Report from Dagstuhl Seminar 21371

Integrated Deduction

Edited by

Maria Paola Bonacina[1], Philipp Rümmer[2], and Renate A. Schmidt[3]

1 **Università degli Studi di Verona, IT,** mariapaola.bonacina@univr.it
2 **Uppsala University, SE,** philipp.ruemmer@it.uu.se
3 **University of Manchester, GB,** renate.schmidt@manchester.ac.uk

Abstract

Logical reasoning plays a key role in fields as diverse as verification and synthesis, programming language foundations, knowledge engineering, and computer mathematics. Logical reasoning is increasingly important in intelligent systems, such as decision support systems, agent programming environments, and data processing systems, where deduction may provide explanation, course of action, and the capability of learning from missing information. Problem formalization in these domains typically involves multiple mathematical theories, knowledge bases, and ontologies, all of which may be very large. Problem solving requires both efficient automation and sophisticated human-machine interaction. The thrust of this seminar was that the key to unleash the power of computerized logical reasoning is *integration*, at different abstraction levels.

This seminar offered a forum to discuss the issues related to *integration of deduction* in a diverse range of applications. In terms of reasoning procedures, the presence of both theories and quantifiers in problems from many contexts calls for methodologies to *integrate state-of-the-art SMT solvers and automated theorem provers*. This leads to investigate techniques such as *model-based reasoning* and *semantic guidance*, that were presented and discussed at the seminar. Similarly, the *integration of inference rules for higher-order reasoning* in inference systems that were born for first-order reasoning, such as *superposition*, was prominent among the topics debated at the seminar. At the architectural level, the sheer difficulty of the problems calls for the *integration of provers and solvers* into *interactive reasoning environments*. These range from *higher-order proof assistants* with background reasoners as *hammers*, to *interactive program verifiers*, both widely covered at the seminar in talks and discussions.

The seminar showed how the application of deduction to intelligent systems necessitates the integration of deduction with other paradigms, such as *probabilistic reasoning* and *statistical inferences*. In fact, it emerged from the seminar that even systems that are not natively deductive, such as agent programming environments and industrial tools for ontology-based processing, benefit significantly from the integration of deduction. A clear and shared uptake from the seminar was that *scalability* and *usability* are crucial challenges at all levels of integration. The seminar fully succeded in promoting the exchange of new ideas and suggestions for future research.

Seminar September 12–17, 2021 – http://www.dagstuhl.de/21371
2012 ACM Subject Classification Theory of computation → Automated reasoning
Keywords and phrases Automated theorem proving, deduction, logic, reasoning, SMT solving
Digital Object Identifier 10.4230/DagRep.11.8.35
Edited in cooperation with Patrick Koopmann

1 Executive Summary

Maria Paola Bonacina
Philipp Rümmer
Renate A. Schmidt

This report contains the program and outcomes of the Dagstuhl Seminar 21371 on *Integrated Deduction* that was held at Schloss Dagstuhl, Leibniz Center for Informatics, during September 12–17, 2017. It was the fourteenth in a series of Dagstuhl Deduction seminars held biennially since 1993.

The motivation for this seminar was the following. Automated deduction has developed a wide and diverse range of methods and tools for logico-deductive reasoning. They include SAT solvers,[1] SMT solvers,[2] automated theorem provers, aka ATP systems, proof assistants, aka interactive theorem provers (ITP), as well as libraries of formalized mathematics and formalized knowledge. These methods and tools have found successful application in computing fields as diverse as the *analysis, verification, and synthesis of systems*, *programming language design*, *knowledge engineering*, and *computer mathematics*. However, no method, tool, paradigm, or even reasoning style can solve all problems, or respond to all demands coming from even a single field of application. Therefore, the next grand challenge for automated deduction is *integration*.

Integration occurs and is needed at different abstraction levels. Within deduction itself, integration of deductive engines allows us to build more powerful, more flexible, more expressive reasoners, that can solve more problems with fewer resources, meaning not only memory and computing time, but also human time and human expertise, the latter two often being the most precious of resources. Next, deductive reasoners get integrated into other tools, such as *automated test generators*, *verifying compilers*, or *program synthesizers*, just to name a few. Yet another level of integration occurs when *logico-deductive reasoning* is integrated with other forms of automated reasoning, such as *probabilistic reasoning* and *statistical inference*. This leads to the integration of deduction within *intelligent systems*, such as *decision support systems*, *agent programming environments*, and *data processing systems*. Here deduction may provide explanation, course of action, and the capability of learning from missing information; it may also aid modelling and facilitate agent communication.

