05431 Abstracts Collection
Deduction and Applications
— Dagstuhl Seminar —

Franz Baader[1], Peter Baumgartner[2], Robert Nieuwenhuis[3] and Andrei Voronkov[4]

[1] TU Dresden, DE
baader@inf.tu-dresden.de
[2] MPI für Informatik, DE
baumgart@mpi-sb.mpg.de
[3] TU Barcelona, ES
roberto@lsi.upc.es
[4] Manchester Univ., GB
voronkov@cs.man.ac.uk

Abstract. From 23.10.05 to 28.10.05, the Dagstuhl Seminar 05431 "Deduction and Applications" was held in the International Conference and Research Center (IBFI), Schloss Dagstuhl. During the seminar, several participants presented their current research, and ongoing work and open problems were discussed. Abstracts of the presentations given during the seminar as well as abstracts of seminar results and ideas are put together in this paper. The first section describes the seminar topics and goals in general. Links to extended abstracts or full papers are provided, if available.

Keywords. Formal logic, deduction, artificial intelligence

05431 Executive Summary – Deduction and Applications

Peter Baumgartner (MPI für Informatik, D)

Formal logic provides a mathematical foundation for many areas of computer science. Logical languages are used as specification language within, e.g., program development and verification, hardware design and verification, relational databases, and many subfields of Artificial Intelligence. Automated Deduction is concerned with the design and implementation of algorithms based on logical deduction for solving problems in these areas.

The last years have seen considerable improvements concerning both basic automated deduction technology and its (real-world) applications. Accordingly, the goal of the seminar was to bring together researchers from both sides in order to get an overview of the state of the art, and also to get ideas how to advance automated deduction from an application oriented point of view.

Keywords: Formal logic, deduction, artificial intelligence

Joint work of: Baader, Franz; Baumgartner, Peter; Nieuwenhuis, Robert; Voronkov, Andrei

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2006/510

A description logic voyage: from inexpressive to expressive languages and back

Franz Baader (TU Dresden, D)

The talk will take you on an adventurous research voyage that starts in the 'swamp of no-semantics' of early knowledge representation systems, regains firmer ground with the first logic-based languages, but then has to cross the 'tractability barrier' towards expressive Description Logics with sound and complete, but intractable inference procedures.

Finally, we will reach the 'island of expressiveness', on which expressive Description Logics provide us with ontology languages for the Semantic Web and practical reasoning tools for such languages. On the way, we will receive a brief introduction into Description Logics, and meet hermaphrodites, OWLs, Lolita, and Chinese parents. But isn't there a smaller, more beautiful island of tractable Description Logics just over the horizon?

Keywords: Description logic, ontology, semantic web

Full Paper:
http://lat.inf.tu-dresden.de/research/papers/2005/BaaderBrandtLutz-IJCAI-05.pdf

See also: F. Baader, S. Brandt, and C. Lutz. Pushing the EL Envelope. In Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence IJCAI-05, Edinburgh, UK, 2005. Morgan-Kaufmann Publishers.

Resolution proofs, analogical reasoning and juridical logic

Matthias Baaz (TU Wien, A)

The logic of legal reasoning is determined by the seemingly contradictory principles that (i) legal arguments should be demonstrably sound, and (ii) decisions have to be achieved (even within a priori limited time and space). We show that arguments in both maximalist (continental/Latin law) and minimalist (common law) legal reasoning systems are consequently based on analogical reasoning. In this lecture we demonstrate how Resolution Calculus can be extended to provide a formal basis for analogical reasoning of the kind described and in consequence for juridical logic and e-justice.

Keywords: Analogical reasoning, resolution calculus, proof descriptions, juridical logic

See also: Matthias Baaz: Note on Formal Analogical Reasoning in the Juridical Context in: LNCS Volume 3634 / 2005, Springer, Computer Science Logic: 19th International Workshop, CSL 2005, 14th Annual Conference of the EACSL, Oxford, UK, August 22-25, 2005. Proceedings Editors: Luke Ong.

DPLL(T) with Generalized Theory Propagation

Clark W. Barrett (Courant Institute - New York, USA)

The DPLL(T) calculus provides an abstract formalization for reasoning about an efficient combination of Boolean and Theory reasoning. Previous versions of DPLL(T) require that a theory be able to determine whether or not a set of literals is satisfiable. However, it is often more convenient for a theory to answer "maybe" and provide some additional clauses, possibly including new variables and literals, to the DPLL solver. In this talk we discuss preliminary work on the extensions necessary to accommodate this and give examples of its application to a fragment of set theory and the theory of arrays.

Keywords: DPLL, DPLL(T), SMT

Second-Order Principles in Specification Languages for Object-Oriented Programs

Bernhard Beckert (Univ. Koblenz-Landau, D)

Within the setting of object-oriented program specification and verification, pointers and object references can be considered as relations between the elements of a data structure. When we specify properties of these data structures, we often describe properties of relations. Hence it is important to be able to talk about relations and their properties when specifying object-oriented programs or programs with pointers. Many interesting properties of relations such as transitive closure, finiteness, and generatedness are not expressible in first-order logic (FOL); hence neither are they expressible in first-order fragments of specification languages. In this paper we give an overview of the different ways such properties can be expressed in various logics, with a particular emphasis on extensions of FOL, i.e. transitive closure logic, fixed-point logic, and first-order dynamic logic. Within the paper we also discuss which of these extensions already are - or in fact should be - implemented within specification languages. We feel that such a discussion is necessary since it is often the case that when an extension of FOL is implemented within a specification language it is done so in an ad hoc manner.

