A Grand Challenge of Theorem Discovery

Geoff Sutcliffe[1], Yi Gao[1], and Simon Colton[2]

[1] Department of Computer Science, University of Miami
geoff@cs.miami.edu, yigao@mail.cs.miami.edu
[2] Department of Computing, Imperial College, London
sgc@doc.ic.ac.uk

Abstract. A primary mode of operation of ATP systems is to supply the system with axioms and a conjecture, and to then ask the system to produce a proof (or at least an assurance that there is a proof) that the conjecture is a theorem of the axioms. This paper challenges ATP to a new mode of operation, by which interesting theorems are generated from a set of axioms. The challenge requires solutions to both theoretical and computational issues.

1 Introduction

Automated Theorem Proving (ATP) deals with the development of computer programs that show that some statement (the conjecture) is a logical consequence of a set of statements (the axioms). ATP systems are used in a wide variety of domains: many problems in mathematics have been tackled with ATP techniques, software and hardware have been designed and verified using ATP systems, and applications to the WWW seem possible. In all of these applications, a primary mode of operation is to supply an ATP system with the axioms and a conjecture, and to ask the system to produce a proof (or at least an assurance that there is a proof) that the conjecture is a theorem of the axioms. Full automation of this task has been highly successful when the problem is expressed in classical 1st order logic, so that a proof by refutation of the clause normal form of the problem can be obtained. There are some well known high performance ATP systems that search for a refutation of a set of clauses, e.g., Gandalf [Tam], SPASS [WBH$^+$02], E [Sch02a], Vampire [RV02]. The Grand Challenge presented in this paper is suitable for ATP systems for any formal system, but will be presented in terms of ATP systems for classical 1st order logic, and henceforth all discussion is in that context.

ATP systems have made some noteworthy contributions to mathematics [SFS95,McC97], are used in real-world applications [Sti94,Sch02b,CHM02], and there is empirical evidence of progress in ATP [SFS01]. Successes of ATP systems have to be viewed in the context of the super-exponential growth of the search space of ATP systems, which is $O(NumberOfFormulae^{2^{searchdepth}})$. The successes are testiment to the heuristics that guide the search towards a proof (in refutation based systems, towards the derivation of a contradiction). Without such targeted search, one might suspect that ATP systems would be of little use to man or beast. Some might even claim that current state-of-the-art ATP systems are capable of proving only an exceptional few interesting theorems. While the huge search space of ATP systems is part of their challenge, it is also the source of the Grand Challenge presented in this paper.

The logical consequences of a set of (non-contradictory) axioms form the theory of those axioms. In such a theory there are many boring theorems, and scattered amoung them there are a few interesting ones. The few interesting ones include those that are *singled out* as theorems by human experts in the domain. Although humans identify many interesting theorems (of a given set of axioms), it seems inevitable that there are more out there. Our Grand Challenge is to *use ATP to generate and identify these unnoticed interesting theorems.*

If an ATP system is given a set of axioms (and infinite resources) it may generate (all) the theorems of the axioms. In order to output only interesting theorems, modifications will be necessary. The modifications may be in the form of internal guidance to prevent or reduce the generation of boring theorems, or in the form of post-processing filters that identify interesting theorems in the output stream. If such modifications can be designed and implemented, the resultant tool may be able to discover new interesting theorems of a set of axioms. Such a system will be proactive in the theorem discovery and proof process, rather than reactive, as ATP systems are now. They will, it might be said, become part of the problem rather than part of the solution.

The notion and a drive for automated discovery in science, including the use of ATP systems, is not new [Lan98,Col01]. Newell and Simon predicted [SN58] that a computer would discover and prove an important maths theorem, Alan Bundy placed discovery in his "DReaM" (the acronym for his research group, "**D**iscovery and **Rea**soning in **M**athematics"), and Bob Kowalski has expressed the opinion that it is "more important to discover the right theorems than to prove unimportant ones" [Kow]. ATP systems have been used, e.g., to discover proofs of open conjectures [McC97,Sla02], to discover new axiomatizations of theories [Hod98,Wos01,MVF$^+$02], to investigate plane geometry [BSZC93], and to test automatically generated conjectures in mathematics [Zha99,Col02]. In all these applications the ATP systems have been used as assistants to prove conjectures that are generated using other techniques. That contrasts with the approach proposed in this Grand Challenge, in which the ATP systems will themselves discover new theorems, and other techniques may be used to determine whether or not the theorems are interesting.

