Integration of the TPTPWorld into SigmaKEE

Steven Trac[1], Geoff Sutcliffe[1], and Adam Pease[2]

[1]University of Miami, USA [2]Articulate Software, USA

Abstract. This paper describes the integration of the ATP support of the TPTPWorld into the Sigma Knowledge Engineering Environment. The result is an interactive knowledge based reasoning environment, with strong knowledge management features, and access to modern state of the art ATP systems for reasoning over knowledge bases.

1 Introduction

The Knowledge Based Reasoning (KBR) community within the field of Artificial Intelligence has long conducted logical reasoning for decision support, planning and many similar applications. The Automated Theorem Proving (ATP) community has grown more out of mathematical disciplines, and its applications have tended to be in that realm. While the various uses of SNARK [12], e.g., [13, 21], are notable exceptions, there has not been significant use of ATP in KBR. This work brings together the KBR tool SigmaKEE and the ATP support of the TPTPWorld. The Sigma Knowledge Engineering Environment (SigmaKEE) [7] provides a mature platform for browsing and querying a knowledge base, often the Suggested Upper Merged Ontology (SUMO) [6]. The TPTPWorld provides well established standards, systems, and tools for first-order reasoning, stemming from the Thousands of Problems for Theorem Provers (TPTP) problem library [16]. While SigmaKEE has strong knowledge management features, it lacks the reasoning capabilities found in state of the art ATP systems. Conversely, while modern ATP systems are capable of proving hard theorems, they have limited features for interfacing with users of large knowledge bases. The integration of the TPTPWorld into SigmaKEE forms an interactive KBR environment, with strong knowledge management features, and access to modern state of the art ATP systems for reasoning over knowledge bases.

This paper is organized as follows: Sections 2 and 3 provide the necessary background about SigmaKEE and the TPTPWorld. Section 4 describes their integration, including extensions added to meet the needs of SigmaKEE users. Section 5 shows a sample use of the integrated system.

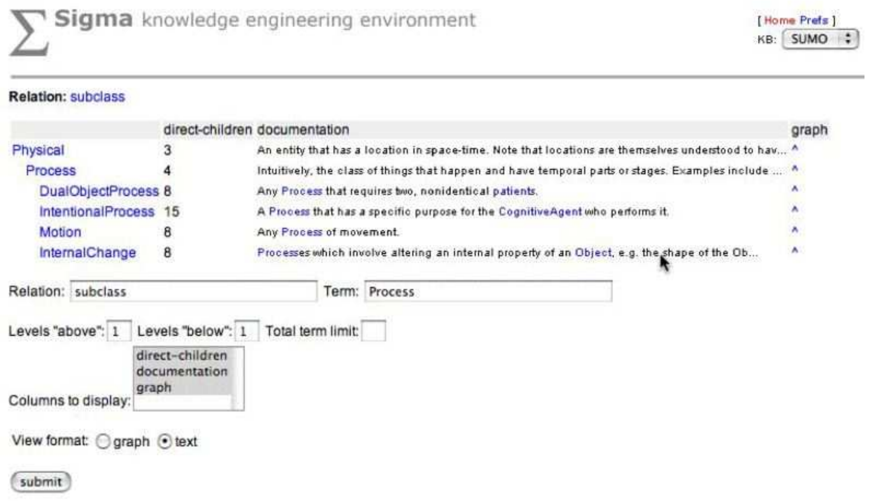
2 SigmaKEE

The Sigma Knowledge Engineering Environment (SigmaKEE) is a KBR environment for developing and using logical theories. It was created to support the Suggested Upper Merged Ontology (SUMO), which is written in a variant of the Knowledge Interchange Format (KIF) language [4] called Standard Upper Ontology Knowledge Interchange Format (SUO-KIF) [7]. SigmaKEE runs as an Apache Tomcat service, providing a browser interface to users. The main components are written in Java, and the user interface is generated by JSP. Users can upload a knowledge base for browsing and querying. An uploaded knowledge base is indexed for high performance browsing and searching. For ontology-like knowledge bases, which have a tree structure, a graph browser is provided. Figure 1 shows the graph browser interface for the top layers of the SUMO. Results from queries are presented in a hyperlinked KIF format that provides linkages back into the knowledge base, as shown in the example in Section 5.

