Progress Towards Effective Automated Reasoning with World Knowledge

Geoff Sutcliffe
University of Miami, USA

Martin Suda
Charles University in Prague, Czech Republic

Alexandra Teyssandier and **Nelson Dellis**
University of Miami, USA

Gerard de Melo
Max-Planck-Institut für Informatik, Germany

Abstract

There is a growing demand for automated reasoning with world knowledge. SPASS-XDB is an automated reasoning system that incorporates world knowledge from multiple external sources, asynchronously on demand, during its reasoning process. This paper describes how SPASS-XDB's reasoning is focussed and controlled, its sources of world knowledge, and their use in solving a range of test problems.

Introduction

In recent years there has been a growing demand for automated reasoning with *world knowledge*. For example, an intelligent system would understand that cities near water areas are subject to flooding, and with world knowledge of what cities are coastal be able to conclude that those cities might be flooded (e.g., by tidal surge). World knowledge is available from a growing number of data sources that have been curated sufficiently for use in formal reasoning systems, e.g., the YAGO knowledge base (Suchanek, Kasneci, and Weikum 2007), WordNet, DBPedia, etc. World knowledge sources can be static (e.g., databases and SPARQL endpoints), dynamic (e.g., web services), or computational (e.g., computer algebra systems). Static sources provide a large but finite number of facts. Dynamic and computational sources can provide infinitely many facts by generating them on demand. Positive ground facts can be used directly by a reasoning system to answer simple "database join" queries. For deep reasoning, world knowledge must be used in conjunction with rich ontological axiomatizations that describe the structure of the world in which the world knowledge resides. Examples of such axiomatizations include the SUMO ontology (Niles and Pease 2001) and the Cyc knowledge base (Lenat 1995). Continuing the example from above, an intelligent system would reason from axioms that specify that coastal cities are near the sea, seas are bodies of water, and bodies of water are water areas, that coastal cities are near water areas, and hence might be flooded.

Automated Theorem Proving (ATP) systems have traditionally not been well suited to reasoning with world knowledge (where the facts are viewed as positive unit axioms), because they expect to load all the available axioms before a deduction starts. The large (possibly infinite) number of axioms that are available from world knowledge sources dictates that the axioms be retrieved dynamically during a deduction. SPASS-XDB (Suda and others 2009) is an ATP system that is able to incorporate world knowledge axioms from multiple external sources, asynchronously on demand, during its deduction process. This paper describes the progress that has been made towards effective automated reasoning with world knowledge in SPASS-XDB. Four aspects are described: focussed and controlled reasoning, a range of external sources of world knowledge, user friendly interfaces, and testing in a range of application domains. Readers are advised to read (Suda and others 2009) to obtain maximal enjoyment of this paper, and for details of the SPASS-XDB core that is overviewed in the next section.

The Core of SPASS-XDB

SPASS-XDB is a modified version of the well-known, state-of-the-art, first-order ATP system SPASS (Weidenbach and others 2009). The system architecture comprises SPASS-XDB system itself, mediators, and external sources of axioms, as shown in Figure 1. SPASS-XDB accepts a problem file containing (i) specifications for external sources of axioms, (ii) internal axioms, and (iii) a conjecture to be proved. All the formulae are written in TPTP format (Sutcliffe and others 2006), the de facto standard for first-order ATP system data. The external specifications provide information about the external sources of axioms: a template for the available axioms, the name of an executable *mediator* program that interfaces to the external source, and auxiliary information for controlling the use of the external source.

The basic SPASS-XDB algorithm augments SPASS' classic CNF saturation algorithm[1] with steps to request and accept axioms from the external sources. External axioms are requested when a negative literal of the "given" clause (of SPASS-XDB's saturation loop) matches the template of an external specification. A request is sent to the `stdin` of the corresponding mediator, which marshalls the TPTP format request into a query for the external source, retrieves matching facts from the external source, unmarshalls the facts into TPTP format axioms, and delivers them on its

[1] The reader is assumed to be familiar with the basic principles of CNF refutation based ATP, e.g., (Weidenbach 2001).