The seminar on *Integrated Deduction* successfully covered as many as possible of these integration issues, including:

- Integration of deductive engines into more general automated deductive systems;
- Integration of automated deductive systems into interactive proof assistants;
- Integration of deduction into formal methods tools;
- Integration of deduction for knowledge processing; and
- Integration of deduction into intelligent systems such as agent-based systems.

Furthemore, the seminar investigated a number of key technological and human-related issues, that are largely orthogonal to most integration contexts, affecting both feasibility and deployment of integrated deduction. Examples of such issues are:

- The development of interfaces for integration;

[1] SAT solvers are solvers for satisfiability queries in propositional logic, known as the SAT problem.
[2] SMT stands for satisfiability modulo theories.

- The generation of continuous feedback during the run of deductive tools, including also information from intermediate or unsuccessful states;
- The reproducibility of results in the presence of tool updates or imposed resource limits (e.g., available computation time or memory) that may introduce non-determinism; and
- Advanced tradeoff's between performance and expressivity as well as between specialization and genericity.

Practical challenges around integrated deductive systems, including collaboration with non-expert users or access to data sets, were also discussed.

The seminar brought together a diverse audience, including both researchers working in deduction and researchers working in neighbouring areas that make use of deduction. Many participants have experience in building, using, and applying systems with integrated deduction.

Following the tradition of the Dagstuhl *Seminars on Deduction*, most of the program consisted of contributed talks by participants on their research. In this manner, the bottom-up style of the Dagstuhl experience was preserved, allowing for spontaneous contributions as they emerged during the seminar.

However, this seminar was also innovative with respect to tradition, in that it featured *five invited tutorials* on key topics in integrated deduction. These tutorials were valuable in highlighting the state-of-the-art in the integration of deduction systems and in fomenting discussions on challenges and open problems.

The program also featured a hike in the woods and a social dinner in a nearby village, that helped establishing or strengthtening collaborations.

The following section contains the abstracts for most of the talks and tutorials listed in alphabetical order.

2 Table of Contents

3 Overview of Talks

3.1 Higher-order superposition in action (Tutorial)

Alexander Bentkamp (VU University Amsterdam, NL), Jasmin Christian Blanchette (VU University Amsterdam, NL), and Sophie Tourret (INRIA Nancy – Grand Est – Villers-lès-Nancy, FR & MPI für Informatik – Saarbrücken, DE)

Joint work of Alexander Bentkamp, Jasmin Blanchette, Simon Cruanes, Sophie Tourret, Petar Vukmirovic, Uwe Waldmann
Main reference Alexander Bentkamp, Jasmin Blanchette, Sophie Tourret, Petar Vukmirovic: "Superposition for Full Higher-order Logic", in Proc. of the Automated Deduction – CADE 28 – 28th International Conference on Automated Deduction, Virtual Event, July 12-15, 2021, Proceedings, Lecture Notes in Computer Science, Vol. 12699, pp. 396–412, Springer, 2021.
URL https://doi.org/10.1007/978-3-030-79876-5_23

Following Blanchette's talk about higher-order superposition, this tutorial delved into higher-order logic topics (syntax, semantics, unification problem), the Zipperposition prover, and the lambda-superposition calculus. To clarify the fine points of the calculus, we ran Zipperposition on actual problems and studied the generated proof diagrams.

3.2 Integrating Optimization Solvers into Proof Assistants

Alexander Bentkamp (VU University Amsterdam, NL)

Joint work of Alexander Bentkamp, Jeremy Avigad
Main reference Alexander Bentkamp, Jeremy Avigad: "Verified Optimization", CoRR, Vol. abs/2111.06807, 2021.
URL https://arxiv.org/abs/2111.06807

Optimization is used extensively in engineering, industry, and finance, and various methods are used to transform problems to the point where they are amenable to solution by numerical methods. I present progress towards developing a framework, based on the Lean interactive proof assistant, for designing and applying such reductions in reliable and flexible ways.

3.3 Integrating higher-order reasoning into superposition

Jasmin Christian Blanchette (VU University Amsterdam, NL)

Joint work of Alexander Bentkamp, Jasmin Blanchette, Simon Cruanes, Sophie Tourret, Petar Vukmirovic, Uwe Waldmann
Main reference Alexander Bentkamp, Jasmin Blanchette, Sophie Tourret, Petar Vukmirovic: "Superposition for Full Higher-order Logic", in Proc. of the Automated Deduction – CADE 28 – 28th International Conference on Automated Deduction, Virtual Event, July 12-15, 2021, Proceedings, Lecture Notes in Computer Science, Vol. 12699, pp. 396–412, Springer, 2021.
URL https://doi.org/10.1007/978-3-030-79876-5_23

I described our journey, in the past five years, from first-order to higher-order superposition, focusing on the key design decisions, including our focus on a graceful generalization and on refutational completeness. I presented the three main milestones along the way and hinted at some ongoing work to optimize the calculus further.