The existing version of SigmaKEE includes a customized version of Vampire [10]. Among state of the art first order logical theorem provers available at the time of SigmaKEE's original development, only this version of Vampire, which is now 5 years old, had all the features required for theorem proving applications in SigmaKEE:

– The ability to extract an answer to a query as bindings of outermost existentially quantified variables in a conjecture.
– The ability to generate a detailed proof that explains how an answer to a query was derived.
– The ability to ask successive queries without reloading the knowledge base.
– The ability to perform basic arithmetic.

In addition, that version of Vampire was released under an approved open source license, and could therefore be tightly integrated with the open source SigmaKEE system.



Relation: subclass

	direct-children	documentation	graph
Physical	3	An entity that has a location in space-time. Note that locations are themselves understood to hav...	^
Process	4	Intuitively, the class of things that happen and have temporal parts or stages. Examples include ...	^
DualObjectProcess	8	Any Process that requires two, nonidentical patients.	^
IntentionalProcess	15	A Process that has a specific purpose for the CognitiveAgent who performs it.	^
Motion	8	Any Process of movement.	^
InternalChange	8	Processes which involve altering an internal property of an Object, e.g. the shape of the Ob...	^

Relation: subclass Term: Process

Levels "above": 1 Levels "below": 1 Total term limit:

Columns to display: direct-children documentation graph

View format: graph [✓ text]

Fig. 1. SigmaKee Graph Browser

3 TPTPWorld

The TPTPWorld is a package of TPTP data and software, including the TPTP problem library, a selection of ATP systems, and a suite of tools for processing TPTP format data. Although the TPTPWorld was developed and is primarily used for inhouse maintenance of the TPTP problem library, various components have become publically available and used in applications, e.g., [14, 19].

One of the keys to the success of the TPTP and related projects is their consistent use of the TPTP language [15]. The TPTP language was designed to be suitable for writing both ATP problems and ATP solutions, to be flexible and extensible, and easily processed by both humans and computers. The TPTP language BNF is easy to translate into parser-generator (`lex/yacc`, `antlr`, etc.) input [20]. The SZS ontology [17] provides a fine grained ontology of result and output forms for ATP systems, so that their results can be precisely understood when used as input to other tools. The ontology also recommends the way that ontology values should be reported in the output from systems and tools. Figure 2 shows an extract from the top of the result ontology (the full ontology is available as part of the TPTP distribution).