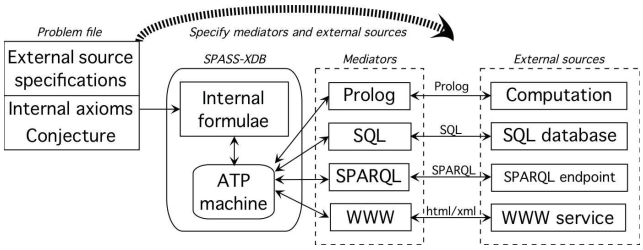

Figure 1: System Architecture

`stdout`. Requests are written as TPTP format "questions", and external axioms are delivered as TPTP format "answers", using the SZS standards (Sutcliffe 2008). One request can result in the delivery of multiple axioms. SPASS-XDB maintains an ordered request queue for each mediator, so that only one request is sent at a time. The requests are made and axioms delivered asynchronously, so that SPASS-XDB continues its deduction process while axioms are being retrieved from the (comparatively slow) external sources. When axioms are delivered from a mediator they are integrated into the deduction process by adding them to SPASS-XDB's "usable" list.

Focus and Control of SPASS-XDB's Reasoning

The soundness of SPASS-XDB follows from the soundness of SPASS. The notion of completeness is somewhat different in this setting, and for various practical reasons completeness is compromised. SPASS, like most (all?) ATP systems, was designed under the assumption that all formulae are in the problem file, i.e., ignorant that external axioms might be delivered. To regain completeness, constraints on SPASS' search had to be relaxed. This increases the search space, i.e., the constraints had to be relaxed in a controlled fashion, as is described below. The search space is also affected by the number of external axioms that can be delivered, and mechanisms to control the delivery and focus the consequent search have been implemented.

Clause Resurrection

A process of clause *resurrection* has been added to the basic SPASS algorithm, to overcome incompleteness that arises when a request for external axioms does not return any axioms (either because the external source does not have any, or the control mechanisms described below have been applied too tightly). The clause that caused the request to be queued is retrieved from the "worked off" list, placed back in the usable list, and all negative literals are marked as selected. This has two effects, the first to cause more requests for external axioms to be queued, the second to allow internal inferencing against all those literals. Resurrection is controlled by the `-EResurrect` flag described below.

Controlling the Use of External Sources

Requests for external axioms are initially controlled by using universally quantified variables in the external specification templates, to indicate positions that must be instantiated when a request is made for matching axioms. This typically reduces the number of axioms delivered (compared to having variables in those positions of the request). For example, requests for birth dates can require that the person is named, to avoid requesting the birth dates of all people.

Other aspects of using external sources are controlled by `xdb()` terms in the auxiliary information areas of the external specifications. The `xdb()` terms used so far are:

- `xdb(limit,`*limit term*`)` to limit the activities of the external source. The limit terms used so far are:
  - `number(`*maximal number of axioms to deliver*`)`
  - `cpu(`*maximal CPU time to be used obtaining axioms*`)`
- `xdb(use,unit)` to specify that the given clause must be a unit clause for its literal to generate a request. The effect is to ensure that the proof (by refutation) can be completed if any axioms are delivered. The predicate is put last in the predicate precedence ordering for literal selection from the given clause. This is used with the PrintTTY external source described later in this paper.
- `xdb(use,side_effect)` to specify that literals with the template's predicate do not contribute to the weight of a clause, and hence do not affect clause selection by weight. This is used with the PrintTTY and XDB-Translator external sources described later in this paper.
- `xdb(translate,`*Translation*`)` to specify that requests must be translated before being sent, and axioms delivered must be translated before being passed back to SPASS-XDB. Translations are necessary to align different external sources' terms that refer to the same entity, e.g., YAGOSUMO's `s__AbrahamLincoln` and DBPedia's `'Abraham Lincoln'`. The *Translation* term provides a template for the translation request. Translations are done by an external source of "translation axioms", described later in this paper.

Controlling SPASS-XDB's Search

Command line options have been implemented to provide control over SPASS-XDB's search for a proof, by modifying SPASS' original constraints on the search space.

Literal selection: SPASS' `-Select` flag controls literal selection from the given clause. The new default selects negative literals of a clause according to a predicate precedence ordering that corresponds to the order in which the predicate symbols occur in the conjecture. The effect is to take advantage of the user's natural ordering of literals in the conjecture, so that earlier literals are dealt with before later ones.

Set of Support: The `-ESOS` flag (on by default) turns on SPASS' set of support strategy, and removes the ordering constraints on the worked off clause in inferences. The set of support strategy provides a goal directed form of deduction. Removing the ordering constraints recovers completeness that is lost when ordering is used with set of support.

Request queueing: The `-EAsk` flag affects the queueing of requests for external axioms. SPASS-XDB queues requests corresponding to given clause literals that are selected or maximal. The default value for `-EAsk` allows only the first such request to be queued.

Axiom alignment: The `-EExtIntAligned` flag (disabled by default) causes the given clause literals for which requests were queued to become selected. This in turn causes only those literals to be used for internal inferences, thus aligning requests for external axioms with internal inferences. The intuitive motivation is to fairly balance inferencing with external and internal axioms.