3.4 Semantically-guided goal-sensitive reasoning: theorem proving and decision procedures (Tutorial)

Maria Paola Bonacina (Università degli Studi di Verona, IT)

Joint work of Maria Paola Bonacina, David A. Plaisted, Sarah Winkler
Main reference Maria Paola Bonacina, Sarah Winkler: "SGGS Decision Procedures", in Proc. of the Automated Reasoning – 10th International Joint Conference, IJCAR 2020, Paris, France, July 1-4, 2020, Proceedings, Part I, Lecture Notes in Computer Science, Vol. 12166, pp. 356–374, Springer, 2020.
URL https://doi.org/10.1007/978-3-030-51074-9_20

SGGS (Semantically-Guided Goal-Sensitive reasoning) is an attractive theorem-proving method for decision procedures, because it generalizes the Conflict-Driven Clause Learning (CDCL) procedure for propositional satisfiability, and it is model-complete in the limit, so that SGGS decision procedures are model-constructing. After summarizing the foundations of SGGS as a theorem-proving method, this tutorial presents recent and ongoing work on SGGS decision procedures for fragments of first-order logic. This includes both negative and positive results about known decidable fragments: for example, SGGS decides the stratified fragment, and hence Effectively PRopositional logic (EPR). SGGS also allows us to discover several new decidable fragments based on well-founded orderings. For most of these new fragments the small model property holds, as the cardinality of SGGS-generated models can be upper bounded, and membership can be tested by applying termination tools for rewriting. A report on experiments with the prototype theorem prover Koala, which is the first implementation of SGGS, closes the presentation.

(SGGS is joint work with David Plaisted; SGGS decision procedures are joint work with Sarah Winkler, who is the author of Koala).

3.5 Proofs in SMT (Tutorial)

Pascal Fontaine (University of Liège, BE)

Joint work of Hans-Jörg Schurr, Mathias Fleury, Haniel Barbosa, Pascal Fontaine
Main reference Hans-Jörg Schurr, Mathias Fleury, Haniel Barbosa, Pascal Fontaine: "Alethe: Towards a Generic SMT Proof Format (extended abstract)", in Proc. of the Proceedings Seventh Workshop on Proof eXchange for Theorem Proving, PxTP 2021, Pittsburg, PA, USA, July 11, 2021, EPTCS, Vol. 336, pp. 49–54, 2021.
URL https://doi.org/10.4204/EPTCS.336.6

Proofs have an increasing importance for automated reasoning, and in particular for integration of deduction engines. In this talk, we present our efforts to produce detailed, checkable proofs in the context of SMT solving. This includes producing proofs for the underlying SAT solver, the various theories, quantifier instantiation and formula processing. We conclude by a short introduction to the Alethe concrete format, an attempt at a versatile, easy to use proof format, in the philosophy of the SMT-LIB format.

This talk mentions the work of many, including Haniel Barbosa, Jasmin Blanchette, Mathias Fleury, and Hans-Jörg Schurr.

References

1 Haniel Barbosa, Jasmin Christian Blanchette, and Pascal Fontaine. Scalable fine-grained proofs for formula processing. In Leonardo de Moura, editor, *Automated Deduction – CADE*

26 - 26th International Conference on Automated Deduction, Gothenburg, Sweden, August 6-11, 2017, Proceedings, volume 10395 of *Lecture Notes in Computer Science*, pages 398–412. Springer, 2017.

2 Hans-Jörg Schurr, Mathias Fleury, Haniel Barbosa, and Pascal Fontaine. Alethe: Towards a generic SMT proof format (extended abstract). *CoRR*, abs/2107.02354, 2021.

3 Hans-Jörg Schurr, Mathias Fleury, and Martin Desharnais. Reliable reconstruction of fine-grained proofs in a proof assistant. In André Platzer and Geoff Sutcliffe, editors, *Automated Deduction – CADE 28 – 28th International Conference on Automated Deduction, Virtual Event, July 12-15, 2021, Proceedings*, volume 12699 of *Lecture Notes in Computer Science*, pages 450–467. Springer, 2021.