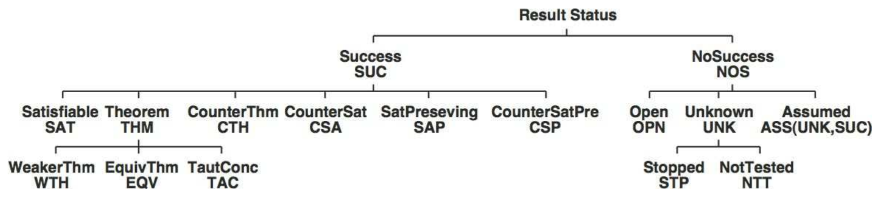


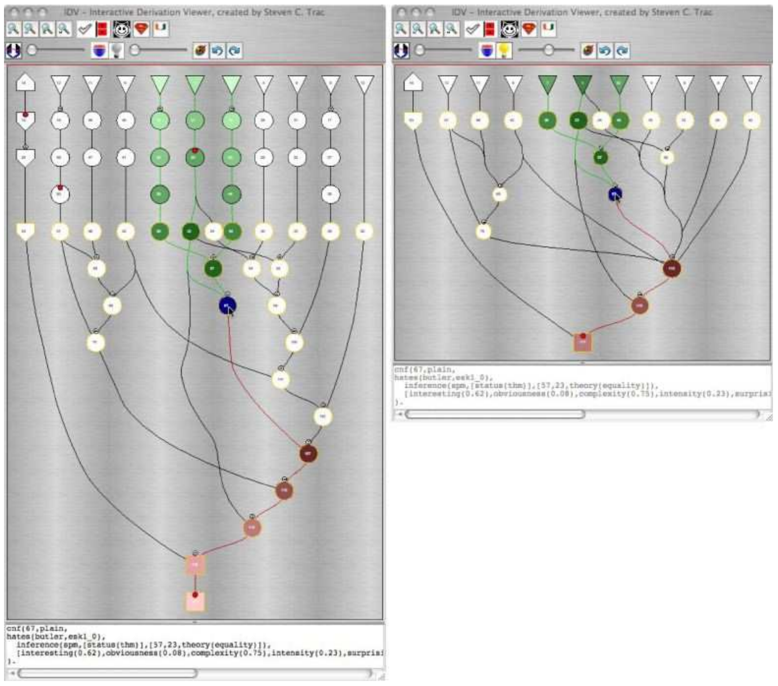
Fig. 2. SZS Ontology

The SystemOnTPTP utility is a harness that allows a problem written in the TPTP language to be easily and quickly submitted to a range of ATP systems and other tools. The implementation of SystemOnTPTP uses several subsidiary tools to prepare the input for the ATP system, control the execution of the chosen ATP system, and postprocess the output to produce an SZS result value and a TPTP format derivation. SystemOnTPTP runs in a UNIX environment, and is also available as an online service via `http POST` requests.[1]

The Interactive Derivation Viewer (IDV) [18] is a tool for graphical rendering and interaction with TPTP format derivations. The rendering uses shape and color to provide visual information about a derivation. The user can interact with the rendering in various ways – zooming, hiding, and displaying parts of the DAG according to various criteria, access to verification of the derivation, and an ability to provide a synopsis of a derivation by identifying interesting

[1] Hosted at the University of Miami. A browser interface to the service is available at `http://www.tptp.org/cgi-bin/SystemOnTPTP`.

lemmas using AGInT [9]. Figure 3 shows the renderings of the derivation and synopsis for the proof output by EP [11] for the TPTP problem PUZ001+1.


IDV – Interactive Derivation Viewer, created by Steven C. Trac
IDV – Interactive Derivation Viewer, created by Steven C. Trac

```
cnf(67,plain,
hates(butler,esk1_0),
  inference(spm,[status(thm)],[57,23,theory(equality)]),
  [interesting(0.62),obviousness(0.08),complexity(0.75),intensity(0.23),surprisi
).
```

```
cnf(67,plain,
hates(butler,esk1_0),
  inference(spm,[status(thm)],[57,23,theory(equality)]),
  [interesting(0.62),obviousness(0.08),complexity(0.75),intensity(0.23),surprisi
).
```

Fig. 3. EP's Proof by Refutation of PUZ001+1

The One Answer Extraction System (OAESys) is a tool for extracting the bindings for outermost existentially quantified variables of a conjecture, from a TPTP format proof of the conjecture. This is done by reproving the conjecture using the Metis system [5], from only the axioms used in the original proof. The variable bindings that Metis reports for each inference step of its proof are analyzed to extract the required bindings (Metis is the only system that we know of that outputs TPTP-compliant proofs and variable bindings for each inference step). The restriction to the axioms used in the original proof makes it highly unlikely for Metis to find a proof that provides different variable bindings to the original proof. If the axioms used are a subset of the axioms that were originally available, the problem given to Metis could be significantly easier than the original problem.

The Multiple ANSwer EXtraction (MANSEX) system is a framework for interpreting a conjecture with outermost existentially quantified variables as a question, and extracting multiple answers to the question by repetitive calls to an ATP system that can report the bindings for the variables in one proof of the conjecture.[2] Suitable ATP systems are SNARK, and a combination of EP and OAESys. At each iteration of MANSEX, the conjecture is augmented by conjoining either inequalities or negative atoms that deny previously extracted answers. The ATP system is then called again to find a proof of the modified conjecture. In the SigmaKEE context the process has been extended to hide the conjecture modifications from the user - details are provided in Section 4.

The TPTP-parser is a highly reusable Java parser for TPTP data, built using the `antlr` parser-generator.[3] The parser can easily be used without modifications in practically any application. This universality is achieved by isolating the parser code by an interface layer that allows creation of and access to abstract syntax representations of various TPTP elements. A simple but reasonably efficient default implementation of the interface layer is provided with the package.