Timing clause resurrection: The `-EResurrect` flag controls the timing of clause resurrection. By default a clause is resurrected only when no more requests can be queued, i.e., when there is no more hope of finding useful external axioms. Higher values for `-EResurrect` allow more inferences against internal axioms, thus bypassing any waiting for possible inferences against external axioms.

Use of external axioms: The `-EWith` flag controls which clauses and literals an external axiom can inference with. By default an external axiom can inference with only literals for which a request was queued. This ensures that external axioms are used for their intended purpose.

External Sources of World Knowledge Axioms

This section describes the mediators that interface with the external sources of world knowledge axioms, and provides details of the external sources currently available.

Mediators

Figure 1 shows the four mediator types that have been implemented. The code for these is generic, so that they can be used for many different external sources through parameterization or minor modification. The Prolog mediator interfaces with external sources implemented in Prolog. Appropriate operator definitions allow the mediator to read a TPTP format question directly, and call a corresponding Prolog procedure that returns the axioms. The SQL mediator interfaces with mySQL databases. A TPTP format question is transformed into a `SELECT...WHERE` query that is sent to the specified database. The query extracts matching tuples from the database, which are then converted to TPTP format axioms. The SPARQL mediator is an extension of the SQL mediator. It forms a URL based on the SPARQL end point that is provided as part of the external specification. The URL is sent to the SPARQL service, which returns the matching RDF triples in XML. The XML is then converted to TPTP format axioms. The WWW mediator is used to talk to general web services. An appropriate URL is built from the TPTP format question and sent to the web service. The XML data returned is converted to TPTP format axioms.

Figure 1 shows a configuration in which SPASS-XDB communicates directly with the mediators via their `stdin` and `stdout` streams. This configuration requires that SPASS-XDB and the mediators execute on the same computer. In order to provide more flexible use of SPASS-XDB, an HTTP proxy mediator and a mediator "Q&A" server have been implemented. These are used in the configuration shown in Figure 2. SPASS-XDB and the proxy mediator run on one computer, and talk to the Q&A server via standard HTTP protocol. The mediators run on the same computer as the Q&A server, and obtain the requested axioms as

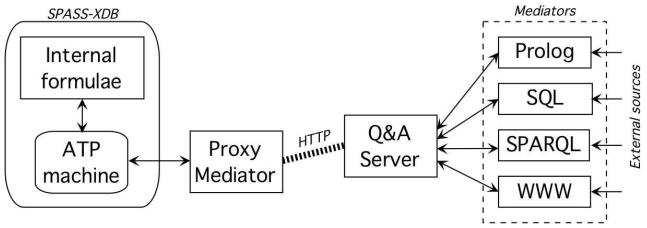

Figure 2: The HTTP Proxy Mediator

before. A web interface to the Q&A server is also available, as described later in this paper.

Static External Sources

The static external sources are databases stored in various forms that can be translated to ground unit axioms. While they might not be absolutely static (they are updated over time), the data is stable and not computed in real time. This contrasts with the dynamic sources described below.

YAGOSUMO: The YAGO knowledge base (Suchanek, Kasneci, and Weikum 2007) has provided a lot (around 14.5 million facts) of world knowledge about people, places, classifications of entities, etc. The facts have been exported from YAGO using syntax that matches the SUMO ontology axioms, then converted to a flattened form (removing function structures) and stored in a MySQL database. The data is accessed using the SQL mediator.

AmazonBooks: Amazon provides a large corpus of information about the products it sells, in particular about books.[2] This external source takes an author name, and returns information about all the books by that author (sold by Amazon). For each book, the title, binding, publication date, and price are provided in the axioms that are delivered. The data is accessed using the WWW mediator.

DBPedia: DBPedia is "a community effort to extract structured information from Wikipedia and to make this information available on the Web".[3] DBPedia provides a SPARQL endpoint, and thus the data is accessed using the SPARQL mediator. Entities in DBPedia are represented by URLs, which provide unique identification and are inappropriate for human users. Therefore the DBPedia source is used in conjunction with SPASS-XDB's translation mechanism to provide a human-friendly presentation of the data.

LinkedMDB: The Linked Movie Database (LinkedMDB) is an "open semantic web database for movies".[4] Like DBPedia, LinkedMDB provides a SPARQL endpoint for querying its data, and the SPARQL mediator is used to access the data. Also like DBPedia, the translation mechanism is used to provide a human-friendly presentation.