Keywords: Software specification, higher-order logic, first-order logic, object-oriented programming

Joint work of: Beckert, Bernhard; Trentelman, Kerry

See also: Bernhard Beckert and Kerry Trentelman. Second-Order Principles in Specification Languages for Object-Oriented Programs. In Proceedings of the 12th International Conference on Logic for Programming, Artificial Intelligence and Reasoning, Montego Bay, Jamaica, pages 154-168. Springer, 2005.

A Structured Set of Higher-Order Problems

Christoph Benzmüller (Universität Saarbrücken, D)

We present a set of problems that may support the development of calculi and theorem provers for classical higher-order logic. We propose to employ these test problems as quick and easy criteria preceding the formal soundness and completeness analysis of proof systems under development. Our set of problems is structured according to different technical issues and along different notions of semantics (including Henkin semantics) for higher-order logic. Many examples are either theorems or non-theorems depending on the choice of semantics. The examples can thus indicate the deductive strength of a proof system.

Keywords: Higher Order Logic, Semantics, Extensionality, Proof Problems

Joint work of: Benzmüller, Christoph; Brown, Chad

Full Paper:
http://www.ags.uni-sb.de/ chris/papers/C17.pdf

See also: C. Benzmüller and C. Brown, A Structured Set of Higher-Order Problems. Proceedings of the 18th International Conference on Theorem Proving in Higher Order Logics (TPHOLs 2005), LNAI vol. 3606, pp. 66-81, Oxford, UK, 2005. Springer.

On Predicting the Grammar of a Normal-Form

Alan Bundy (University of Edinburgh, GB)

We introduce the problem of predicting the formal grammar of the normal-form that is generated from a class of expressions by exhaustive application of a set of rewrite rules. We describe and implement a sound but incomplete procedure for solving this problem and report on its theoretical and experimental properties.

Keywords: Rewrite rules, normal forms, context free grammar

Event B method

Dominique Cansell (LORIA - Nancy, F)

In this talk we present the event B method using an example: constructing a spanning tree of a connected graph. The first model is very abstract and computes the spanning tree in one shot. Refinement is used to introduce the depth-first algorithm. We give and justify (thanks to the refinement) some invariant and explain how we can prove it almost automatically.

Keywords: Modelisation, incremental development, abstraction, refinement, invariant, proof

Efficient Satisfiability Modulo Theories via Delayed Theory Combination

Alessandro Cimatti (ITC-irst - Trento, I)

The problem of deciding the satisfiability of a quantifier-free formula with respect to a background theory, also known as Satisfiability Modulo Theories (SMT), is gaining increasing relevance in verification: representation capabilities beyond propositional logic allow for a natural modeling of real-world problems (e.g., pipeline and RTL circuits verification, proof obligations in software systems).

In this paper, we focus on the case where the background theory is the combination $T_1 \cup T_2$ of two simpler theories. Many SMT procedures combine a boolean model enumeration with a decision procedure for $T_1 \cup T_2$, where conjunctions of literals can be decided by an integration schema such as Nelson-Oppen, via a structured exchange of interface formulae (e.g., equalities in the case of convex theories, disjunctions of equalities otherwise).

We propose a new approach for SMT($T_1 \cup T_2$), called Delayed Theory Combination, which does not require a decision procedure for $T_1 \cup T_2$, but only individual decision procedures for T_1 and T_2, which are directly integrated into the boolean model enumerator. This approach is much simpler and natural, allows each of the solvers to be implemented and optimized without taking into account the others, and it nicely encompasses the case of non-convex theories. We show the effectiveness of the approach by a thorough experimental comparison.

Keywords: SMT, decision procedure, delayed theory combination

Equinox – A Lazy Explicating Theorem Prover for Full Pure First-Order Logic

Koen Claessen (Chalmers UT - Göteborg, S)

I will describe the new theorem prover Equinox, for full first-order logic with equality.

Equinox is based on successively augmenting a propositional logic theorem prover with more and more expressivity, using the lazy explicating proof technique. Equinox is still in a baby-state but already shows promising results. I will discuss its current design, strengths and weaknesses as well as future directions.

Keywords: Automated theorem proving, first-order logic, SAT solving

Locality modulo

Hubert Comon-Lundh (ENS - Cachan, F)

Locality is a property of an inference system, which states that, if there is a proof of a formula, then there is a proof only involving "pieces" of the hypotheses and the conclusion. D. McAllester investigated local inference systems when a "piece" means a subterm. Basin and Ganzinger considered locality when a "piece" means a smaller formula. Local inference systems yield efficient proof search. As an example, most of the intruder models (in the analysis of cryprographic protocols) are local. We present an ongoing work on local inference systems in presence of algebraic properties.

Joint work of: Comon-Lundh, Hubert; Delaune, Stephanie

Practical Proof Checking for Program Certification

Bernd Fischer (NASA (RIACS) - Moffett Field, USA)

Program certification aims to provide explicit evidence that a program meets a specified level of safety. This evidence must be independently reproducible and verifiable. We have developed a system, based on theorem proving, that generates proofs that auto-generated aerospace code adheres to a number of safety policies. For certification purposes, these proofs need to be verified by a proof checker.

