Proceedings of the Twenty-Ninth International
Florida Artificial Intelligence Research Society Conference

Hoping for the Truth – A Survey of the TPTP Logics

Geoff Sutcliffe
Department of Computer Science
University of Miami, USA

Francis Jeffry Pelletier*
Department of Philosophy
University of Alberta, Canada

Abstract

This paper compares features of the classical logics that are commonly used in the TPTP-based automated reasoning community for representing chosen aspects of "the world", and the consequent implications for reasoning about these representations. The paper argues that increases in complexity in terms of representation and reasoning force users to compromise between the reliability of the representation and the reliability of the reasoning.

Introduction

This paper compares features of the classical logics that are commonly used in the TPTP-based (Sutcliffe 2010) automated reasoning (AR) community for representing chosen aspects of "the world", and the consequent implications for reasoning about the representations. It provides a high-level, novice-friendly, but also authoritative, overview, highlighting key characteristics of the logics, and reporting on the state-of-the-art in reasoning for the logics. The valuations made about the logics, and the conclusions drawn, are founded in the features of the logics. Additionally, the experience of the first author using these logics and evaluating AR systems for the logics has provided useful insights. The analyses and discussion might be viewed as contributions to the goals of Universal Logic (Béziau 2007).

When comparing different ways of representing and reasoning about a user's domain of interest[1], it is necessary for the representation and reasoning to capture a common notion of truth (Sutcliffe and Suttner 2006) (see the discussion of "realism" vs. "nominalism" in (Pelletier 1991)). The common notion of truth in classical logics is founded in *soundness* and *completeness*. Soundness is *required*: The representation does not include anything that is not actually the case in the domain of interest, and reasoning over the representation does not lead from truths to non-truths. In the terminology of the AR community, the representation is the *axioms*, and sound reasoning deduces only *theorems*. Completeness is *desired*: The representation includes everything that is actually the case in the domain of interest, and reasoning leads to all the truths about the things that have been represented. In the terminology of the AR community, complete reasoning can deduce all theorems, and a statement about which the user would like to determine theoremhood is a *conjecture*. Together, soundness and completeness constitute the *reliability* of the representation and the reasoning. Reliably representing a user's domain of interest is typically easier with more expressive logics, but reliably reasoning about a representation is typically easier in less expressive logics. Thus depending on the choice of logic, the common notion of truth can become more or less attenuated.

This paper examines a hierarchy of classical logics, of increasing expressivity, for reliability. Most of these logics are fully supported in the TPTP world, and the TPTP acronyms for the logics are given in parentheses in the list. Description logic has minimal support in the TPTP, and support for extended typed higher-order logic is in its infancy at the time of writing. The logics[2] are[3]

- Propositional Logic (PL)
- Description Logic (DL)
- Effectively PRopositional first-order logic (EPR)
- Clause Normal Form of first-order logic (CNF)
- First-Order Form of first-order logic (FOF)
- Monomorphic Typed First-order logic (TF0)
- Typed First-order logic with Arithmetic (TFA)
- Polymorphic Typed First-order logic (TF1)
- Monomorphic Typed Higher-order logic (TH0)
- EXtended Typed Higher-order logics (THX)

Each of these logics is examined in terms of features that determine representational and reasoning reliability:

1. Representation – how easy is it to reliably describe the user's domain of interest?

*Supported by Canadian NSERC A5525.

[1]The phrase "domain of interest" is used to neutrally name the part of "the world" that the user wants to represent and reason about. This avoids clashes with notions of "worlds" and "indices" in, e.g., modal or tense logics.

[2]Equality is assumed to be included in all except PL.

[3]It is acknowledged that this survey does not consider other classical logics that are not in the TPTP world, and that have different features and hence different representational and reasoning reliability, e.g., quantified boolean formulae (QBF), equality logic with uninterpreted functions (EUF).

2. Reasoning in principle – what is the decidability; what algorithms and techniques are known for reliably deducing theorems?
3. Reasoning in practice – what is the state-of-the-art in implemented systems for reliably deducing theorems?

For reasoning in principle and in practice, it is assumed that the underlying hardware and software are reliable. For reasoning in principle it is assumed that infinite time and memory (and randomness if required) are available, while for reasoning in practice it is acknowledged that the reality of finite resources affects reliability.