3.6 AProVE as a Platform for Integrated Deduction

Carsten Fuhs (Birkbeck, University of London, GB)

Joint work of Thaïs Baudon, Jürgen Giesl, Cornelius Aschermann, Marc Brockschmidt, Fabian Emmes, Florian Frohn, Carsten Fuhs, Jera Hensel, Carsten Otto, Martin Plücker, Peter Schneider-Kamp, Thomas Ströder, Stephanie Swiderski, René Thiemann

Main reference Jürgen Giesl, Cornelius Aschermann, Marc Brockschmidt, Fabian Emmes, Florian Frohn, Carsten Fuhs, Jera Hensel, Carsten Otto, Martin Plücker, Peter Schneider-Kamp, Thomas Ströder, Stephanie Swiderski, René Thiemann: "Analyzing Program Termination and Complexity Automatically with AProVE", J. Autom. Reason., Vol. 58(1), pp. 3–31, 2017.

URL https://doi.org/10.1007/s10817-016-9388-y

This talk addresses the following points:

- How does AProVE use other deduction tools? (SAT/SMT solvers, tools for termination and complexity analysis for specific formats, compilers, proof checkers, ...)
- How do other deduction tools use AProVE? (Proof checkers, Knuth-Bendix completion tools, higher-order termination analysis tools, ...)
- Work in Progress: Using Complexity Bounds for Automated Scheduling (joint work with Thaïs Baudon and Laure Gonnord)

References

1 Jürgen Giesl, Cornelius Aschermann, Marc Brockschmidt, Fabian Emmes, Florian Frohn, Carsten Fuhs, Jera Hensel, Carsten Otto, Martin Plücker, Peter Schneider-Kamp, Thomas Ströder, Stephanie Swiderski, René Thiemann. *Analyzing Program Termination and Complexity Automatically with AProVE*. Journal of Automated Reasoning 58(1):3-31, 2017. `https://doi.org/10.1007/s10817-016-9388-y`

2 Thaïs Baudon, Carsten Fuhs, Laure Gonnord. *Parallel Complexity of Term Rewriting Systems*. In Proc. 17th Workshop on Termination (WST'21), pages 39 – 44, 2021. `https://costa.fdi.ucm.es/wst2021/WST2021_proceedings.pdf`, `https://hal.archives-ouvertes.fr/hal-03418400`

3.7 From MCSAT to CDSAT and beyond

Stéphane Graham-Lengrand (SRI – Menlo Park, US)

Joint work of Stéphane Graham-Lengrand, Maria Paola Bonacina, Natarajan Shankar, Dejan Jovanovic, Bruno Dutertre
Main reference Beniamino Accattoli, Stéphane Graham-Lengrand, Delia Kesner: "Tight typings and split bounds, fully developed", J. Funct. Program., Vol. 30, p. e14, 2020.
URL https://doi.org/10.1017/S095679682000012X

We present the model-constructing satisfiability approach (MCSAT) to SMT-solving, and illustrate it with the theory of linear arithmetic. We then describe an abstract framework for integrating multiple reasoning modules, called CDSAT for Conflict-Driven Satisfiability, which in particular generalizes MCSAT, CDCL(T), and the equality sharing scheme for disjoint theory combination. CDSAT comes with soundness, completeness, and termination results based on the individual reasoners satisfying appropriate conditions. We discuss proof production for the UNSAT answers of CDSAT. We also present a new algorithm that leverages conflict-driven reasoning to solve quantified satisfiability problems for complete theories; this was implemented in the form of the YicesQS solver, which entered the SMT competition in the BV and NRA logics.

3.8 Efficient local reductions to basic modal logic

Ullrich Hustadt (University of Liverpool, GB)

Joint work of Ullrich Hustadt, Fabio Papacchini, Cláudia Nalon, Clare Dixon
Main reference Fabio Papacchini, Cláudia Nalon, Ullrich Hustadt, Clare Dixon: "Efficient Local Reductions to Basic Modal Logic", in Proc. of the Automated Deduction – CADE 28 – 28th International Conference on Automated Deduction, Virtual Event, July 12-15, 2021, Proceedings, Lecture Notes in Computer Science, Vol. 12699, pp. 76–92, Springer, 2021.
URL https://doi.org/10.1007/978-3-030-79876-5_5

We present the model-constructing satisfiability approach (MCSAT) to SMT-solving, and illustrate it with the theory of linear arithmetic. We then describe an abstract framework for integrating multiple reasoning modules, called CDSAT for Conflict-Driven Satisfiability, which in particular generalizes MCSAT, CDCL(T), and the equality sharing scheme for disjoint theory combination. CDSAT comes with soundness, completeness, and termination results based on the individual reasoners satisfying appropriate conditions. We discuss proof production for the UNSAT answers of CDSAT. We also present a new algorithm that leverages conflict-driven reasoning to solve quantified satisfiability problems for complete theories; this was implemented in the form of the YicesQS solver, which entered the SMT competition in the BV and NRA logics.