I will describe and evaluate a semantic derivation verification approach to proof checking. The evaluation is based on safety obligations that are attempted by EP and SPASS. Our system is able to verify almost all of the proofs found by the two provers. The majority of the proofs are checked completely in less than 15 seconds wall clock time. This shows that the proof checking task arising from a substantial prover application is practically tractable.

Keywords: Program certification, proof checking

Joint work of: Fischer, Bernd; Sutcliffe, Geoff; Denney, Ewen

Proving and Disproving Termination in the Dependency Pair Framework

Jürgen Giesl (RWTH Aachen, D)

The dependency pair framework is a new general concept to integrate arbitrary techniques for termination analysis of term rewriting.

In this way, the benefits of different techniques can be combined and their modularity and power are increased significantly. Moreover, this framework facilitates the development of new methods for termination analysis. Traditionally, the research on termination focused on methods which prove termination and there were hardly any approaches for disproving termination. We show that with the dependency pair framework, one can combine the search for a proof and for a disproof of termination. In this way, we obtain the first powerful method which can also verify non-termination of term rewrite systems.

We implemented and evaluated our contributions in the automated termination prover AProVE. Due to these results, AProVE was the winning tool in the International Competition of Termination Provers 2005, both for proving and for disproving termination of term rewriting.

Keywords: Termination, non-termination, term rewriting, dependency pairs

Joint work of: Giesl, Jürgen; Thiemann, René, Schneider-Kamp, Peter

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2006/509

Full Paper:
http://www-i2.informatik.rwth-aachen.de/giesl/papers/LPAR04-distribute.ps

Full Paper:
http://www-i2.informatik.rwth-aachen.de/giesl/papers/FROCOS05-distribute.ps

See also: J. Giesl, R. Thiemann, P. Schneider-Kamp. The Dependency Pair Framework: Combining Techniques for Automated Termination Proofs. Proceedings of the 11th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR 2004), Montevideo, Uruguay, Lecture Notes in Computer Science 3452, pages 301-331, 2005.

See also: J. Giesl, R. Thiemann, P. Schneider-Kamp. Proving and Disproving Termination of Higher-Order Functions. Proceedings of the 5th International Workshop on Frontiers of Combining Systems (FroCoS '05), Vienna, Austria, Lecture Notes in Artificial Intelligence 3717, pages 216-231, 2005.

Processor Datapath Verification with SPASS

Thomas Hillenbrand (MPI für Informatik, D)

Initiated by the VERISOFT project, we have carried out an attempt to verify the datapath of a simple DLX-like processor with the superposition-based theorem prover SPASS, which is not a straightforward choice for this domain because reasoning on bitvectors effectively is essential here. After experiments with a number of problem encodings we came up with a method that looks promising. Proving the processor correct manually, or with the ISABELLE system, was complicated and required splitting the theorem up into 18 propositions some of which are non-trivial. In contrast, with the right encoding SPASS can do the proof automatically in less than one minute. In the talk, the application will be introduced, and an account of the encoding method be given as far as it is worked out by now.

This is joint work with Carsten Ihlemann, and in progress still.

Keywords: First-order reasoning, bitvectors

Joint work of: Hillenbrand, Thomas; Ihlemann, Carsten

Inductive Proofs in Information Flow Control - Security in Multiagent Systems

Dieter Hutter (DFKI Saarbrücken, D)

The multi-agent-systems paradigm is becoming more and more popular as a basis for realizing net-based solutions. This development is accompanied by an increasing relevance of security issues. For instance, the potential loss of privacy and other assets is a major concern for both merchants and customers, in Internet-based commerce and, without being properly addressed, such very legitimate concerns hamper the growth of e-commerce.

This talk uses a comparison-shopping scenario to introduce a general methodology for formally verifying the security of multi-agent systems. Following the approach of possibilistic information flow security, the security requirements for the overall system are decomposed into requirements for the individual agents and verified with the help of unwinding techniques. We present heuristics of how to guide an inductive theorem prover in verifying the arising proof obligations.

Keywords: Security, multi-agent system, deduction, induction

Formal verification of for-loops without induction and invariants

Reiner Hähnle (Chalmers UT - Göteborg, S)

Loops are the bottleneck in formal software verification, because they generally require user interaction: typically, induction hypotheses and invariants must be modified by hand in order to go through. This involves expert-level knowledge of the technicalities of the underlying calculus and proof engine. We show that one can replace interactive proof techniques such as induction with automated reasoning in order to deal with certain well-behaved for-loops. The latter means that the execution of loop bodies avoids certain dependencies (this is made precise).

In a first step we check with a static analysis that a given for-loop is well-behaved. This guarantees soundness of a proof rule that transforms a for-loop into a universally quantified update of the state change represented by the loop body. After this transformation of the problem, it is possible to use automatic techniques to compute the strongest postcondition of the loop. The method has been implemented in the KeY verification tool. We evaluated it with representative case studies from the Java Card domain.

Keywords: Formal software verification, dependency analysis, Java Card

Joint work of: Hähnle, Reiner; Gedell, Tobias

Teaching a Quantifier Elimination based Method for Generating Loop Invarinats

Deepak Kapur (University of New Mexico, USA)

In 2003-2004, the author proposed a method for automatically generating loop invariants of imperative programs based on quantifier elimination in theories. An overview of this method is first presented. This semester, this method is being taught in a graduate level class on programming language semantics at the University of New Mexico. Students' experience in using this method is briefly reviewed. The method is especially appealing for this purpose since benefits of the method are proportional to the amount of effort being put into heuristics for approximating quantifier-elimination. The method is sound in the sense that an incomplete method for quantifier-elimination does not produce an incorrect invariant but rather, a weak invariant. The methodology is illustrated using a variety of examples.