This Grand Challenge of discovering new interesting theorems is not one that can be easily met (see Section 3). However, any short term successes can make an immediate contribution to the advancement of ATP: The development and testing of ATP systems is somewhat dependent on the availability of test problems that are somehow representative of applications. At present the TPTP problem library [SS98] is a de facto standard for testing ATP systems (it includes, amoung others, many theorems that have been idenitifed as interesting by humans). In order for the TPTP to continue to be effective it is necessary to regularly add new problems to the TPTP, so that systems do not become over-tuned to solving problems in the TPTP. Since the first release of the TPTP in 1993 it has grown from 2295 problems in 23 domains, to the current 6672 problems in 30 domains. Despite this reasonable long-term growth, some years have been leaner than others, and consistent significant growth would make the TPTP a more valuable resource. Any new theorems produced in partial answer to this Grand Challenge can be added to the TPTP, which will be of immediate benefit to ATP system developers. It is necessary that the theorems be difficult (in the TPTP sense [SS01]) for state-of-the-art ATP systems.

2 Automatic Conjecture and Theorem Creation

There are at least four ways in which new conjectures and theorems can be created, including the approach proposed in Section 1.

The first approach, the *inductive* approach, is often used by humans. Upon observing many similar events, people often induce a general conjecture. For example, noting that the prime numbers 5, 7, 11, 13 are all odd, the general conjecture that all prime numbers are odd may be made. An attempt is then made to prove (or disprove) the conjecture, and if disproved, modifications may be made, e.g., that all prime numbers greater than 2 are odd. Such reparation of faulty conjectures is described in [Lak76], and a project to automate such tasks is described in [PCSL01]. The inductive approach has the advantage of being stimulated by observations in reality, but has the disadvantage that the rule used to produce the conjectures, induction, is unsound. As a result, many of the conjectures produced are non-theorems, and effort is expended finding disproofs. In some cases a disproof many be hard to find, and a conjecture remains open, possibly considered in folklore to be a theorem, for a considerable period. An example is the conjecture that $P \neq NP$.[1]

The second approach, the *generative* approach, is an extension of the inductive approach. There are various ways in which the generative approach can proceed. The simplest form of generation is syntactic, in which conjectures are created by mechanical manipulation of symbols, e.g., [Pla94]. The MCS system [Zha99] generates conjectures syntactically and filters them against models of the domain. A more intelligent, semantically based, approach is taken by the HR system [Col02]. HR starts with an initial set of concepts – supplied with both a set of examples and a definition – and some axioms that relate the concepts. It then iteratively invents new concepts based on previous ones, and uses the examples of concepts to check for empirical relationships between the concepts. Such relationships are stated as conjectures and are passed along with the axioms to an ATP system to test for theoremhood. Conjectures that pass the filtering are sent to an ATP system to test for theoremhood. Like induction, generation is unsound. However, if the rules by which the generation is performed are sufficiently conservative then this approach may generate a higher fraction of theorems than the inductive approach. The effectiveness of just the

[1] One could argue that unproved conjectures found inductively, such as Goldbach's conjecture and Fermat's last theorem, are very advantageous to mathematics.

generation functions used in HR is illustrated by an empirical study in which all 46186 group theory conjectures generated were proved to be theorems, and of those 184 were added to the TPTP [CS02]. Note that this was just an initial study of HR's ability to find conjectures of difficulty for theorem provers, and no measures were used to filter the conjectures. A more sophisticated application of HR to conjecture generation is described in [ZFCS02].