3.9 Conjecture Synthesis, Lemma Discovery and Reasoning

Moa Johansson (Chalmers University of Technology – Göteborg, SE)

Joint work of Moa Johansson, Solrun Halla Einarsdottir, Nicholas Smallbone
Main reference Moa Johansson, Nicholas Smallbone: "Conjectures, Tests and Proofs: An Overview of Theory Exploration", in Proc. of the 9th International Workshop on Verification and Program Transformation, VPT@ETAPS 2021, Luxembourg, Luxembourg, 27th and 28th of March 2021, EPTCS, Vol. 341, pp. 1–16, 2021.
URL https://doi.org/10.4204/EPTCS.341.1

Formulating interesting novel conjectures about a new problem domain is a key component of mathematical reasoning. How could this be done by a machine? It is neither a purely deductive problem, nor is it easily solved by data driven machine learning methods. Theory exploration is a technique which tries to address this problem, by combining heuristic search over possible terms, with automated testing to evaluate terms and form equational conjectures. A key for tractability is to start with smaller and more general terms, and exclude any terms which can be reduced by already discovered ones. We demonstrate the QuickSpec system for conjecture generation, and its combination with several theorem provers through the TIP-interface.

References

1 Solrun Halla Einarsdottir, Nicholas Smallbone and Moa Johansson. *Template-based Theory Exploration: Discovering Properties of Functional Programs by Testing*. Proceedings of IFL'20, 2021.
2 Moa Johansson, Koen Claesson and Maximilian Algehed. *Quick Specifications for the Busy Programmer*. Nicholas Smallbone, Journal of Functional Programming, 2017.
3 Moa Johansson and Nicholas Smallbone. *Conjectures, Tests and Proofs: An Overview of Theory Exploration*. Invited paper, Proceedings of 9th Workshop on Verification and Program Transformation, 2021.
4 Moa Johansson. *Automated Theory Exploration for Interactive Theorem Proving*. Conference on Interactive Theorem Proving (ITP), p. 1-11, 2017.

3.10 Using Deduction within Methods for Non-Standard Reasoning in Description Logics

Patrick Koopmann (TU Dresden, DE)

Joint work of Patrick Koopmann, Warren Del-Pinto, Sophie Tourret, Renate A. Schmidt, Fajar Haifani, Franz Baader, Francesco Kriegel, Adrian Nuradiansyah
Main reference Patrick Koopmann, Warren Del-Pinto, Sophie Tourret, Renate A. Schmidt: "Signature-Based Abduction for Expressive Description Logics", in Proc. of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020, Rhodes, Greece, September 12-18, 2020, pp. 592–602, 2020.
URL https://doi.org/10.24963/kr.2020/59

Central reasoning tasks for description logic ontologies such as consistency testing, subsumption checking and classification require deductive reasoning, in the sense that logical consequences from an ontology have to decided or computed. In addition to these standard-reasoning tasks, non-deductive reasoning is useful for many applications of ontologies such as in diagnosis, privacy, as well as for debugging and managing large complex ontologies.

Examples for those non-standard reasoning tasks include abduction, repair, module extraction and forgetting, but also proof generation in the absence of a reasoning calculus. While not directly deduction problems, methods for solving these tasks often use deduction internally. We look at a class of such methods where deduction is used to saturate a set of sentences. That means, in order to solve a non-standard reasoning task, we first translate part of the problem into an appropriate logic, compute all entailments within some class of logical sentences, and then use the saturated set of sentences to compute the solution to our problem. Often, the challenge is to define such a class of sentences that is both bounded by the input and sufficient for constructing the solution, since otherwise our method would either not terminate or produce an incomplete solution. We illustrate this by presenting three examples of such methods, two for solving variants of abduction in description logics [1, 2], and one for ABox repair and anonymization [3], including both published results and current research, which integrate different deduction systems such as the theorem prover SPASS, the datalog engine VLog, as well as a newly developed calculus dedicated to the problem at hand.

References

1 Patrick Koopmann, Warren Del-Pinto, Sophie Tourret, Renate A. Schmidt. *Signature-Based Abduction for Expressive Description Logics*. Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020, pages 592–602, 2020. https://doi.org/10.24963/kr.2020/59

2 Fajar Haifani, Patrick Koopmann, Sophie Tourret. *Abduction in $\mathcal{EL}$ via Translation to FOL*. Proceedings of the Second Workshop on Second-Order Quantifier Elimination and Related Topics (SOQE 2021) associated with the 18th International Conference on Principles of Knowledge Representation and Reasoning (KR 2021), pages 46–58, 2021. http://ceur-ws.org/Vol-3009/paper2.pdf