Keywords: Program invariants, quantifier elimination, heuristics

Full Paper:
http://www.cs.unm.edu/ kapur/myabstracts/aca2004.html

See also: Deepak Kapur, Automatically Generating Loop Invariants using Quantifier Elimination, Proc. IMACS Intl. Conf. on Applications of Computer Algebra (ACA), 2004, Beaumont, TX, USA, July 2004

Automatically Generating Loop Invariants Using Quantifier Elimination

Deepak Kapur (University of New Mexico, USA)

An approach for automatically generating loop invariants using quantifier-elimination is proposed. An invariant of a loop is hypothesized as a parameterized formula. Parameters in the invariant are discovered by generating constraints on the parameters by ensuring that the formula is indeed preserved by the execution path corresponding to every basic cycle of the loop. The parameterized formula can be successively refined by considering execution paths one by one; heuristics can be developed for determining the order in which the paths are considered. Initialization of program variables as well as the precondition and postcondition of the loop, if available, can also be used to further refine the hypothesized invariant. Constraints on parameters generated in this way are solved for possible values of parameters. If no solution is possible, this means that an invariant of the hypothesized form does not exist for the loop. Otherwise, if the parametric constraints are solvable, then under certain conditions on methods for generating these constraints, the strongest possible invariant of the hypothesized form can be generated from most general solutions of the parametric constraints. The approach is illustrated using the first-order theory of polynomial equations as well as Presburger arithmetic.

Keywords: Program verification, loop invariants, inductive assertions, quantifier elimination

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2006/511

Temporal logics over transitive states and their relation to dynamical systems

Boris Konev (University of Liverpool, GB)

Dynamical systems are usually represented by some 'mathematical' space W (modelling possible system states) and a function f on W (modelling the evolution of the system), with one of the main research problems being the study of iterations of f, in particular, the orbits $O(w) = \{w, f(w), f^2(w), ...\}$ of states w in W.

Since infinite iterations of f cannot be represented in first-order logic, a different logical formalism is required. One possible formalism for speaking about such iterations is a variant of temporal logic. We investigate the computational

behaviour of ‘two-dimensional’ propositional temporal logics over natural numbers (with and without the next-time operator) that are capable of reasoning about states with transitive relations. We show that temporal logics over infinite expanding domains are undecidable even for the language with the sole temporal operator ‘eventually.’ From this result we get the undecidability of the dynamical topological logic of Aleksandrov spaces.

Keywords: Modal logic, temporal logic, dynamical systems, lossy computations

Theory Instantiation

Konstantin Korovin (Manchester University, GB)

We continue our previous work on instantiation-based theorem proving and present a method for integrating theory reasoning in this framework.

This method allows us to integrate theory reasoning in a uniform manner for different theories. The theory reasoner can be seen as a black-box that is only required to satisfy certain natural requirements. We prove completeness of the resulting calculus provided that the theory reasoner satisfies the requirements.

As one of the applications of our approach we show how it is possible to combine instantiation calculus with other calculi, e.g., ordered resolution and paramodulation.

Keywords: Automated deduction; instantiation methods

Joint work of: Ganzinger, Harald; Korovin, Konstantin

Modular Static Analysis with Sets and Relations

Viktor Kuncak (MIT - Cambridge, USA)

Complexity of data structures in modern programs presents a challenge for current analysis and verification tools, forcing them to report false alarms or miss errors. I will describe a new approach for verifying programs with complex data structures. This approach builds on program analysis techniques, as well as decision procedures and theorem provers.

The approach is based on specifying interfaces of data structures by writing procedure preconditions and postconditions in terms of abstract sets and relations. Our system then separately verifies that 1) each data structure conforms to its interface, 2) each data structure interface is used correctly, and 3) desired high-level application-specific invariants hold. The system verifies these conditions by combining decision procedures, theorem provers, and static analyses, promising an unprecedented tradeoff between precision and scalability. In the context of this system, we have developed new decision procedures for reasoning about sets and their cardinalities, approaches for extending the applicability of existing decision procedures, and techniques for modular analysis of dynamically created data structure instances.

Keywords: Static analysis, data structure consistency, program verification, decision procedures

Joint work of: Kuncak, Viktor; Rinard, Martin; Lam, Patrick; Wies, Thomas; Podelski, Andreas; Marnette, Bruno

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2006/512

Predicate Abstraction via Symbolic Decision Procedures

Shuvendu Lahiri (Microsoft Research, USA)

We present a new approach for performing predicate abstraction based on *symbolic decision procedures*. A symbolic decision procedure for a theory T (SDP_T) takes sets of predicates G and E and symbolically executes a decision procedure for T on $G' \cup \{\neg e \mid e \in E\}$, for all the subsets G' of G. The result of SDP_T is a shared expression (represented by a directed acyclic graph) that implicitly represents the answer to a predicate abstraction query. We present symbolic decision procedures for the logic of Equality and Uninterpreted Functions(EUF) and Difference logic (DIF) and show that these procedures run in pseudo-polynomial (rather than exponential) time. We then provide a method to construct SDP's for simple mixed theories (including EUF + DIF) using an extension of the Nelson-Oppen combination method. We present preliminary evaluation of our procedure on predicate abstraction benchmarks from device driver verification in SLAM.

