper/TWVW
14. Heule, M.J.H.: The DRAT format and drat-trim checker. CoRR **abs/1610.06229** (2016), http://arxiv.org/abs/1610.06229
15. Heule, M.J.H., Biere, A.: What a difference a variable makes. In: Beyer, D., Huisman, M. (eds.) Tools and Algorithms for the Construction and Analysis of Systems - 24th International Conference, TACAS 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10806, pp. 75–92. Springer (2018). https://doi.org/10.1007/978-3-319-89963-3_5, https://doi.org/10.1007/978-3-319-89963-3_5
16. Heule, M.J.H., Kiesl, B., Biere, A.: Short proofs without new variables. In: de Moura, L. (ed.) Automated Deduction - CADE 26 - 26th International Conference on Automated Deduction, Gothenburg, Sweden, August 6-11, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10395, pp. 130–147. Springer (2017). https://doi.org/10.1007/978-3-319-63046-5_9, https://doi.org/10.1007/978-3-319-63046-5_9
17. Huang, G.: Constructing craig interpolation formulas. In: Du, D., Li, M. (eds.) Computing and Combinatorics, First Annual International Conference, COCOON '95, Xi'an, China, August 24-26, 1995, Proceedings. Lecture Notes in Computer Science, vol. 959, pp. 181–190. Springer (1995). https://doi.org/10.1007/BFb0030832, https://doi.org/10.1007/BFb0030832
18. Kiesl, B., Rebola-Pardo, A., Heule, M.J.H.: Extended resolution simulates DRAT. In: Galmiche, D., Schulz, S., Sebastiani, R. (eds.) Automated Reasoning - 9th International Joint Conference, IJCAR 2018, Held as Part of the Federated Logic Conference, FloC 2018, Oxford, UK, July 14-17, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10900, pp. 516–531. Springer (2018). https://doi.org/10.1007/978-3-319-94205-6_34, https://doi.org/10.1007/978-3-319-94205-6_34
19. Krajícek, J.: Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. J. Symb. Log. **62**(2), 457–486 (1997). https://doi.org/10.2307/2275541, https://doi.org/10.2307/2275541
20. Krajícek, J., Pudlák, P.: Some consequences of cryptographical conjectures for s^1_2 and EF. Inf. Comput. **140**(1), 82–94 (1998). https://doi.org/10.1006/inco.1997.2674, https://doi.org/10.1006/inco.1997.2674
21. Kullmann, O.: On a generalization of extended resolution. Discret. Appl. Math. **96-97**, 149–176 (1999). https://doi.org/10.1016/S0166-218X(99)00037-2, https://doi.org/10.1016/S0166-218X(99)00037-2
22. Manthey, N., Heule, M., Biere, A.: Automated reencoding of boolean formulas. In: Biere, A., Nahir, A., Vos, T.E.J. (eds.) Hardware and Software: Verification and Testing - 8th International Haifa Verification Conference, HVC 2012,

Haifa, Israel, November 6-8, 2012. Revised Selected Papers. Lecture Notes in Computer Science, vol. 7857, pp. 102–117. Springer (2012). https://doi.org/10.1007/978-3-642-39611-3_14, https://doi.org/10.1007/978-3-642-39611-3_14
23. Manthey, N., Philipp, T.: Formula simplifications as DRAT derivations. In: Lutz, C., Thielscher, M. (eds.) KI 2014: Advances in Artificial Intelligence - 37th Annual German Conference on AI, Stuttgart, Germany, September 22-26, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8736, pp. 111–122. Springer (2014). https://doi.org/10.1007/978-3-319-11206-0_12, https://doi.org/10.1007/978-3-319-11206-0_12
24. McMillan, K.L.: Interpolation and sat-based model checking. In: Jr., W.A.H., Somenzi, F. (eds.) Computer Aided Verification, 15th International Conference, CAV 2003, Boulder, CO, USA, July 8-12, 2003, Proceedings. Lecture Notes in Computer Science, vol. 2725, pp. 1–13. Springer (2003). https://doi.org/10.1007/978-3-540-45069-6_1, https://doi.org/10.1007/978-3-540-45069-6_1
25. Moskewicz, M.W., Madigan, C.F., Zhao, Y., Zhang, L., Malik, S.: Chaff: Engineering an efficient SAT solver. In: Proceedings of the 38th Design Automation Conference, DAC 2001, Las Vegas, NV, USA, June 18-22, 2001. pp. 530–535. ACM (2001). https://doi.org/10.1145/378239.379017, https://doi.org/10.1145/378239.379017
26. Philipp, T., Rebola-Pardo, A.: DRAT proofs for XOR reasoning. In: Michael, L., Kakas, A.C. (eds.) Logics in Artificial Intelligence - 15th European Conference, JELIA 2016, Larnaca, Cyprus, November 9-11, 2016, Proceedings. Lecture Notes in Computer Science, vol. 10021, pp. 415–429 (2016). https://doi.org/10.1007/978-3-319-48758-8_27, https://doi.org/10.1007/978-3-319-48758-8_27
27. Philipp, T., Rebola-Pardo, A.: Towards a semantics of unsatisfiability proofs with inprocessing. In: Eiter, T., Sands, D. (eds.) LPAR-21, 21st International Conference on Logic for Programming, Artificial Intelligence and Reasoning, Maun, Botswana, May 7-12, 2017. EPiC Series in Computing, vol. 46, pp. 65–84. EasyChair (2017), https://easychair.org/publications/paper/V8G