About the Logics

Propositional Logic (PL)

Representation: In PL the only unit of representation is the *proposition* – a statement that is either true or false, but has no further internal structure. There is no way to explicitly denote objects in the user's domain of interest. These restrictions limit the representational reliability.

Reasoning in principle: PL is decidable. PL supports a range of decision procedures. The simplest approach is to use a truth table, but more sophisticated methods are more practically useful. The most commonly used algorithms are the Conflict-Driven Clause Learning (CDCL) and Davis-Putnam-Logemann-Loveland (DPLL) algorithms (Biere et al. 2009).

Reasoning in practice: There are modern trusted implementations, probably the best known of which is (one of the implementations of) MiniSAT (Eén and Sörensson 2005).

Reliability: The representational reliability of PL is rather low – the inability to denote objects, parameterize truth statements by objects, or make quantified statements, makes it difficult to represent any other than the most simple domains of interest. Reasoning in PL is highly reliable, although representations with a large number of propositions can absorb significant resources.

Description Logic (DL)

Representation: DL statements are built from *individuals* in the domain of interest, *classes* that place the individuals into sets, and *roles* that specify binary relationships between individuals. There is no way to denote non-explicit objects, e.g., a DL representation can include the individual `Geoff`, but is unable to refer to the uncle of `Geoff` without including an explicitly named individual to be the uncle. Different DL variants include extensions that add varying expressive power. Most DLs can be seen as fragments of FOF, although some DLs provide operators that are second-order. Thus the representational reliability is greater than PL and some restricted fragments of FOF, but generally less than that of FOF.

Reasoning in principle: DL is decidable, and the complexity of various reasoning tasks is well understood (Donini 2003). One of the important operations in DL is subsumption, to determine which classes are subclasses of which others. Modern algorithms for reasoning in DL are mostly based on semantic tableaux.

Reasoning in practice: The DL community has produced some reasonably effective and trusted implementations. Recent evaluations point to the Konclude system (Steigmiller, Liebig, and Glimm 2014) as being the most reliable.

Reliability: DL limits representational reliability as a trade-off against maintaining decidability in the reasoning component. As such, users are restricted as to what aspects of their domains of interest can be represented, but if enough can be represented then reasoning is very reliable in principle. In practice the reasoning reliability is acceptable, although not as mature as for other logics that have a longer history of reasoning tool development, e.g., PL and FOF.

Effectively Propositional Logic (EPR)

Representation: EPR is a restriction of CNF, to clause sets that are known to be reducible to PL, e.g., CNF problems that have no functions with arity greater than zero. The restrictions that make the clause sets reducible to PL limit the representational reliability, e.g., the absence of functions with arity greater than zero means that, as in DL, it is impossible to represent objects implicitly.

Reasoning in principle: EPR is decidable, and is NEXPTIME complete. If an EPR problem is translated to PL, e.g., (Schulz 2002a), then the principles for reasoning in PL apply. However, other approaches e.g., Inst-Gen (Ganzinger and Korovin 2003), normally perform better.

Reasoning in practice: The EPR logic has been a focus of research in the AR community for about 14 years. Early claims that standard saturation-based systems for CNF such as Vampire (Kovacs and Voronkov 2013) would be most effective for EPR have now been dispelled by systems with specialized EPR reasoning, e.g., iProver (Korovin 2009).

Reliability: The representational reliability of EPR is limited by the constrained use of connectives, and by the inability to refer to non-explicit objects. However, EPR is adequate for many current applications in software and hardware verification. Reasoning in EPR is highly reliable, with well-understood theory, principles, and practice. All together, EPR is reliable if the logic is adequate for representing the domain of interest.

Clause Normal Form (CNF)

Representation: A CNF representation is a conjunction of disjunctions of atomic statements and their negations. CNF extends EPR (really, EPR restricts CNF) by the use of *functions* to represent non-explicit objects, e.g., `uncle_of(geoff)`. If a FOF representation is translated to an equisatisfiable CNF representation, the use of Skolem functions to replace objects that are claimed to exist by existential quantification hampers the representational reliability. Also in comparison to FOF, the limited set of connectives can make it difficult to produce a natural encoding of the domain of interest.