3 Franz Baader, Patrick Koopmann, Francesco Kriegel, Adrian Nuradiansyah. *Computing Optimal Repairs of Quantified ABoxes w.r.t. Static $\mathcal{EL}$ TBoxes*. Automated Deduction – CADE 28 – 28th International Conference on Automated Deduction, pages 309–326, 2021. https://doi.org/10.1007/978-3-030-79876-5_18

3.11 Verified Proof Checkers

Magnus Myreen (Chalmers University of Technology – Göteborg, SE)

Joint work of Jared Davis, Yong Kiam Tan, and many others

Main reference Yong Kiam Tan, Marijn J. H. Heule, Magnus O. Myreen: "cake_lpr: Verified Propagation Redundancy Checking in CakeML", in Proc. of the Tools and Algorithms for the Construction and Analysis of Systems – 27th International Conference, TACAS 2021, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 – April 1, 2021, Proceedings, Part II, Lecture Notes in Computer Science, Vol. 12652, pp. 223–241, Springer, 2021.

URL https://doi.org/10.1007/978-3-030-72013-1_12

In this talk, I described my work on program verification and particularly my focus on proving functional correctness down to the machine code that runs the programs, including for proof checkers. I have worked on and have supervised work on several checkers, but only talked about (1) my old work on proving end-to-end correctness of Jared Davis's Milawa prover, and (2) recent work on a LPR/LRAT checker for UNSAT proofs. My talk included a description of the CakeML project, which was the context of (2). I used demos to show what the tools look like when running.

References

1 Yong Kiam Tan, Marijn J. H. Heule, and Magnus O. Myreen. `cake_lpr`: *Verified Propagation Redundancy Checking in CakeML*. Tools and Algorithms for the Construction and Analysis of Systems (TACAS), 2021.
2 Jared Davis, Magnus O. Myreen.. *The self-verifying Milawa theorem prover is sound (down to the machine code that runs it)*. Journal of Automated Reasoning (JAR), 2015.
3 Yong Kiam Tan, Magnus O. Myreen, Ramana Kumar, Anthony Fox, Scott Owens, and Michael Norrish. *The verified CakeML compiler backend*. Journal of Functional Programming (JFP), 2019.

3.12 On reasoning about multisets (Tutorial)

Ruzica Piskac (Yale University – New Haven, US)

When reasoning about container data structures that can hold duplicate elements, multisets are the obvious choice for representing the data structure abstractly. However, the decidability and complexity of constraints on multisets has been much less studied and understood than for constraints on sets. In this presentation, we outline an efficient decision procedure for reasoning about multisets with cardinality constraints. We describe how to translate, in linear time, multisets constraints to constraints in an extension of quantifier-free linear integer arithmetic, which we call LIA*. LIA* extends linear integer arithmetic with unbounded sums over values satisfying a given linear arithmetic formula. We show how to reduce a LIA* formula to an equisatisfiable linear integer arithmetic formula. However, this approach requires an explicit computation of semilinear sets and in practice it scales poorly even on simple benchmarks. We then describe a recent more efficient approach for checking satisfiability of LIA*. The approach is based on the use of under- and over-approximations of LIA* formulas. This way we avoid the space overhead and explicitly computing semilinear sets. Finally, we report on our prototype tool which can efficiently reason about sets and multisets formulas with cardinality constraints.

3.13 Constrained Horn Clauses in Verification: 11 Years later

Philipp Rümmer (Uppsala University, SE)

Constrained Horn Clauses (CHC) have over the last decade emerged as a uniform framework for reasoning about different aspects of software safety. CHCs form a fragment of first-order logic, modulo various background theories, in which models can be constructed effectively with the help of model checking algorithms. In the talk I have given an overview of CHCs, including their use in program verification and the recently established competition CHC-COMP [2, 1]. I have then presented some of our work on the development of CHC solvers that can handle relevant theories such as integers, bit-vectors, and ADTs [3], and outlined challenges remaining in the area.

References

1 Grigory Fedyukovich and Philipp Rümmer. Competition report: CHC-COMP-21. *CoRR*, abs/2109.04635, 2021.

2 Philipp Rümmer. Competition report: CHC-COMP-20. In Laurent Fribourg and Matthias Heizmann, editors, *Proceedings 8th International Workshop on Verification and Program Transformation and 7th Workshop on Horn Clauses for Verification and Synthesis, VPT/HCVS@ETAPS 2020, Dublin, Ireland, 25-26th April 2020*, volume 320 of *EPTCS*, pages 197–219, 2020.