28. Pipatsrisawat, K., Darwiche, A.: On the power of clause-learning SAT solvers as resolution engines. Artif. Intell. **175**(2), 512–525 (2011). https://doi.org/10.1016/j.artint.2010.10.002, https://doi.org/10.1016/j.artint.2010.10.002
29. Pudlák, P.: Lower bounds for resolution and cutting plane proofs and monotone computations. J. Symb. Log. **62**(3), 981–998 (1997). https://doi.org/10.2307/2275583, https://doi.org/10.2307/2275583
30. Rebola-Pardo, A.: Interference-based proofs in SAT solving. Ph.D. thesis, TU Wien (2022)
31. Rebola-Pardo, A., Suda, M.: A theory of satisfiability-preserving proofs in SAT solving. In: Barthe, G., Sutcliffe, G., Veanes, M. (eds.) LPAR-22. 22nd International Conference on Logic for Programming, Artificial Intelligence and Reasoning, Awassa, Ethiopia, 16-21 November 2018. EPiC Series in Computing, vol. 57, pp. 583–603. EasyChair (2018), https://easychair.org/publications/paper/zr7z
32. Rebola-Pardo, A., Weissenbacher, G.: RAT elimination. In: Albert, E., Kovács, L. (eds.) LPAR 2020: 23rd International Conference on Logic for Programming, Artificial Intelligence and Reasoning, Alicante, Spain, May 22-27, 2020. EPiC Series in Computing, vol. 73, pp. 423–448. EasyChair (2020), https://easychair.org/publications/paper/cMtF
33. Reckhow, R.A.: On the lengths of proofs in the propositional calculus. Ph.D. thesis, University of Toronto (1975)

34. Schlaipfer, M., Weissenbacher, G.: Labelled interpolation systems for hyper-resolution, clausal, and local proofs. J. Autom. Reason. **57**(1), 3–36 (2016). https://doi.org/10.1007/s10817-016-9364-6, https://doi.org/10.1007/s10817-016-9364-6
35. Silva, J.P.M., Sakallah, K.A.: GRASP - a new search algorithm for satisfiability. In: Rutenbar, R.A., Otten, R.H.J.M. (eds.) Proceedings of the 1996 IEEE/ACM International Conference on Computer-Aided Design, ICCAD 1996, San Jose, CA, USA, November 10-14, 1996. pp. 220–227. IEEE Computer Society / ACM (1996). https://doi.org/10.1109/ICCAD.1996.569607, https://doi.org/10.1109/ICCAD.1996.569607
36. Soos, M.: Enhanced gaussian elimination in dpll-based SAT solvers. In: Berre, D.L. (ed.) POS-10. Pragmatics of SAT, Edinburgh, UK, July 10, 2010. EPiC Series in Computing, vol. 8, pp. 2–14. EasyChair (2010), https://easychair.org/publications/paper/j1D
37. Soos, M., Nohl, K., Castelluccia, C.: Extending SAT solvers to cryptographic problems. In: Kullmann, O. (ed.) Theory and Applications of Satisfiability Testing - SAT 2009, 12th International Conference, SAT 2009, Swansea, UK, June 30 - July 3, 2009. Proceedings. Lecture Notes in Computer Science, vol. 5584, pp. 244–257. Springer (2009). https://doi.org/10.1007/978-3-642-02777-2_24, https://doi.org/10.1007/978-3-642-02777-2_24
38. Tseitin, G.S.: On the complexity of derivation in propositional calculus. In: Siekmann, J.H., Wrightson, G. (eds.) Automation of Reasoning: 2: Classical Papers on Computational Logic 1967–1970, pp. 466–483. Springer Berlin Heidelberg (1983). https://doi.org/10.1007/978-3-642-81955-1_28, https://doi.org/10.1007/978-3-642-81955-1_28
39. Urquhart, A.: Hard examples for resolution. J. ACM **34**(1), 209–219 (1987). https://doi.org/10.1145/7531.8928, https://doi.org/10.1145/7531.8928
40. Urquhart, A.: The symmetry rule in propositional logic. Discret. Appl. Math. **96-97**, 177–193 (1999). https://doi.org/10.1016/S0166-218X(99)00039-6, https://doi.org/10.1016/S0166-218X(99)00039-6
41. Vizel, Y., Grumberg, O.: Interpolation-sequence based model checking. In: Proceedings of 9th International Conference on Formal Methods in Computer-Aided Design, FMCAD 2009, 15-18 November 2009, Austin, Texas, USA. pp. 1–8. IEEE (2009). https://doi.org/10.1109/FMCAD.2009.5351148, https://doi.org/10.1109/FMCAD.2009.5351148
42. Vizel, Y., Weissenbacher, G., Malik, S.: Boolean satisfiability solvers and their applications in model checking. Proc. IEEE **103**(11), 2021–2035 (2015). https://doi.org/10.1109/JPROC.2015.2455034, https://doi.org/10.1109/JPROC.2015.2455034
43. Wetzler, N., Heule, M., Jr., W.A.H.: DRAT-trim: Efficient checking and trimming using expressive clausal proofs. In: Sinz, C., Egly, U. (eds.) Theory and Applications of Satisfiability Testing - SAT 2014 - 17th International Conference, Held as Part of the Vienna Summer of Logic, VSL 2014, Vienna, Austria, July 14-17, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8561, pp. 422–429. Springer (2014). https://doi.org/10.1007/978-3-319-09284-3_31, https://doi.org/10.1007/978-3-319-09284-3_31