The third approach, the *manipulative* approach, generates conjectures from existing theorems. An existing theorem is modified by operations such as generalization, specialization, combination, etc. This approach is used in abstraction mapping, which converts a problem to a simpler problem, and uses a solution to the simpler problem to help find a solution of the original problem [Pla80]. Manipulation of ATP problems has also been used to produce new problems for testing the robustness of ATP systems' performances [Vor00]. An advantage of the manipulative approach is that if the manipulations are satisfiability preserving, then theorems, rather than conjectures, are produced from existing theorems. However, the conjectures produced by the manipulative approach are typically artificial in nature, and therefore uninteresting.

The fourth approach, the *deductive* approach, is that proposed in this paper. It generates only theorems consequences, and may or may not use internal mechanisms to guide the generation towards interesting theorems. The advantage of this approach is that every output is known to be a theorem. The disadvantage is that many boring theorems will be generated. A major part of the challenge is to overcome this tedium, so that boring theorems are either not generated or are discarded internally. The deductive approach, like many functions performed by computers, is one that cannot be realistically adopted by humans. The large number of logical consequences that need to be considered would swamp the ponderous human brain. It is only the capability of high-speed computing that makes this approach viable. To paraphrase Emma Lazarus' words [Laz83], computers happily accept "your boring, your trivial, your huddled theorems who yearn TPTP".

3 Attacking the Challenge

Using the deductive approach to discover new interesting theorems is a significant computational challenge. Goal directed theorem proving, e.g., the search for the empty clause in CNF refuation based ATP, provides some obvious opportunities for pruning and ordering the search space. In contrast, pruning and ordering so as to ignore boring theorems and to discover interesting theorems early in the search, seems to be a more difficult task - if ATP systems have any difficulty with deducing a target formula, removing the target is going to make things worse. Interesting theorems may be widely distributed in the space of logical consequences of a set of axioms, and a search through a larger fragment of that space seems likely to be necessary.

As a basis for a first attack on this challenge, a filtering approach is being implemented. It is already possible to have a CNF based ATP system generate a stream of logical consequences, filtered so that no generated formula is subsumed by an earlier one. These logical consequences can then be assessed for interestingness. Several filters are being considered:

Non-obviousness requires a theorem to be reasonably difficult to prove. This can be measured from the size of its proof tree, or the time taken for its proof. It may be the case that the derivation that created a theorem is significantly non-minimal. A directed search for a smaller derivation may lead to a theorem being given a higher measure of obviousness than it initially received.

Novelty requires that a theorem is non-tautologous and non-redundant with respect to other theorems and the axioms. This includes subsumption checking, and ensuring that a theorem is not easily proved from the axioms and any of its descendants.

Surprisingness measures new relationships between concepts. In the logic formalism, one way to measure is to examine the extent to which new combinations of predicate and functions symbols occur in a theorem.

Intensity measures the extent to which a theorem summarizes the information contained in the axioms from which it is deduced. A naive possibility is to consider the ratio of the symbol count of the theorem and the symbol count of the axioms (symbol counting is a common heuristic used for evaluating the quality of a clause in ATP systems [SM96]). More complex measures involve consideration of the structure of the proof tree leading to the logical consequence. One idea which seems to warrant further attention is to rate a formula by the average branching factor of the formula's proof tree. This measures "bushiness" of the proof tree. Initial empirical tests of this measure against the instincts of a mathematician seems to correlate bushiness with interestingness. A related notion is the *axiom dependency* of a theorem, which measures the number of axioms required to prove the theorem. Given a set of axioms for a domain, a particular theorem may be provable using only a subset of those axioms. For instance, in group theory, various theorems are provable using only the identity axiom. One could argue that such theorems are more general, hence more interesting. However, it seems more likely that if you were using a theorem generation program to discover theorems in group theory, then the theorems which are not true of a more general algebra would be more interesting. A good example of a theorem in group theory requiring all the axioms to prove it is: $\forall a \ (a * a = a \leftrightarrow a = id)$.