4 Integration of the TPTPWorld into SigmaKEE

The integration of the TPTPWorld provides SigmaKEE with new capabilities:

- Internal support for TPTP format problems and derivations, using a SUO-KIF to TPTP translation and the the TPTP-parser.
- Access to ATP systems for reasoning tasks, using SystemOnTPTP.
- Question answering using OAESys, with the ability to provide multiple answers through use of the MANSEX framework.
- Presentation of TPTP format proofs using IDV, or using the existing hyper-linked KIF format extended with SZS ontology status values.
- An extended browser interface for access to these capabilities.

The integration has been implemented by adding external TPTPWorld tools to the SigmaKEE distribution, and embedding Java implementations of TPTPWorld tools directly into SigmaKEE.

SigmaKEE was developed to support knowledge bases written in SUO-KIF, e.g., SUMO. In order to make a large suite of ATP systems available for reasoning over such knowledge bases, through use of the SystemOnTPTP utility, knowledge bases are translated to the TPTP language when they are loaded. While much of the translation is syntactic, there are some constructs in SUMO that require special processing [8]. These include use of sort signatures, sequence variables, variable predicates and functions, and embedded (higher-order) formulae.

Once a knowledge base has been loaded into SigmaKEE, queries can be submitted. A query is translated to a TPTP format conjecture, and the previously translated knowledge base provides the axioms. These are submitted to an ATP system through the SystemOnTPTP utility. Queries with outermost existentially quantified variable are treated as questions whose answers are the values bound

[2] **Acknowledgement:** The original multiple answer extraction system was developed outside the SigmaKEE project by Aparna Yerikalapudi, at the University of Miami.

[3] **Acknowledgement:** The TPTP-parser was written primarily by Andrei Tchaltsev at ITC-irst. It is available from `http://www.freewebs.com/andrei_ch/`

to those variables in proofs. Three versions of SystemOnTPTP are available: remote access to the online SystemOnTPTP service via `http POST` requests, execution of a locally installed SystemOnTPTP, and a limited internal implementation of SystemOnTPTP. The ATP systems supported by the internal implementation are required to be TPTP-compliant in terms of both input and output, and have licensing that allows them to be added to SigmaKEE. At this stage E/EP, Metis, SNARK, and Paradox [3] are being used.

The advantage of using the local installation or internal implementation of SystemOnTPTP is that they do not rely on an online connection to the remote server. The advantage of the internal implementation is that it is portable to operating systems that do not have the UNIX environment required for SystemOnTPTP, e.g., Windows XP. The user chooses whether to use the remote SystemOnTPTP or a local one, and within that which ATP system to use. In the remote case the online system is queried to get a list of the available ATP systems. In the local case the ATP systems available in the local SystemOnTPTP installation (if any) and the internal implementation are available. If an ATP system is supported by the internal implementation and is also available through a local SystemOnTPTP installation, the internally supported one is used.

Answers to "question" conjectures are extracted from proofs using an embedding of OAESys into SigmaKEE. As Metis is one of the internally supported systems, it is available for use in OAESys. When the user requests more than one answer, an embedding of the MANSEX framework is used. In the SigmaKEE context the MANSEX process has been extended to hide the conjecture modifications from the user. This extension is done for the second and subsequent proofs found, as follows. After each answer has been extracted by OAESys, the existentially quantified variables in the original conjecture, i.e., the conjecture without the augmentations, are instantiated with the answer values. This instantiated conjecture and just the axioms used in the proof found by the chosen ATP system are passed to that ATP system. This this additional ATP system run finds a proof of the (instantiated form of the) original conjecture, rather than of the augmented conjecture. The use of MANSEX to get multiple answers is somewhat different to use of the customized version of Vampire mentioned in Section 2. MANSEX with OAESys requires multiple ATP system runs: two for the first answer (one to get a proof using the chosen ATP system and another to Metis within OAESys), and three for each successive answer (additionally the final call to the chosen ATP system). In contrast, the customized Vampire backtracks in its proof space to find multiple answers. As a side-effect, the customized Vampire can return the same answer multiple times if there are multiple proofs that produce the same variable bindings, while MANSEX does not.