3 Hossein Hojjat and Philipp Rümmer. The ELDARICA Horn solver. In Nikolaj Bjørner and Arie Gurfinkel, editors, *2018 Formal Methods in Computer Aided Design, FMCAD 2018, Austin, TX, USA, October 30 - November 2, 2018*, pages 1–7. IEEE, 2018.

3.14 Development and integration of deduction for the medical ontology SNOMED CT

Renate Schmidt (University of Manchester, GB)

Joint work of Warren Del-Pinto, Renate A. Schmidt, Yongshen Gao, Ghadah Alghamdi

In my presentation I talked about our experiences and ongoing work in a Partnership with SNOMED Intl to develop bespoke content extraction software for the medical ontology SNOMED CT. SNOMED CT is a large knowledge base of standardised, precise definitions of clinical terms and medical codes for use in health care systems in many countries. It has long been an aim to have the capability to compute smaller self-contained extracts of the ontology that are restricted to a narrow focus, for example, kidney diseases, dentistry or nursing practice. Such subontologies would make it easier to reuse and share content, to assist with new ontology creation, quality assurance, ontology update and debugging. In addition, reasoning tasks such as querying and classification take less time to execute over a smaller extract than over the original ontology. Our subontology builder takes as input a set of focus concepts, which could be a reference set that the user is interested in, and generate a subontology that can be used in place of SNOMED CT in a specific application. The idea is that the subontology equivalently captures the semantics of these focus concepts and their relationship to other concepts in a certain abstracted form. Because the subtype relationships between concepts are so important for SNOMED enabled search in patient data, a further requirement is that, the concept hierarchy over the focus and supporting concepts in SNOMED CT must be included in the subontology. Subontologies computed for a collection of standard lists of clinical concepts and reference sets for specialities can be viewed in the new subontology browser here:

`https://iaa.snomed.tools`

and compared with SNOMED CT here:

`https://browser.ihtsdotools.org/`

The research was undertaken in EPSRC IAA Project 290 funded by the UK EPSRC, the University of Manchester and IHTSDO.

3.15 Tackling Commonsense Reasoning Problems with a First-Order Logic Reasoner

Claudia Schon (Universität Koblenz-Landau, DE)

Joint work of Ulrich Furbach, Teresa Krämer, Claudia Schon
Main reference Ulrich Furbach, Teresa Krämer, Claudia Schon: "Names Are Not Just Sound and Smoke: Word Embeddings for Axiom Selection", in Proc. of the Automated Deduction – CADE 27 – 27th International Conference on Automated Deduction, Natal, Brazil, August 27-30, 2019, Proceedings, Lecture Notes in Computer Science, Vol. 11716, pp. 250–268, Springer, 2019.
URL https://doi.org/10.1007/978-3-030-29436-6_15

This talk reports on our experiences using a first-order logic reasoner to solve commonsense reasoning benchmark problems like the Choice of Plausible Alternatives (COPA) Challenge. Most approaches in this area rely on pre-trained language models and do not use reasoning. In contrast, we combine a deductive theorem prover with large background knowledge bases and machine learning. Since these background knowledge bases represent a very large amount of knowledge from a wide variety of domains, they cannot be used by the reasoner as a whole. Selection methods are used to select suitable background knowledge for a specific task. In the area of commonsense reasoning, these selection methods can benefit from the integration of statistical techniques such as word embeddings. We show that incorporating word embeddings into the selection process enables the selection of background knowledge that is thematically appropriate to a commonsense reasoning task. This approach is implemented and we present experimental results.

3.16 Deduction as a Service

Stephan Schulz (Duale Hochschule Baden-Württemberg – Stuttgart, DE)

Joint work of Mohamed Hassona, Stephan Schulz
Main reference Mohamed Hassona, Stephan Schulz: "Deduction as a Service", in Proc. of the 5th Workshop on Practical Aspects of Automated Reasoning co-located with International Joint Conference on Automated Reasoning (IJCAR 2016), Coimbra, Portugal, July 2nd, 2016, CEUR Workshop Proceedings, Vol. 1635, pp. 32–40, CEUR-WS.org, 2016.
URL http://ceur-ws.org/Vol-1635/paper-04.pdf

Traditionally, problems for automated theorem provers were small, tightly specified, and often with long, complicated proofs. In contrast, many more recent problems are automatically generated from large artefacts, or posed over large common-sense or mathematical knowledge bases, but often with rather simple and short proofs.

In these cases, the total time for a proof attempt is often dominated by the overhead of parsing and premise selection. Offering deduction over a large, persistent knowledge base as a service can amortise this overhead, reducing the time of single proof attempts to a level acceptable even for interactive use.

I describe the architecture of such a system, some of the practical challenges, and the state of the art so far.