Reasoning in principle: CNF is semi-decidable. The most common approach to establishing theoremhood in CNF is to test the conjunction of the axioms and the negated

conjecture for satisfiability. The most common algorithms for establishing satisfiability are based on resolution and some form of paramodulation (Bachmair et al. 2001), or superposition (Weidenbach 2001). The soundness and completeness properties of these approaches and algorithms are well established. Alternative approaches, based more directly on Herbrand's theorem, e.g., as described in (Newborn 2001), are also known.

Reasoning in practice: There are mature implementations of both the resolution-based and superposition-based approaches, e.g., SPASS (Weidenbach et al. 2009), that are trusted to be sound due to the developer's conservative release policy, but that do not perform as well as more recent implementations, e.g., E (Schulz 2002b) and Vampire. These more recent implementations use a range of advanced techniques that improve performance, but the necessarily complex implementations are regularly revealed to have bugs. The semi-decidability of the logic assumes infinite time and space, so all these systems are incomplete in practice.

Reliability: CNF has better representational reliability than EPR thanks to the representation of non-explicit objects. It is one of the most mature areas of AR, and as such the reasoning reliability is high, but sometimes suffers from developers' quests to improve reasoning performance, resulting in more complex and buggy systems (as shown in some TPTP-based soundness testing).

First Order Form (FOF)

Representation: FOF extends the representational ability of CNF (really, CNF restricts FOF) to allow full use of the connectives and operators of first-order logic, in particular, arbitrary nesting of quantifers and subformulae. This makes it easier for users to naturally encode their domains of interest with a high degree of reliability.

Reasoning in principle: FOF is semi-decidable. FOF representations can be translated to equisatisfiable CNF representations, so the reasoning capabilities of CNF can (and typically are) used. Additionally, techniques that manipulate FOF directly, e.g., analytic tableaux (Hähnle 2001), and natural deduction (Pastre 2001) have been used. The properties of all approaches are well established, ensuring soundness in principle.

Reasoning in practice: Currently the best reasoning systems for FOF are those that translate to CNF, e.g., E and Vampire, and the comments regarding CNF reasoning apply. The reliability of the reasoning can be hampered if the translation is not done with care (Nonnengart and Weidenbach 2001). Systems based on analytic tableaux, etc., are hardly developed, and their implementations are rarely used.

Reliability: The benefits of the representational reliability of FOF outweigh the negative impact of more complex reasoning requirements. Historically, many users of AR represented their domains of interest with CNF because reasoning tools for FOF were inadequately developed, e.g., see early work by mathematicians (Wos, Robinson, and Carson 1965). Times have changed, and by now CNF is the "assembly language" of AR, so that the use of FOF or even more expressive logics is prevalent thanks to the higher representational reliability.

Monomorphic Typed First-order Logic (TF0)

Representation: One of the weaknesses of FOF is that it does not distinguish between types of objects. The TF0 logic adds simple types (or "sorts" - TF0 is also known as "many-sorted logic") to FOF, so that every object has a known type, and every function and predicate has a known signature. Monomorphic types are very easy for users to understand and use. As with programming languages, this increases the representational reliability because compiler-style tools can be used to ensure that a representation is well-typed.

Reasoning in principle: TF0 is semi-decidable. TF0 can be translated to FOF, so the reasoning principles of FOF can be employed. However, the translation procedure appears to be only (unpublished) AR reasoning folklore. There are variants of the translation procedure that lead to different sets of theorems, which makes a translation-based approach somehow unreliable. Some early attempts to develop systems for reasoning directly with TF0, e.g., (Walther 1983; Cohn 1987), did not work well enough to popularize the logic, but more recent approaches have been more successful, e.g. (Barrett et al. 2011; Baumgartner and Waldmann 2013).

Reasoning in practice: The first reliable reasoning system for TF0 was SNARK (Stickel 2012), but recent implementations of the new approaches mentioned above are rapidly maturing, e.g., CVC4 (Barrett et al. 2011).

Reliability: The use of types gives TF0 high representational reliability, and the trajectory of modern reasoning tools for the logic is towards high reliability. If the logic is strong enough for a user's need, TF0 might be optimal.

Typed First-order Logic with Arithmetic (TFA)

Representation: The TFA logic extends TF0 to include arithmetic over integers, rationals, and reals. A reasonably rich set of arithmetic functions and relations have been specified. This increases the representational reliability (compared to the underlying TF0 logic) quite dramatically, because numbers and arithmetic are a natural part of many domains of interest. This logic provides linkage to SMT logics such as QF_LIA and UFNIRA (Barrett, Stump, and Tinelli 2010).