The integration of the TPTPWorld provides SigmaKEE with three options for displaying results: TPTP format derivations in plain text format, IDV for displaying TPTP format derivations graphically, and the hyperlinked KIF format. IDV has been embedded into SigmaKEE so that it is directly available. The hyperlinked KIF format has been implemented by translation of TPTP format derivations into SUO-KIF using an augmentation of the TPTP-parser code, and then calling SigmaKEE's hyperlinked KIF presentation feature. The hyperlinked KIF format has been mildly extended to provide more precise information about the formulae in a proof, and to provide SZS status values. An example with a hyperlinked KIF format proof is given in Section 5.

The top part of Figure 4 shows the GUI interface. The interface allows the user to submit a query or to add to the current knowledge base. The interface has the following components (top to bottom, left to right):

- Formula text box - The query or additions are put into this text box in SUO-KIF format.
- Local or Remote SystemOnTPTP, System - Choose which SystemOnTPTP to use, and which ATP system.
- Maximum answers - Desired number of answers for the query.
- Query time limit - CPU time limit for the query.
- Output format - TPTP, IDV, or hyperlinked KIF
- Ask button - Execute the ATP system on the query.
- Tell button - Add the data to the knowledge base.

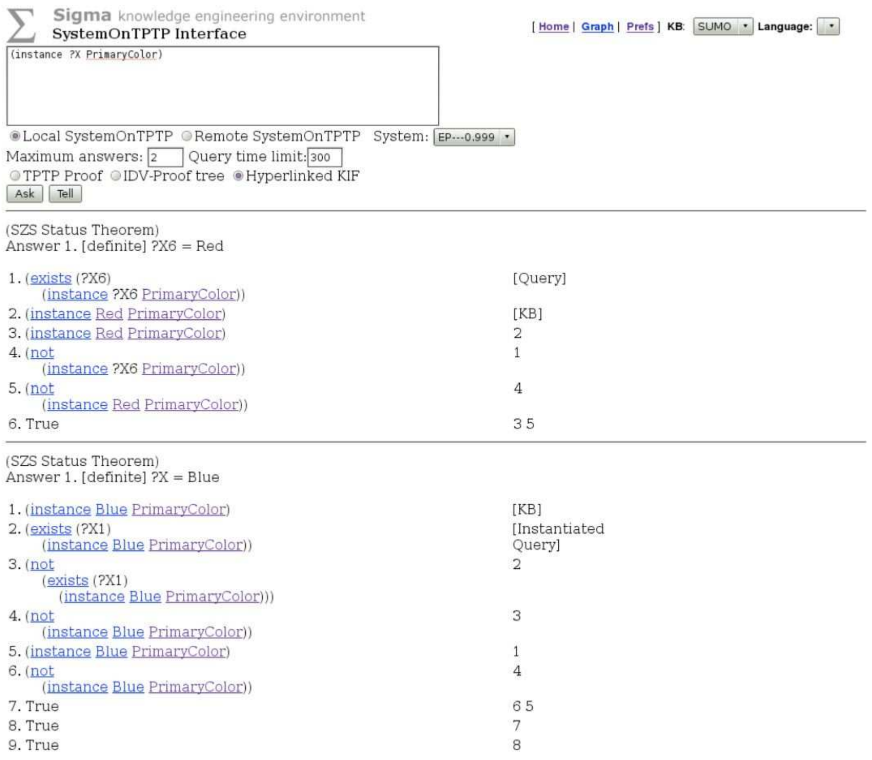
5 Sample Use

As an example, EP's proof of the following SUO-KIF format query to the SUMO knowledge base is considered: `(instance ?X PrimaryColor)`. The query asks for an instance of a primary color in the SUMO knowledge base. In SUMO the following are considered primary colors: Black, Blue, Red, White, and Yellow. The query was run using the internal implementation of SystemOnTPTP, asking for two answers, with a CPU limit of 300s, and hyperlinked KIF output. Figure 4 shows the result.