Keywords: Predicate abstraction, decision procedures, software verification

Joint work of: Lahiri, Shuvendu; Ball, Thomas; Cook, Byron

Full Paper:
http://research.microsoft.com/research/pubs/view.aspx?tr_id=906

Constructing Bachmair-Ganzinger Models

Christopher Lynch (Clarkson University - Potsdam, USA)

The Bachmair-Ganzinger Model Construction technique was originally developed for proving completeness of inference systems. We show how this model construction technique can also be used to answer questions about the model in the nonequality case. We believe that this is the first such method for nonground clauses. For non-Horn clauses, we must add a mild splitting inference. We make an assumption about the ordering that is often true about orderings used in practice.

Keywords: Automated deduction, model construction

Towards Efficient Boolean Circuit Satisfiability Checking

Ilkka Niemelä (Helsinki University of Technology, FIN)

Boolean circuits offer a natural, structured, and compact representation of Boolean functions for many application domains such as computer aided verification. We study satisfiability checking methods for Boolean circuits. As a starting point we take the successful Davis-Putnam-Logemann-Loveland (DPLL) procedure for satisfiability checking of propositional formulas in conjunctive normal form and study its generalization to Boolean circuits. We employ a tableau formulation where DPLL propagation rules correspond to tableau deduction rules and splitting corresponds to a tableau cut rule. It turns out that Boolean circuits enable interesting deduction (simplification) rules not typically available in DPLL where the idea is to exploit the structure of the circuit. We also study the relative efficiency of different variations of the cut (splitting) rule obtained by restricting the use of cut in several natural ways. A number of exponential separation results are obtained showing that the more restricted variations cannot polynomially simulate the less restricted ones. The results also apply to DPLL for formulas in conjunctive normal form obtained from Boolean circuits by using Tseitin's translation. Thus DPLL with the considered cut restrictions, such as allowing splitting only on the variables corresponding to the input gates, cannot polynomially simulate DPLL with unrestricted splitting.

Keywords: Boolean circuits, satisfiability, tableau method, proof complexity, DPLL method

Joint work of: Niemelä, Ilkka; Junttila, Tommi; Järvisalo, Matti

Full Paper:
http://dx.doi.org/10.1007/s10472-005-7034-1

See also: Matti Järvisalo, Tommi Junttila, and Ilkka Niemelä. Unrestricted vs Restricted Cut in a Tableau Method for Boolean Circuits. Annals of Mathematics and Artificial Intelligence, 44(4), 373-399, 2005.

The DPLL(T) approach in the BarcelogicTools system

Albert Oliveras (TU of Catalonia - Barcelona, E)

In this talk we will describe the DPLL(T) approach for Satisfiability Modulo Theories (SMT), the problem of deciding the satisfiability of a formula with respect to a background theory of interest, which has important industrial applications. Special emphasis will be made on our DPLL(T) implementation: the BarcelogicTools for SMT.

First, an overview of SMT procedures will be given using the Abstract DPLL Modulo Theories framework, an extension of our rule-based formulation of the Davis-Putnam-Logemann-Loveland (DPLL) procedure.

Apart from providing a simple and clean presentation of DPLL(T), this framework also allows one to model the various existing techniques for SAT and for SMT and, unlike when reasoning on pseucode fragments, properties such as soundness, completeness or termination can be discussed in a clear and uniform way for all these variants.

The second part of the talk will be devoted to a concrete implementation of the DPLL(T) approach: our BarcelogicTools system for SMT. Its architecture will be described and experimental results will be reported. Finally, time permitting, special emphasis will be made on the different techniques and strategies which might help to explain the successful performance of BarcelogicTools.

Keywords: Satisfiability Modulo Theories, decision procedures

Contextual Modal Type Theory: A Foundation for Meta-variables

Brigitte Pientka (McGill University - Montreal, CDN)

In recent years, higher-order reasoning systems and logical frameworks have matured and been successful in several applications such as proof-carrying code. Maybe surprisingly, there are still some foundational and implementation issues which are poorly understood.

In this talk, we concentrate on the notion of meta-variables and their role in higher-order unification. Both concepts are fundamental to proof search, type reconstruction and representation of incomplete proofs. First, we will present a contextual modal type theory which provides an elegant, uniform foundation for understanding meta-variables and explicit substitutions. Second, we will sketch applications in higher-order pattern unification. This will provide new insights for efficient implementation strategies and justifies logically design decisions which have so far been largely motivated operationally. Finally, if time permits, we will briefly discuss applications of this work to the representation of incomplete proofs in theorem proving and open code in functional programming.

Keywords: Type theory, logical frameworks, intuitionistic modal logic

Generation of Invariant Conjunctions of Polynomial Inequalities Using Convex Polyhedra

Enric Rodríguez-Carbonell (TU of Catalonia - Barcelona, E)

A technique for generating invariant polynomial inequalities of bounded degree is presented using the abstract interpretation framework. It is based on overapproximating basic semi-algebraic sets, i.e., sets defined by conjunctions of polynomial inequalities, by means of convex polyhedra. While improving on the existing methods for generating invariant polynomial equalities, since polynomial inequalities are allowed in the guards of the transition system, the approach does not suffer from the prohibitive complexity of the methods based on quantifier-elimination. The application of our implementation to benchmark programs shows that the method produces non-trivial invariants in reasonable time. In some cases the generated invariants are essential to verify safety properties that cannot be proved with classical linear invariants.