Reasoning in principle: TFA is undecidable, due to the inclusion of arithmetic. However, some restricted subsets of TFA, e.g., using only linear integer arithmetic, are semi-decidable (Nipkow 2008). The most common technique for combining the symbolic reasoning of TF0 with the theory reasoning of arithmetic is to keep them separate, and pass logical and control information between the two components, e.g., as in DPLL(T) (Ganzinger et al. 2004). There are some approaches that integrate the two aspects more tightly, but these are largely ad hoc, e.g., as implemented in SNARK. Both approaches lose completeness for various

reasons, e.g., some approaches are limited to evaluation of ground arithmetic expressions.

Reasoning in practice: Associated with the maturing implementations of systems for TF0, there are rapidly maturing systems for TFA, e.g., CVC4. However, the integration of arithmetic reasoning reduces the reliability due to (i) the incompleteness of the arithmetic reasoning; (ii) informal methods for exchanging information between the separate components; (iii) imprecise implementations of arithmetic reasoning. These weaknesses are either impossible or at least very difficult to overcome.

Reliability: TFA has a higher representational reliability than TF0, but as a result a lower reasoning reliability. In combination it's hard to say if the trade-off results in a higher or lower overall reliability. As reasoning systems for TFA mature it can be expected that the reasoning reliability will improve, more so than for TF0. In the future TFA is likely to be preferred over TF0 because most users will want arithmetic in their representational language.

Polymorphic-typed First-Order Logic (TF1)

Representation: The TF1 logic is the polymorphic extension of TF0. This allows for compact reuse of type constructs, which is an advantage in many applications, e.g, in interactive theorem provers and program specification languages. For naïve users, the representational reliability might be diminished (relative to TF0), due to the added complexity, but for power users it's a definite plus.

Reasoning in principle: TF1 is semi-decidable. Monomorphization can be used to translate TF1 to TF0, but this is generally incomplete, and hence unreliable. There are techniques for reliably dealing with polymorphism completely within the reasoning process (Bobot et al. 2008).

Reasoning in practice: The only known (at the time of writing) AR systems for TF1 are Alt-Ergo (Conchon, Iguernelala, and Mebsout 2013), ZenonModulo (Delahaye et al. 2013), and Zipperposition (Cruanes 2015). Alt-Ergo is respected as reliable (since it correctly implements the process described in (Bobot et al. 2008)) and of reasonably high performance. The lack of diversity of reasoning systems is a weakness in the reasoning reliability of TF1.

Reliability: TF1 is a reliable logic for users who are able to master the expressivity, and who use an appropriately reliable reasoning tool. In the broader view of AR, this is a comparatively narrow band of reliability, and further familiarity and development of systems would enhance the reliability of TF1.

Monomorphic Typed Higher-order Logic (TH0)

Representation: The TH0 logic is the simply typed lambda calculus with Henkin semantics. As such it provides the useful features of higher-order logics that are not found in less expressive logics: quantification over functions and relations, lambda abstraction, logical operators as individuals, definite description, etc. (Farmer 2008). As is the case for TF1, the richness of TH0 might overwhelm some users, but for power users it provides high representational reliability. In particular, much of mathematics (e.g., set theory (Benzmüller et al. 2008)) can be elegantly and succinctly encoded in TH0.

Reasoning in principle: TH0 is semi-decidable. There are various approaches to reasoning in TH0, including translation to FOF, encoding with combinators, tableau-based approaches, and resolution/paramodulation approaches.

Reasoning in practice: The last decade has seen the development of significantly more reliable reasoning systems for TH0, e.g., Satallax (Brown 2012). Many systems partially rely on translation to a less expressive logic and use of a corresponding reasoning system, which results in some loss of completeness due to optimizations that aim to make the translated version easier for the subordinate reasoning system. The reliability of systems that reason directly in TH0 has fluctuated as new features have been added to increase their reasoning power but their initial implementations have introduced bugs. Generally, however, due to the well-established theory of the logic, the reasoning reliability is quite high.

Reliability: In principle, the high expressivity of TH0 allows for high representational reliability, and the maturity of the logic allows for high reasoning reliability. In practice the complexity of the logic might reduce the representational reliability for some users, but the relatively recent development of fully automated systems for TH0 positively impacts the reasoning reliability. Finally, TH0 does not provide for reasoning with arithmetic, which was observed to improve the representational reliability of TFA over TF0.