EP returns the first proof shown in the output, with SZS status Theorem. OAESys is used to extract the first answer - Red. The proof and answer are translated to the hyperlinked KIF format by SigmaKEE. MANSEX then augments the query to deny the answer Red, and EP returns another TPTP proof behind the scenes. OAESys is used to extract the second answer - Blue, which is used to instantiate the existentially quantified variable of the conjecture. EP returns the second proof shown in the output. The left column of the hyperlinked KIF is labeled SUO-KIF format formulae, with embedded HTML hyperlinks back to terms in the SUMO knowledge base. The right column describes the source of the formula: the parent formulae, the knowledge base (KB), or the query.

6 Conclusion

While KBR and ATP are both mature fields, there has not been significant cross-fertilization between the two communities. Both communities would benefit from a greater degree of interaction. The integration of the TPTPWorld into SigmaKEE brings together tools that support both communities, which should make collaboration easier, and drive further cross-disciplinary research.



Sigma knowledge engineering environment
SystemOnTPTP Interface
[Home | Graph | Prefs] KB: SUMO Language:
Ask
Tell

```
(instance ?X PrimaryColor)

[✓ Local SystemOnTPTP] Remote SystemOnTPTP   System: EP---0.999
Maximum answers: 2  Query time limit: 300
TPTP Proof  IDV-Proof tree  [✓ Hyperlinked KIF]

(SZS Status Theorem)
Answer 1. [definite] ?X6 = Red

1. (exists (?X6)                                   [Query]
     (instance ?X6 PrimaryColor))
2. (instance Red PrimaryColor)                     [KB]
3. (instance Red PrimaryColor)                     2
4. (not                                            1
     (instance ?X6 PrimaryColor))
5. (not                                            4
     (instance Red PrimaryColor))
6. True                                            3 5

(SZS Status Theorem)
Answer 1. [definite] ?X = Blue

1. (instance Blue PrimaryColor)                    [KB]
2. (exists (?X1)                                   [Instantiated
     (instance Blue PrimaryColor))                 Query]
3. (not                                            2
     (exists (?X1)
       (instance Blue PrimaryColor)))
4. (not                                            3
     (instance Blue PrimaryColor))
5. (instance Blue PrimaryColor)                    1
6. (not                                            4
     (instance Blue PrimaryColor))
7. True                                            6 5
8. True                                            7
9. True                                            8
```

Fig. 4. Sample hyperlinked KIF format proofs

Future work includes translation of SUMO and other knowledge bases to the new typed higher-order format (THF) of the TPTP language [2], and use of higher-order ATP systems such as LEO II [1] to answer higher-order queries over the knowledge bases.

References

1. C. Benzmüller and L. Paulson. Exploring Properties of Normal Multimodal Logics in Simple Type Theory with LEO-II. In C. Benzmüller, C. Brown, J. Siekmann, and R. Statman, editors, *Festschrift in Honour of Peter B. Andrews on his 70th Birthday*, page To appear. IfCoLog, 2007.
2. C. Benzmüller, F. Rabe, and G. Sutcliffe. THF0 - The Core TPTP Language for Classical Higher-Order Logic. In P. Baumgartner, A. Armando, and D. Gilles, editors, *Proceedings of the 4th International Joint Conference on Automated Reasoning*, Lecture Notes in Artificial Intelligence, 2008.
3. K. Claessen and N. Sorensson. New Techniques that Improve MACE-style Finite Model Finding. In P. Baumgartner and C. Fermueller, editors, *Proceedings of the CADE-19 Workshop: Model Computation - Principles, Algorithms, Applications*, 2003.