The talk is based on joint work with Roberto Bagnara and Enea Zaffanella.

Keywords: Program verification; abstract interpretation; invariants; convex polyhedra

Joint work of: Bagnara, Roberto; Rodríguez-Carbonell, Enric; Zaffanella, Enea

Full Paper:
http://www.lsi.upc.edu/ erodri

See also: Roberto Bagnara, Enric Rodríguez-Carbonell and Enea Zaffanella. Generation of Basic Semi-algebraic Invariants Using Convex Polyhedra. In 12th International Symposium on Static Analysis (SAS'05), Volume 3672 of LNCS, pp. 19-34, September 2005, London (UK).

Using Deduction in Safety Analysis: the Elbtunnel case study

Gerhard Schellhorn (Universität Augsburg, D)

Safety Analysis is an important task in the design and analysis of embedded systems like automotive, railway or avionic systems.

Usually techniques like Fault Tree Analysis (FTA) or Failure Mode and Effects Analysis (FMEA) are applied informally to identify safety critical components and to quantify the risks associated with them.

Using the height control of the Elbtunnel in Hamburg as an example application, the talk will present our ForMoSA (Formal Models and Safety Analysis) approach, which integrates safety analysis, formal methods and mathematical optimization.

The approach generates temporal logic proof obligations from a formal model together with an FTA or DCCA (Discrete Cause-Consequence Analysis; a generalization of both FTA and FMEA) which ensure that no safety relevant events have been overlooked. They are proved using interactive verification or model checking in case of finite-state models. The results of FTA and DCCA are in turn used to optimize system parameters such that safety risks are minimized.

Keywords: Safety Analysis, Fault Tree Analysis, Formal Methods, Deduction, Model Checking

Polynomial Equality Testing for Terms with Shared Substructures

Manfred Schmidt-Schauss (Universität Frankfurt, D)

Sharing of substructures like subterms and subcontexts in terms is a common method for space-efficient representation of terms, which allows for example to represent exponentially large terms in polynomial space, or to represent terms with iterated substructures in a compact form. We show that the equality test of terms with sharing can be done in polynomial time in the size of the representation.

We present singleton tree grammars as a general formalism for the treatment of sharing in terms. Singleton tree grammars (STG) are recursion-free context-free tree grammars without alternatives for non-terminals and at most unary second-order nonterminals. STGs generalize Plandowski's singleton context free grammars to trees. We show that the test, whether two different nonterminals representing terms in a singleton tree grammar generate the same term can be done in polynomial time.

This will allow better algorithms for terms exploiting sharing and improved upper complexity bounds for second order unification algorithms, in particular for variants of context unification and bounded second order unification.

Keywords: Sharing of Terms, tree grammars, context-unification

Solvability with Resolution of Problems in the Bernays-Schoenfinkel Class

Renate Schmidt (Manchester University, GB)

We present a number of results on the solvability and complexity of problems in the Bernays-Schoenfinkel (BS) class. We focus on resolution methods and show how the BS class and some natural subclasses can be decided by general-purpose resolution procedures. The decision procedures can be easily implemented in current resolution theorem provers for first-order logic and initial tests are encouraging.

Keywords: Decidability, resolution, Bernays-Schoenfinkel class

Using Change Information. The Frame Problem in Software Verification

Peter H. Schmitt (Universität Karlsruhe, D)

The performance and usability of deductive program verification systems can be greatly enhanced if specifications of programs and program parts not only consist of the usual pre-/postcondition pairs and invariants but also include additional information on which memory locations are changed by executing a program.

This allows to separate the aspects of which locations change from how they change. Modern specification and annotations languages allow to state information of this kind in a compact way.

In this presentation we present a method how change information can be efficiently used in proving program properties. The method has been implemented and is successfully used in the KeY software verification system.

This talk is based on the publication "An Improved Rule for While Loops in Deductive Program Verification" by Bernhard Beckert, Steffen Schlager and Peter H. Schmitt.

Keywords: Program verification, frame problem, Dynamic Logic

Joint work of: Beckert, Bernhard; Schlager, Steffen; Schmitt, Peter H.

See also: Accepted for 7th International Conference on Formal Engineering Methods, Manchester, UK

Synthesis of Reliable Programs

Johann M. Schumann (NASA (RIACS) - Moffett Field, USA)

In this talk, I will describe the AutoBayes/AutoFilter program synthesis system. It is a schema-based system for the automatic generation of reliable data-analysis and state estimation programs from high-level specifications.

Reliability and high quality of the generated code is paramount for all mission- and safety-critical code. Therefore, the AutoBayes/AutoFilter system generates standardized design documents and supports safety policy based certification. Using a Hoare-style approach and a verification condition generator, we use an automated theorem prover to prove important safety aspects of the software (e.g., array-bounds or variable initialization).

Full automation can be accomplished by having AutoBayes/AutoFilter synthesize all required annotations (e.g., loop invariants) and by performing powerful simplification steps for the proof obligations.

Keywords: Program Synthesis, Data Analysis, Automated Theorem Proving

Proof Presentation

Jörg Siekmann (DFKI Saarbrücken, D)

The talk is based on a book about the human-oriented presentation of a mathematical proof in natural language, in a style as we may find it in a typical mathematical text book.