Extended Typed Higher-order Logic (THX)

Representation: Some aspects of THX have been designed, but none have been formally added to the TPTP world. The 'X' in the name is meant as a variable, standing for various areas of logic. At the very least, 'X' includes 'A' for TH0 with arithmetic, and '1' for polymorphic typed higher-order logic. Other extensions, e.g., to allow types as terms and terms as types, tuples, sequents, etc. (Barendregt, Dekkers, and Statman 2013), have also been designed. These extensions will provide greater (but more complex) representational reliability.

Reasoning in principle: The THX logics are at best semi-decidable, but depending on what extensions are included they might be undecidable. The principles for reasoning in these rich logics have often emerged in the context of implementations, and some might be considered rather ad hoc.

Reasoning in practice: Various interactive theorem proving systems, e.g., Isabelle (Nipkow, Paulson, and Wenzel 2002) and Coq (Bertot and Casteran 2004), support different extensions. There has been almost no development of AR for the extensions, and therefore the reasoning reliability is low.

Reliability: The lack of AR systems for the THX logics means that the reliability cannot be assessed.

Comparing the Logics

A graphical distribution of the representational and reasoning reliability of the logics examined in this paper is given in Figure 1. The horizontal axis judges the logics for the representational reliability, and the vertical axis judges them on their reasoning reliability. The placement of the points are qualitative judgements based on the observations made above, but have the strength of the first author's experience in the area. The trade-off between representational reliability and reasoning reliability is clearly evident. Note that the representational reliability of each logic is fixed, but over time the principles and practice of AR for a logic typically improve, i.e., the slope of the regression line decreases.

It is tempting to claim that the logic whose vector from the origin is longest is the most reliable logic. However, for a logic to be generally applicable and useful it needs to have some minimal representational and reasoning reliability. As such the PL and THX might be excluded, despite their long vectors. At the time of writing, the "sweet spot" appears to be TF0, but the additional representational reliability of TFA, coupled with rapidly maturing principles and implementations of reasoning systems for the logic suggest it might soon become the logic of choice.

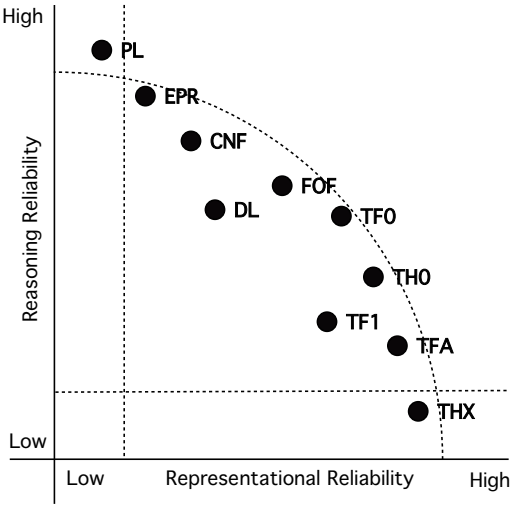

Figure 1: Representational and Reasoning Reliability

Conclusions (about the Truth)

This paper has surveyed the TPTP hierarchy of classical logics for representing real-world problems, and the concomitant computational difficultly of reasoning about the problems in an efficient manner. The quest to reliably represent a user's domain of interest, and reliably reason over that representation, is dependent on features of the chosen logic. More expressivity typically increases the reliability of representation but decreases the reliability of reasoning, and conversely less expressivity typically decreases the reliability of representation but increases the reliability of reasoning. Currently there is no logic that simultaneously provides very high representational reliability and very high reasoning reliability. Therefore, it might be necessary to let go of the desired "common notion of truth" that is discussed in the introduction, and rather have a "hope for truth". Thus, instead of saying "QED" (*quod erat demonstrandum* – that "which had to be demonstrated"), we should really say "QSD": (*quod speramus demonstravisse* – that "which we *hope* to have demonstrated").[4]

References

Bachmair, L.; Ganzinger, H.; McAllester, D.; and Lynch, C. 2001. Resolution Theorem Proving. In Robinson, A., and Voronkov, A., eds., *Handbook of Automated Reasoning*. Elsevier Science. 19–99.