Usefulness measures how much a theorem may contribute to the proof of further theorems, i.e, its usefulness as a lemma. There have been several efforts to identify relevant lemmas produced by ATP systems, especially for model elmination based systems, e.g., [Fuc00,DS01]. The focus of those efforts is different from that here, their aim being to identify lemmas that will contribute to the completion of a search for a refutation. Despite this difference, it may be possible to adapt the underlying ideas to identifying interesting theorems. The identification of interesting lemmas has also been investigated for proof presentation purposes [DS96]. Another way to measure usefulness of implications may be to generate a suite of models of the axioms, and determine the fraction of models that make the antecedent true.

Comprehensibility estimates the effort required for a user to understand the theorem. Theorems with many or deeply nested structures may be considered incomprehensible.

Applicability measures the number of objects of interest to which a theorem applies. For instance, suppose we have this number theory theorem:

$$\forall X \, (iseven(X) \land isprime(X) \rightarrow isodd(sum\_of\_divisors\_of(X)))$$

The left hand side of this theorem states that X is an even prime, which is only true of the number 2. Hence this theorem scores low for applicability because it boils down to saying that the number 3 (the sum of divisors of 2) is odd.

Interestingness in theorem discovery is a highly subjective thing. The final arbitration as to the interestingness of a theorem must be left to human experts. It will be necessary to be *very* selective about which theorems are presented for consideration, as a stream of boring theorems will soon leave the arbitrators disillusioned, and will leave the generator without arbitrators. The difficulty will include *predicting* whether a theorem will turn out to be interesting. The worth of a theorem may not become obvious for some time, perhaps even years, and theorem generation programs have to predict this worth. Such prediction is notoriously difficult.[2]

Most of the above measures of interestingness apply as much to open conjectures as to proved theorems. Moreover, measures such as novelty, surprisingness, usefulness, comprehensibility, axiom dependency and applicability have been identified elsewhere in the literature (in particular [CBW00]). Indeed, these measures have been implemented in the HR system, as described in chapter 10 of [Col02].

Given the above considerations, Figure 1 shows a reasonable architecture for a system that discovers interesting theorems. The logical consequences generated by the ATP system are filtered at runtime to reduce the number to a manageable level. Techniques used here have to be very quick so as to cope with the stream of production. They include requiring a minimal proof tree size (an aspect of non-obviousness), minimal intensity, and exclusion of easily identified tautologies (an aspect of novelty). The results are stored in secondary storage. When sufficient logical consequences have been stored the ATP system is stopped, and static filters are used to reduce the number of stored formulae. Techniques used here include further non-obviousness, applicability, and novelty checks. Once filtered the logical consequences are ranked in order of interestingness for presentation to the user. Measures used here include intensity, suprisingness, comprehensibility, and usefulness.

4 Conclusion

A Grand Challenge for ATP has been described: *Use ATP systems to discover interesting theorems of the axioms of a domain.* Additionally, for theorems to be

[2] Would Diophantine equations have been so studied if Fermat had left a proof of his so-called "Last Theorem" in the margin of his book?

Fig. 1. System Architecture

added to the TPTP, they should be difficult for ATP systems. This challenge can be approached incrementally:

– Generate theorems.
– Generate theorems that are difficult for ATP systems to prove.
– Generate theorems that are interesting to humans.
– Generate theorems that are interesting to humans and difficult for ATP systems to prove.
– Generate theorems that are interesting to humans and difficult for humans and ATP systems to prove.

Progress past the first of these is certainly possible, and progress to the last will be a wonderful success.

The UK Computing Research Committee has proposed a set of criteria for assessing grand challenges in computing [GC-], which can be applied to grand challenges in automated reasoning. Does this Grand Challenge meet these?

1. It arises from scientific curiosity about the foundation, the nature or the limits of the discipline.
   Although some theorems are proved out of commercial or other mundane need, the search for interesting theorems is largely a scientific endeavour. The theorems that may be generated by ATP in response to this challenge are not specifically of interest to ATP research. Rather, it is a challenge to see whether or not such theorems can be generated automatically.
2. It gives scope for engineering ambition to build something that has never been seen before.