How can a proof be other than human-oriented? What we have in mind is a deduction systems, which is implemented on a computer, that provesŮwith some human interactionŮa mathematical textbook as may be used in an undergraduate course. The proofs generated by these systems today are far from being human-oriented and can in general only be read by an expert in the respective field: proofs between several hundred (for a common mathematical theorem), for more than a thousand steps (for an unusually difficult theorem) and more than ten thousand deduction steps (in a program verification task) are not uncommon.

Although these proofs are provably correct,they are typically marred by many problems: to start with, that are usually written in a highly specialised logic such as the resolution calculus, in a matrix format, or even worse, they may be generated by a model checker. Moreover they record every logical step that may be necessary for the minute detail of some term transformation (such as, for example, the rearrangement of brackets) along side those arguments, a mathematician would call important steps or heureka-steps that capture the main idea of the proof. Only these would he be willing to communicate to his fellow mathematicians-provided they have a similar academic background and work in the same mathematical discipline. If not, i.e. if the proof was written say for an undergraduate textbook, the option of an important step may be viewed differently depending on the intended reader.

Now, even if we were able to isolate the ten important steps Ů out of those hundreds of machine generated proof steps Ů there would still be the startling problem that they are usually written in the 'wrong' order. A human reader might say: 'they do not have a logical structure'; which is to say that of course they follow a logical pattern (as they are correctly generated by a machine), but, given the convention of the respective field and the way the trained mathematician in this field is used to communicate, they are somewhat strange and ill structured.

And finally, there is the problem that proofs are purely formal and recorded in a predicate logic that is very far from the usual presentation that relies on a mixture of natural language arguments interspersed with some formalism.

The book (about 800 page) which gives an answer to some of these problems is to appear with Elsevier

Keywords: Artificial intelligence, mathematics, proof presentation

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2006/561

See also: to appear with Elsevier

A Fresh Look at the Given Clause Algorithm

John Slaney (Australian National University - Canberra, AU)

The given clause algorithm is the basis of most contemporary high performance theorem provers, and has been so for some decades. Surprisingly, in all that time nobody seems to have looked carefully at what actually happens during bottom-up proof searches. The present talk reports some early findings from a project

on visualisation of automated reasoning. It does not reach a simple conclusion but rather demonstrates the complexity of the processes hidden behind the usual runtime statistics such as the number of loop iterations or the number of clauses generated.

Keywords: Theorem proving, first order deduction, given clause loop, visualisation

On Properties of Local Theory Extensions: Hierarchical and Modular Reasoning, Interpolation.

Viorica Sofronie-Stokkermans (MPI für Informatik, D)

We present the properties of a special type of extensions of a base theory, which we call local. Many theories important for computer science or mathematics are local extensions of a base theory. Some examples are:

– theories of data structures, e.g. theories of lists or arrays,
– theories of constructors and selectors (important in program verification, but also in cryptography),
– theories of monotone functions over an ordered domain (applications in knowledge representation, or in verification),
– theories of (monotone) functions satisfying certain boundedness conditions (relevant e.g. in the parametric verification of real-time or hybrid systems),
– theories of mathematics, e.g. the theory of functions satisfying the Lipschitz conditions at a given point.

The notion of local extension of a theory generalizes the notion of locality of a theory introduced by Givan and McAllester (1992), and of locality of equational theories studied by Ganzinger (2001). For local theories, validity of ground Horn clauses can be checked in polynomial time.

We show that for local extensions of a base theory efficient hierarchical reasoning, in which a theorem prover for the base theory is used as a "black box", is possible. We identify situations in which, for an extension $\mathcal{T}_1$ of a theory $\mathcal{T}_0$, the decidability (and complexity) of the universal theory of $\mathcal{T}_1$ can be expressed in terms of the decidability (resp. complexity) of suitable fragments of the theory $\mathcal{T}_0$ (universal or $\forall\exists$). We also discuss combinations of local theory extensions, possibilities of modular reasoning, as well as interpolation properties.

Keywords: Automated reasoning, Combinations of decision procedures

Disconnection Tableaux with Link Blocking

Gernot Stenz (TU München, D)

The disconnection tableaux calculus is currently the most successful tableaux-based method for first-order automated theorem proving. Among its characteristics the branch saturation property is the most interesting.

This means that when an open branch cannot be extended in a non-redunant manner, it describes a model for the formula. In practice, however, finding such a model heavily depends on using the proper selection functions.

In our talk we describe link blocking, a new method of redundancy elimination similar to so-called "productivity restrictions" in other calculi like model evolution. Here, as a kind of semantic guidance, the current branch is treated like a candidate model that is continuously refined by applying inference steps. Link blocking disallows the application of inference steps producing literals that are already true in the candidate model. This way the termination behaviour of the calculus becomes much more robust and independent of the branch and inference selection functions employed.

In terms of proving completeness, link blocking requires an extension of the exception-based representation (EBR) technique developed for actually extracting a model from a saturated branch. The move from EBRs to candidate models necessitated the introduction of an acyclic ordering on links no longer based mainly on the instantiatedness of the involved literals.

For ground literals, the general technique of link blocking collapses to the regularity restriction on branch literals. As regularity cannot be enforced for proofs involving theory equality, link blocking promises to provide a more general framework still applicable also in the equational case. Then, however, several important questions have to be addressed regarding completeness and the decidability of the candidate model entailment condition.

Keywords: First-order tableaux models disconnection

Canonizing Congruence Proofs

Aaron Stump (Washington University, USA)

In this talk, I describe a family of convergent term rewriting systems for putting congruence proofs into canonical form. Congruence proofs are built from assumptions using proof rules for reflexivity, symmetry, and transitivity, as well as one congruence rule for every pair of function symbol and argument position for that function symbol. The algebra of congruence proofs is shown to be that of commuting group endomorphisms, under a certain extension of the condition that assumptions are logically independent.