Barendregt, H.; Dekkers, W.; and Statman, R. 2013. *Lambda Calculus with Types*. Cambridge University Press.

Barrett, C.; Conway, C.; Deters, M.; Hadarean, L.; Jovanovic, D.; King, T.; Reynolds, A.; and Tinelli, C. 2011. CVC4. In Gopalakrishnan, G., and Qadeer, S., eds., *Proceedings of the 23rd International Conference on Computer Aided Verification*, number 6806 in Lecture Notes in Computer Science, 171–177. Springer-Verlag.

Barrett, C.; Stump, A.; and Tinelli, C. 2010. The SMT-LIB Standard: Version 2.0. In Gupta, A., and Kroening, D., eds., *Proceedings of the 8th International Workshop on Satisfiability Modulo Theories*.

Baumgartner, P., and Waldmann, U. 2013. Hierarchic Superposition With Weak Abstraction. In Bonacina, M., ed., *Proceedings of the 24th International Conference on Automated Deduction*, number 7898 in Lecture Notes in Artificial Intelligence, 39–57. Springer-Verlag.

Benzmüller, C.; Sorge, V.; Jamnik, M.; and Kerber, M. 2008. Combined Reasoning by Automated Cooperation. *Journal of Applied Logic* 6(3):318–342.

Bertot, Y., and Casteran, P. 2004. *Interactive Theorem Proving and Program Development - Coq'Art: The Calculus of Inductive Constructions*. Texts in Theoretical Computer Science. Springer-Verlag.

Béziau, J.-Y. 2007. *Logica Universalis: Towards a General Theory of Logic*. Springer-Verlag.

Biere, A.; Heule, M.; Van Maaren, H.; and Walsh, T. 2009. *Handbook of Satisfiability*. IOS Press.

Bobot, F.; Conchon, S.; Contejean, E.; and Lescuyer, S. 2008. Implementing Polymorphism in SMT solvers. In Barrett, C., and De Moura, L., eds., *Proceedings of the 6th International Joint Conference on Satisfiability Modulo*, number 367 in ACM International Conference Proceedings, 1–5.

Brown, C. 2012. Satallax: An Automated Higher-Order Prover (System Description). In Gramlich, B.; Miller, D.; and Sattler, U., eds., *Proceedings of the 6th International Joint Conference on Automated Reasoning*, number 7364 in Lecture Notes in Artificial Intelligence, 111–117.

Cohn, A. 1987. A More Expressive Formulation of Many Sorted Logic. *Journal of Automated Reasoning* 3(2):113–200.

[4] Thanks to Jack Zupko for this.

Conchon, S.; Iguernelala, M.; and Mebsout, A. 2013. A Collaborative Framework for Non-Linear Integer Arithmetic Reasoning in Alt-Ergo. In Bjorner, N., ed., *Proceedings of the 15th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, 161–168. IEEE.

Cruanes, S. 2015. *Extending Superposition with Integer Arithmetic, Structural Induction, and Beyond*. Ph.D. Dissertation, Ecole Polytechnique, Paris, France.

Delahaye, D.; Doligez, D.; Gibert, F.; Halmagrand, P.; and Hermant, O. 2013. Zenon Modulo: When Achilles Outruns the Tortoise using Deduction Modulo. In McMillan, K.; Middeldorp, A.; and Voronkov, A., eds., *Proceedings of the 19th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning*, number 8312 in Lecture Notes in Computer Science, 274–290. Springer-Verlag.

Donini, F. 2003. Complexity of Reasoning. In Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P., eds., *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press. 96–136.

Eén, N., and Sörensson, N. 2005. MiniSat - A SAT Solver with Conflict-Clause Minimization. In Bacchus, F., and Walsh, T., eds., *Posters of the 8th International Conference on Theory and Applications of Satisfiability Testing*.

Farmer, W. 2008. The Seven Virtues of Simple Type Theory. *Journal of Applied Logic* 6:267–286.

Ganzinger, H., and Korovin, K. 2003. New Directions in Instantiation-Based Theorem Proving. In Kolaitis, P., ed., *Proceedings of the 18th IEEE Symposium on Logic in Computer Science*, 55–64. IEEE Press.

Ganzinger, H.; Hagen, G.; Nieuwenhuis, R.; Oliveras, A.; and Tinalli, C. 2004. DPLL(T): Fast Decision Procedures. In Alur, R., and Peled, D., eds., *Proceedings of the 16th International Conference on Computer Aided Verification*, number 3114 in Lecture Notes in Computer Science, 175–188.