The development of ATP systems is not new, and some of the techniques proposed for filtering out boring theorems have been tried and tested. The use of these in combination to find interesting theorems is new. Further, the development of techniques that will internally guide ATP systems towards interesting theorems will be a new feature of ATP systems.

3. It will be obvious how far and when the challenge has been met (or not).
   Human arbitration of the interestingness of the theorems produced will measure progress and success.
4. It has enthusiastic support from (almost) the entire research community, even those who do not participate and do not benefit from it.
   Not yet ... in fact, only a tiny proportion of automated reasoning researchers have addressed the question of discovering theorems, rather than proving known theorems.
5. It has international scope: participation would increase the research profile of a nation.
   Not yet ...
6. It is generally comprehensible, and captures the imagination of the general public, as well as the esteem of scientists in other disciplines.
   The idea is simple, and comprehensible to anyone who has ever had to prove a high school geometry theorem. The possibility of a computer inventing something new has long been a matter of debate between proponents and critics of artificial intelligence [Dre79]. Success here would weigh heavily in that debate, and would be of direct interest to the AI community and to those from the domain of the theorems.
7. It was formulated long ago, and still stands.
   Automated discovery in science is not a new idea (although I came up with this specific instance in the shower last week), and the challenge of generating theorems has been identified previously, most notably in [Col01].
8. It promises to go beyond what is initially possible, and requires development of understanding, techniques and tools unknown at the start of the project.
   At this stage only the first level of challenge (as itemized above) can be met. Current understanding of what makes a theorem interesting, rather than simply true, will be extended and refined. Effective implmentations of filtering and search techniques will have to be developed. Appropriate interfaces between ATP systems, filtering systems, and human arbitrators, will have to be designed.
9. It calls for planned co-operation among identified research teams and communities.
   There is scope and necessity for cooperation between the ATP community and experts from the domains of application.
10. It encourages and benefits from competition among individuals and teams, with clear criteria on who is winning, or who has won.
    The organizers of the annual CADE ATP System Competition [Nie02] have discussed the design of a competition for the submission of the "most interesting ATP problem". Some of the adjudication would be subjective, and done by human experts, while some would be based on empirical testing of the

problems on ATP systems. These ideas may be further developed to evaluate success in meeting this challenge.

11. It decomposes into identified intermediate research goals, whose achievement brings scientific or economic benefit, even if the project as a whole fails.
    The application of existing criteria, and the development of new criteria, for recognizing a theorem as "interesting" will be interesting in their own right, even if they cannot be used successfully to filter the output of ATP systems. A deeper understanding of the structure of ATP systems' search spaces, resulting from attempts to guide ATP systems towards generating interesting theorems, would be valuable to ATP in general.
12. It will lead to radical paradigm shift, breaking free from the dead hand of legacy.
    Using ATP systems to generate theorems, rather than find proofs for theorems, is a departure from their common usage.
13. It is not likely to be met simply from commercially motivated evolutionary advance.
    Current commercial applications of ATP aim to find proofs and models. At this point in time there is no apparent commercial demand for the generation of interesting theorems.

Working on this challenge will be a pleasure and a virtue, because, as Bob Boyer remarked about working on open problems, it is impossible to cheat.

References

[BSZC93] R. Bagai, V. Shanbhogue, J.M. Zytkow, and S-C. Chou. Automatic Theorem Generation in Plane Geometry. In H.J. Komorowski and Z.W. Ras, editors, *Proceedings of the 7th International Symposium on Methodologies for Intelligent Systems*, number 689 in Lecture Notes in Artificial Intelligence, pages 415–424. Springer-Verlag, 1993.

[CBW00] S. Colton, A. Bundy, and T. Walsh. On the Notion of Interestingness in Automated Mathematical Discovery. *International Journal of Human Computer Studies*, 53(3):351–375, 2000.