4. M.R. Genesereth and R.E. Fikes. Knowledge Interchange Format, Version 3.0 Reference Manual. Technical Report Logic-92-1, Computer Science Department, Stanford University, 1992.
5. J. Hurd. First-Order Proof Tactics in Higher-Order Logic Theorem Provers. In M. Archer, B. Di Vito, and C. Munoz, editors, *Proceedings of the 1st International Workshop on Design and Application of Strategies/Tactics in Higher Order Logics*, number NASA/CP-2003-212448 in NASA Technical Reports, pages 56–68, 2003.
6. I. Niles and A. Pease. Towards A Standard Upper Ontology. In C. Welty and B. Smith, editors, *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems*, pages 2–9, 2001.
7. A. Pease. The Sigma Ontology Development Environment. In F. Giunchiglia, A. Gomez-Perez, A. Pease, H. Stuckenschmidt, Y. Sure, and S. Willmott, editors, *Proceedings of the IJCAI-03 Workshop on Ontologies and Distributed Systems*, volume 71 of *CEUR Workshop Proceedings*, 2003.
8. A. Pease and G. Sutcliffe. First Order Reasoning on a Large Ontology. In J. Urban, G. Sutcliffe, and S. Schulz, editors, *Proceedings of the CADE-21 Workshop on Empirically Successful Automated Reasoning in Large Theories*, volume 257 of *CEUR Workshop Proceedings*, pages 59–69, 2007.
9. Y. Puzis, Y. Gao, and G. Sutcliffe. Automated Generation of Interesting Theorems. In G. Sutcliffe and R. Goebel, editors, *Proceedings of the 19th International FLAIRS Conference*, pages 49–54. AAAI Press, 2006.
10. A. Riazanov and A. Voronkov. The Design and Implementation of Vampire. *AI Communications*, 15(2-3):91–110, 2002.
11. S. Schulz. E: A Brainiac Theorem Prover. *AI Communications*, 15(2-3):111–126, 2002.
12. M.E. Stickel. SNARK - SRI's New Automated Reasoning Kit. http://www.ai.sri.com/ stickel/snark.html.
13. 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.
14. G. Sutcliffe, E. Denney, and B. Fischer. Practical Proof Checking for Program Certification. In G. Sutcliffe, B. Fischer, and S. Schulz, editors, *Proceedings of the Workshop on Empirically Successful Classical Automated Reasoning, 20th International Conference on Automated Deduction*, 2005.
15. G. Sutcliffe, S. Schulz, K. Claessen, and A. Van Gelder. Using the TPTP Language for Writing Derivations and Finite Interpretations. In U. Furbach and N. Shankar, editors, *Proceedings of the 3rd International Joint Conference on Automated Reasoning*, number 4130 in Lecture Notes in Artificial Intelligence, pages 67–81, 2006.
16. G. Sutcliffe and C.B. Suttner. The TPTP Problem Library: CNF Release v1.2.1. *Journal of Automated Reasoning*, 21(2):177–203, 1998.
17. G. Sutcliffe, J. Zimmer, and S. Schulz. TSTP Data-Exchange Formats for Automated Theorem Proving Tools. In W. Zhang and V. Sorge, editors, *Distributed Constraint Problem Solving and Reasoning in Multi-Agent Systems*, number 112 in Frontiers in Artificial Intelligence and Applications, pages 201–215. IOS Press, 2004.
18. S. Trac, Y. Puzis, and G. Sutcliffe. An Interactive Derivation Viewer. In S. Autexier and C. Benzmüller, editors, *Proceedings of the 7th Workshop on User Interfaces for Theorem Provers, 3rd International Joint Conference on Automated Reasoning*, volume 174 of *Electronic Notes in Theoretical Computer Science*, pages 109–123, 2006.

19. J. Urban and G. Sutcliffe. ATP Cross-verification of the Mizar MPTP Challenge Problems. In N. Dershowitz and A. Voronkov, editors, *Proceedings of the 14th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning*, number 4790 in Lecture Notes in Artificial Intelligence, pages 546–560, 2007.
20. A. Van Gelder and G. Sutcliffe. Extending the TPTP Language to Higher-Order Logic with Automated Parser Generation. In U. Furbach and N. Shankar, editors, *Proceedings of the 3rd International Joint Conference on Automated Reasoning*, number 4130 in Lecture Notes in Artificial Intelligence, pages 156–161. Springer-Verlag, 2006.
21. R. Waldinger. Whatever Happened to Deuctive Question Answering? In N. Dershowitz and A. Voronkov, editors, *Proceedings of the 14th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning*, number 4790 in Lecture Notes in Artificial Intelligence, pages 15–16, 2007.