Hähnle, R. 2001. Tableaux and Related Methods. In Robinson, A., and Voronkov, A., eds., *Handbook of Automated Reasoning*. Elsevier Science. 100–178.

Korovin, K. 2009. Instantiation-Based Reasoning: From Theory to Practice. In Schmidt, R., ed., *Proceedings of the 22nd International Conference on Automated Deduction*, number 5663 in Lecture Notes in Computer Science. Springer-Verlag. 163–166.

Kovacs, L., and Voronkov, A. 2013. First-Order Theorem Proving and Vampire. In Sharygina, N., and Veith, H., eds., *Proceedings of the 25th International Conference on Computer Aided Verification*, number 8044 in Lecture Notes in Artificial Intelligence, 1–35. Springer-Verlag.

Newborn, M. 2001. *Automated Theorem Proving: Theory and Practice*. Springer.

Nipkow, T.; Paulson, L.; and Wenzel, M. 2002. *Isabelle/HOL: A Proof Assistant for Higher-Order Logic*. Number 2283 in Lecture Notes in Computer Science. Springer-Verlag.

Nipkow, T. 2008. Linear Quantifier Elimination. In Baumgartner, P.; Armando, A.; and Gilles, D., eds., *Proceedings of the 4th International Joint Conference on Automated Reasoning*, number 5195 in Lecture Notes in Artificial Intelligence, 18–33. Springer-Verlag.

Nonnengart, A., and Weidenbach, C. 2001. Computing Small Clause Normal Forms. In Robinson, A., and Voronkov, A., eds., *Handbook of Automated Reasoning*. Elsevier Science. 335–367.

Pastre, D. 2001. Muscadet 2.3 : A Knowledge-based Theorem Prover based on Natural Deduction. In Gore, R.; Leitsch, A.; and Nipkow, T., eds., *Proceedings of the International Joint Conference on Automated Reasoning*, number 2083 in Lecture Notes in Artificial Intelligence, 685–689. Springer-Verlag.

Pelletier, F. 1991. The Philosophy of Automated Theorem Proving. In Mylopolous, J., and Reiter, R., eds., *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, 1039–1045. Morgan-Kaufmann.

Schulz, S. 2002a. A Comparison of Different Techniques for Grounding Near-Propositional CNF Formulae. In Haller, S., and Simmons, G., eds., *Proceedings of the 15th International FLAIRS Conference*, 72–76. AAAI Press.

Schulz, S. 2002b. E: A Brainiac Theorem Prover. *AI Communications* 15(2-3):111–126.

Steigmiller, A.; Liebig, T.; and Glimm, B. 2014. Konclude: System Description. *Web Semantics* 27(1):To appear.

Stickel, M. 2012. SNARK - SRI's New Automated Reasoning Kit. http://www.ai.sri.com/~stickel/snark.html.

Sutcliffe, G., and Suttner, C. 2006. The State of CASC. *AI Communications* 19(1):35–48.

Sutcliffe, G. 2010. The TPTP World - Infrastructure for Automated Reasoning. In Clarke, E., and Voronkov, A., eds., *Proceedings of the 16th International Conference on Logic for Programming Artificial Intelligence and Reasoning*, number 6355 in Lecture Notes in Artificial Intelligence, 1–12. Springer-Verlag.

Walther, C. 1983. A Many-Sorted Calculus Based on Resolution and Paramodulation. In A., B., ed., *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, 882–891.

Weidenbach, C.; Fietzke, A.; Kumar, R.; Suda, M.; Wischnewski, P.; and Dimova, D. 2009. SPASS Version 3.5. In Schmidt, R., ed., *Proceedings of the 22nd International Conference on Automated Deduction*, number 5663 in Lecture Notes in Artificial Intelligence, 140–145. Springer-Verlag.

Weidenbach, C. 2001. Combining Superposition, Sorts and Splitting. In Robinson, A., and Voronkov, A., eds., *Handbook of Automated Reasoning*. Elsevier Science. 1965–2011.

Wos, L.; Robinson, G.; and Carson, D. 1965. Automatic Generation of Proofs in the Language of Mathematics. In W.A., K., ed., *Proceedings of the IFIPS Congress*, 325–326. Sparton Books.