[CHM02] K. Claessen, R. Hähnle, and J. Mårtensson. Verification of Hardware Systems with First-Order Logic. In G. Sutcliffe, J. Pelletier, and C. Suttner, editors, *Proceedings of the CADE-18 Workshop - Problem and Problem Sets for ATP*, number 02/10 in Department of Computer Science, University of Copenhagen, Technical Report, 2002.

[Col01] S. Colton. Automated Theorem Generation: A Future Direction for Theorem Provers. In M. Kerber, editor, *Proceedings of the IJCAR 2001 workshop: Future Directions in Automated Reasoning*, 2001.

[Col02] S. Colton. *Automated Theory Formation in Pure Mathematics.* Springer-Verlag, 2002.

[CS02] S. Colton and G. Sutcliffe. Automatic Generation of Benchmark Problems for Automated Theorem Proving Systems. In B. Faltings, editor, *Proceedings of the 7th International Symposium on Artificial Intelligence and Mathematics*, pages AI–M 4–2002., 2002.

[Dre79] H. Dreyfus. *What Computers Can't Do: The Limits of Artificial Intelligence*. Harper collins, 1979.

[DS96] J. Denzinger and S. Schulz. Recording and Analysing Knowledge-Based Distributed Deduction Processes. *Journal of Symbolic Computation*, 21:523–541, 1996.

[DS01] J. Draeger and S. Schulz. Improving the Performance of Automated Theorem Provers by Redundancy-free Lemmatization. In I. Russell and J. Kolen, editors, *Proceedings of the 14th Florida Artificial Intelligence Research Symposium*, pages 345–349. AAAI Press, 2001.

[Fuc00] M. Fuchs. Controlled Use of Clausal Lemmas in Connection Tableau Calculi. *Journal of Symbolic Computation*, 29(2):299–341, 2000.

[GC-] Grand Challenges for Computing Research. http://umbriel.dcs.gla.ac.uk/NeSC/general/esi/events/Grand_Challenges/.

[Hod98] K. Hodgson. Shortest Single Axioms for the Equivalential Calculus with CS and RCD. *Journal of Automated Reasoning*, 20(3):283–316, 1998.

[Kow] R.A. Kowalski. A Short Story of My Life and Work. http://www-lp.doc.ic.ac.uk/UserPages/staff/rak/history.html.

[Lak76] I. Lakatos. *Proofs and Refutations: The Logic of Mathematical Discovery*. Cambridge University Press, 1976.

[Lan98] P. Langley. The Computer-Aided Discovery of Scientific Knowledge. In S. Arikawa and A. Motoda, editors, *Proceedings of the 1st International Conference on Discovery Science*, number 1532 in Lecture Notes in Computer Science, pages 25–39. Springer-Verlag, 1998.

[Laz83] E. Lazarus. The New Colossus. 1883.

[McC97] W.W. McCune. Solution of the Robbins Problem. *Journal of Automated Reasoning*, 19(3):263–276, 1997.

[MVF$^+$02] W.W. McCune, R. Veroff, B. Fitelson, K. Harris, A. Feist, and L. Wos. Short Single Axioms for Boolean Algebra. *Journal of Automated Reasoning*, 29(1):1–16, 2002.

[Nie02] R. Nieuwenhuis. Special Issue: The CADE ATP System Competition. *AI Communications*, 15(2-3), 2002.

[PCSL01] A. Pease, S. Colton, A. Smaill, and J. Lee. A Multi-Agent Approach to Modelling Interaction in Human Mathematical Reasoning. In *Proceedings of the 2001 International Conference on Intelligent Agent Technology*, Intelligent Agent Technology: Research and Development. World Scientific, 2001.

[Pla80] D.A. Plaisted. Abstraction Mappings in Mechanical Theorem Proving. In W. Bibel and R. Kowalski, editors, *Proceedings of the 5th International Conference on Automated Deduction*, number 87 in Lecture Notes in Computer Science, pages 264–280. Springer-Verlag, 1980.