Convergent completions of this algebra are obtained first for $k = 2$ commuting group endomorphisms, and then extended to the cases where $k > 2$. Obtaining these completions is non-trivial, due to the commutation rules, which are not orientable using LPOs or KBOs. For example, Waldmeister can obtain ground convergent completions only up to $k = 5$, where the computation generates over 100 million critical pairs, takes 22 hours, and results in a saturated set of 670 rules and 11240 equations. In contrast, our method describes the completion for all $k \geq 2$, and the completion for $k = 5$ consists of just 50 rules.

Keywords: Completion, congruence closure, Waldmeister, commuting group endomorphisms
Joint work of: Stump, Aaron; Loechner, Bernd

Solving Binary Integer Programs with DPLL(T)

Cesare Tinelli (University of Iowa, USA)

Binary Integer Programming (BIP), also known as pseudo-Boolean constraint solving, is a special cause of linear integer programming in which all input variables range over the set 0,1. In this talk I will describe a method for readily building solvers for the feasibility of BIP programs on top of the DPLL(T) scheme for satisfiability modulo theories. The method requires some simple preprocessing of the input program and the implementation of a similarly simple theory solver for the DPLL(T) scheme. I will present preliminary results showing that BIP solvers built this way can be competitive with state-of-the-art BIP solvers.

Keywords: Binary integer programming, satisfiability modulo theories, pseudo-boolean constraints, DPLL

Unification in assertion checking over logical lattices

Ashish Tiwari (SRI - Menlo Park, USA)

We explicate the connection between unification and assertion checking. Specifically, we show that weakest preconditions can be strengthened by replacing equalities by their unifiers, without losing any precision during backward analysis of programs. Using this result, we establish the decidability of assertion checking for programs (with nondeterministic conditionals) over the combined language of linear arithmetic and uninterpreted symbols. This result is surprising since the corresponding lattice has unbounded height. For loop-free programs, assertion checking is coNP-complete. In contrast, programs over languages of the individual theories have polynomial time assertion checking algorithms.

Keywords: Abstract interpretation, weakest precondition, unification, NP-hardness, decidability

Joint work of: Tiwari, Ashish; Gulwani, Sumit

Combatting SUMO and Terrorism with Vampire

Andrei Voronkov (Manchester University, GB)

We identify a number problem hampering the use of first-order theorem provers for reasoning with large ontologies based on first-order logic and its extensions.

Then we describe a modification of the theorem prover Vampire able to reason with ontologies containing over 20,000 first-order formulas with equality. We also give a brief analysis of inconsistencies found by Vampire in the SUMO and terrorism ontologies.

Keywords: Ontology reasoning inconsistency

SPASS + T

Uwe Waldmann (MPI für Informatik, D)

We describe how we have linked decision procedures for arithmetic to the superposition theorem prover SPASS; we discuss the capabilities, limitations, and applications of such a combination; and we sketch extensions of the current implementation.

Keywords: SPASS, superposition, theorem proving, decision procedures, arithmetic

Reasoning about Incompletely Defined Programs

Christoph Walther (TU Darmstadt, D)

We consider automated reasoning about recursive partial functions with decidable domain, i.e. functions computed by incompletely defined but terminating functional programs. Incomplete definitions provide an elegant and easy way to write and to reason about programs which may halt with a run time error by throwing an exception or printing an error message, e.g. when attempting to divide by zero. We investigate the semantics of incompletely defined programs, define an interpreter for those programs and discuss the termination of incompletely defined procedures. We then analyze which problems need to be solved if a theorem prover designed for verification of completely defined programs is modified to work for incompletely defined programs as well. We also discuss how to reason about stuck computations which arise when calling incompletely defined procedures with invalid arguments. Our method of automated reasoning about incompletely defined programs has been implemented in the verification tool VeriFun. We conclude by discussing experiences obtained in several case studies with this implementation and also compare and relate our proposal to other work.

Keywords: Loose / Underspecifications, partial functions, program verification, automated reasoning

Joint work of: Walther, Christoph; Schweitzer, Stephan

Full Paper:
http://www.inferenzsysteme.informatik.tu-darmstadt.de/ walther/

See also: Proceedings LPAR-12, Springer LNAI 3835, pp. 427-442 (2005)

Formal Verification of an Off-Line Result Checker for Priorities Queus

Hans de Nivelle (MPI für Informatik, D)

We describe how we formally verified the result checker for priority queues that is implemented in LEDA (a large software library developed at MPI).

A result checker of some datastructure is an additional datastructure that checks all input and output of the first datastructure, and which is able to detect when the first datastructure makes a mistake. We have developed a formalism, based on the notion of implementation function, that links abstract specifications to concrete implementations. The formalism allows non-determinism in the abstract specification that can be filled in by the concrete implementation. The formalism is general: It can be used for testing, result checking and also for verification. Using the formalism, we have formally verified that, if the checker has not reported an error up to a certain moment, then the structure checked by it has behaved like a priority queue up to that moment.

For the verification, we made use of the first-order theorem prover Saturate.

Keywords: Verification of Object-Oriented Specifications, Result Checking, Theorem Proving

Joint work of: de Nivelle, Hans; Piskac, Ruzica