3.17 Integrating Machine Learning into Saturation-based ATPs (Tutorial)

Martin Suda (Czech Technical University – Prague, CZ)

Main reference Martin Suda: "Improving ENIGMA-style Clause Selection while Learning From History", in Proc. of the Automated Deduction – CADE 28 – 28th International Conference on Automated Deduction, Virtual Event, July 12-15, 2021, Proceedings, Lecture Notes in Computer Science, Vol. 12699, pp. 543–561, Springer, 2021.
URL https://doi.org/10.1007/978-3-030-79876-5_31

Applying the techniques of machine learning (ML) promises to dramatically improve the performance of modern automatic theorem provers (ATPs) and thus to positively impact their applications. The most successful avenue in this direction explored so far is machine-learned clause selection guidance, where we learn to recognize and prefer for selection clauses that look like those that contributed to a proof in past successful runs. In this talk I present Deepire, an extension of the ATP Vampire where clause selection is guided by a recursive neural network (RvNN) for classifying clauses based solely on their derivation history.

3.18 On what is wrong with higher-order SMT and what we are doing to fix it

Sophie Tourret (INRIA Nancy – Grand Est – Villers-lès-Nancy, FR & MPI für Informatik – Saarbrücken, DE)

Joint work of Haniel Barbosa, Daniel El-Ouraoui, Pascal Fontaine, Sophie Tourret

Recent work has extended ground SMT solvers to higher-order logic (HOL), but SMT solving has yet to show its full power in HOL. It remains to lift quantifier instantiation algorithms to perform higher-order unification. As a consequence, widely used instantiation techniques, such as trigger- and particularly conflict-based instantiation, can only be applied in a limited manner. Congruence closure with free variables (CCFV) is a decision procedure for the E-ground (dis-)unification problem, which is at the heart of these instantiation techniques. Here, as a first step towards fully supporting trigger- and conflict-based instantiation in HOL, we define the E-ground (dis-)unification problem in λ-free higher-order logic (λfHOL), an extension of first-order logic where function symbols may be partially applied and functional variables may occur, and extend CCFV to solve it. To improve scalability in the context of handling higher-order variables, we rely on a SAT encoding of the CCFV search. We present a solution reconstruction procedure so that the propositional models lead to solutions for the respective E-ground (dis-)unification problem. This is instrumental to fully port trigger- and conflict-based instantiation to be fully applied in λfHOL.

3.19 Linear-Time Verification of Data-Aware Dynamic Systems with Arithmetic

Sarah Winkler (University of Verona, IT)

Joint work of Sarah Winkler, Paolo Felli, Marco Montali

Counter systems have been investigated in formal methods, database theory, and AI. Though model checking of such systems is undecidable since two-counter machines can be simulated, different decidable classes have been discovered by restricting the constraint language and/or the control flow. This talk presents a new, abstract criterion for the decidability of linear-time verification of such systems, called finite summary, which guarantees the existence of a faithful finite-state abstraction. We demonstrate that several decidability conditions studied in formal methods and database theory can be seen as concrete, checkable instances of this property. To this end, we exploit results from SMT, and automated reasoning in general. Finally, we show how the finite summary property leads to modularity results: a system enjoys finite summary if it can be partitioned appropriately into smaller systems that possess the property. Our results allow us to analyze systems that cannot be handled by earlier approaches.

Participants

- Erika Abraham
RWTH Aachen University, DE
- Alexander Bentkamp
VU University Amsterdam, NL
- Jasmin Christian Blanchette
VU University Amsterdam, NL
- Maria Paola Bonacina
University of Verona, IT
- Pascal Fontaine
University of Liège, BE
- Carsten Fuhs
Birkbeck, University of GLondon, GB
- Stéphane Graham-Lengrand
SRI – Menlo Park, US
- Ullrich Hustadt
University of Liverpool, GB
- Moa Johansson
Chalmers University of Technology – Göteborg, SE
- Patrick Koopmann
TU Dresden, DE
- Magnus Myreen
Chalmers University of Technology – Göteborg, SE
- Ruzica Piskac
Yale University – New Haven, US
- Philipp Rümmer
Uppsala University, SE
- Renate Schmidt
University of Manchester, GB
- Claudia Schon
Universität Koblenz-Landau, DE
- Stephan Schulz
Duale Hochschule Baden-Württemberg – Stuttgart, DE
- Alexander Steen
University of Luxembourg, LU
- Martin Suda
Czech Technical University – Prague, CZ
- Sophie Tourret
INRIA Nancy – Grand Est – Villers-lès-Nancy, FR & MPI für Informatik – Saarbrücken, DE
- Sarah Winkler
University of Verona, IT