[Pla94] D.A. Plaisted. The Search Efficiency of Theorem Proving Strategies. In A. Bundy, editor, *Proceedings of the 12th International Conference on Automated Deduction*, number 814 in Lecture Notes in Artificial Intelligence, pages 57–71. Springer-Verlag, 1994.

[RV02] A. Riazanov and A. Voronkov. The Design and Implementation of Vampire. *AI Communications*, 15(2-3):91–110, 2002.

[Sch02a] S. Schulz. E: A Brainiac Theorem Prover. *AI Communications*, 15(2-3):111–126, 2002.

[Sch02b] J. Schumann. *Automated Theorem Proving in Software Engineering*. Springer-Verlag, 2002.

[SFS95] J.K. Slaney, M. Fujita, and M.E. Stickel. Automated Reasoning and Exhaustive Search: Quasigroup Existence Problems. *Computers and Mathematics with Applications*, 29(2):115–132, 1995.

[SFS01] G. Sutcliffe, M. Fuchs, and C. Suttner. Progress in Automated Theorem Proving, 1997-1999. In H. Hoos and T. Stützle, editors, *Proceedings of the IJCAI'01 Workshop on Empirical Methods in Artificial Intelligence*, pages 53–60, 2001.

[Sla02] J.K. Slaney. More Proofs of an Axiom of Lukasiewicz. *Journal of Automated Reasoning*, 29(1):59–66, 2002.

[SM96] G. Sutcliffe and S. Melville. The Practice of Clausification in Automatic Theorem Proving. *South African Computer Journal*, 18:57–68, 1996.

[SN58] H. Simon and A. Newell. Heuristic Problem Solving: The Next Advance in Operations Research. *Operations Research*, 6(1):1–10, 1958.

[SS98] G. Sutcliffe and C.B. Suttner. The TPTP Problem Library: CNF Release v1.2.1. *Journal of Automated Reasoning*, 21(2):177–203, 1998.

[SS01] G. Sutcliffe and C.B. Suttner. Evaluating General Purpose Automated Theorem Proving Systems. *Artificial Intelligence*, 131(1-2):39–54, 2001.

[Sti94] M.E. Stickel. The Deductive Composition of Astronomical Software from Subroutine Libraries. In A. Bundy, editor, *Proceedings of the 12th International Conference on Automated Deduction*, number 814 in Lecture Notes in Artificial Intelligence, pages 341–355. Springer-Verlag, 1994.

[Tam] T. Tammet. Gandalf Family of Automated Theorem Provers. http://www.cs.chalmers.se/ tammet/gandalf/.

[Vor00] A. Voronkov. CASC-16 1/2. Technical Report CSPP-4, Department of Computer Science, University of Manchester, Manchester, England, 2000.

[WBH$^+$02] C. Weidenbach, U. Brahm, T. Hillenbrand, E. Keen, C. Theobald, and D. Topic. SPASS Version 2.0. In A. Voronkov, editor, *Proceedings of the 18th International Conference on Automated Deduction*, number 2392 in Lecture Notes in Artificial Intelligence, pages 275–279. Springer-Verlag, 2002.

[Wos01] L. Wos. Conquering the Meredith Single Axiom. *Journal of Automated Reasoning*, 27(2):175–199, 2001.

[ZFCS02] J. Zimmer, A. Franke, S. Colton, and G. Sutcliffe. Integrating HR and tptp2X into MathWeb to Compare Automated Theorem Provers. In G. Sutcliffe, J. Pelletier, and C.B. Suttner, editors, *Proceedings of the CADE-18 Workshop - Problem and Problem Sets for ATP*, number 02/10 in Department of Computer Science, University of Copenhagen, Technical Report, 2002.

[Zha99] J. Zhang. System Description: MCS: Model-Based Conjecture Searching. In H. Ganzinger, editor, *Proceedings of the 16th International Conference on Automated Deduction*, number 1632 in Lecture Notes in Artificial Intelligence, pages 393–397. Springer-Verlag, 1999.