LatLong and Location: The LatLong and Location source provides geographical information about cities. The LatLong source obtains its data from the Yahoo Maps Web

[2] `http://www.amazon.com/`
[3] `http://dbpedia.org`
[4] `http://www.linkedmdb.org/`

Service[5], while the Location source obtains its data from GeoNames[6]. Both use the WWW mediator.

Mondial: The Mondial database is a corpus of international geographic information, and is available in a variety of formats.[7] The XML format has been retrieved, and a Prolog program has been developed to provide a source of various geographic axioms, e.g., cities in countries, common borders of countries. The Prolog mediator is used.

Dynamic External Sources

The dynamic external sources obtain real time world knowledge. Reasoning with these axioms is interesting because different things can be proved at different times.

Weather: Two sources of current weather axioms are available. The first converts a location to a latitude and longitude using the Location source, and then obtains the weather for those coordinates from the GeoNames service. The second finds cities that have a specified weather condition. Some rough matching is used so that a specified condition matches all conditions that contain that word, e.g., "cloudy" matches "partly cloudy". This weather data is obtained from Yahoo. Both sources use the WWW mediator.

XChange: XChange provides axioms containing currency conversion data. A question specifies a source currency and amount, and a target currency. The axioms delivered include the amount in the target currency. The exchange data is obtained from Time Genie[8], using the WWW mediator.

Computational External Sources

The computational external sources use computer programs to generate axioms. As such, these sources typically offer an infinite number of axioms.

ArithmeticInt and ArithmeticReal: Current ATP systems have notoriously weak support for arithmetic. The arithmetic sources are used to evaluate ground arithmetic expressions and provide corresponding axioms. The implementation is a simple Prolog program that relies on the arithmetic capabilities of Prolog. However, it would be realistic to extend this source to use a powerful computer algebra system (Bauer and Clarke 1998). The Prolog mediator is used.

Babelfish: The Babelfish source provides axioms translating phrases from one natural language to another. The service is provided by Yahoo[9], using the WWW mediator.

Service External Sources

The service external sources provide axioms that do not contain knowledge per se, but rather provide a service or side-effect to support the use of SPASS-XDB.

[5] `http://developer.yahoo.com/maps/`

[6] `http://www.geonames.org/`

[7] `http://www.dbis.informatik.uni-goettingen.de/Mondial/`

[8] `http://rss.timegenie.com/`

[9] `http://babelfish.yahoo.com`

LookDifferent: Many sources of world knowledge have a Unique Names Assumption (UNA), whereby different atoms are assumed to have different interpretations. The LookDifferent source provides axioms for pairs of terms that are syntactically unequal, according to Prolog's \== operator, thus providing a controlled implementation of a UNA. The Prolog mediator is used.

RegExp: The RegExp source provides axioms for matching strings and regular expressions (provided as atoms), e.g., `regexp('abbcd','b+cg*d$')`. RegExp reads TPTP format data directly, and thus acts as its own mediator. It is written in Perl.

PrintTTY: The PrintTTY source prints the contents of its request to the terminal, and returns the corresponding axiom. This is useful for printing, e.g., the variables of a proven conjecture that contain the answer to a question. This source is typically used with `xdb(use,unit)`, so that it ends the proof. A variant that does not return the axiom is also available, which causes SPASS-XDB to continue searching for alternative proofs. This provides a way to obtain multiple answers to a question. The Prolog mediator is used.

XDB-Translator: The XDB-Translator provides axioms that translate atoms between various sources' terminology. These axioms have a standardized format. They are intended to answer requests from the SPASS-XDB's translation mechanism, but can also be requested explicitly (like any other external axiom). The Prolog mediator is used.

User Interfaces

Three web-based user interfaces have been developed/extended to support use of SPASS-XDB and the external sources of axioms.

SystemOnTPTP is a web interface for submitting problems to ATP systems. SPASS-XDB has been added to the suite of available systems. SystemOnTPTP is available at

`www.tptp.org/cgi-bin/SystemOnTPTP`

AleXSI is a web interface for building a conjecture that SPASS-XDB is able to (in principle) prove, using the available sources of external axioms and SUMO axioms. The conjecture has outermost existentially quantified variables, and a body that is a conjunction of disjuncts. SUMO axiom files to be `included` can also be specified. The predicate and function symbols in the external and SUMO axioms are presented in an interactive interface that allows the user to search for symbols using regular expressions. The completed conjecture can be syntax checked and passed to the SystemOnTPTP interface, or submitted directly to SPASS-XDB. AleXSI is available at

`www.tptp.org/cgi-bin/AleXSI`

SystemQATPTP is a web interface that provides direct access to the external sources of axioms. The user can specify a question in the format used by SPASS-XDB, and see the axioms delivered. SystemQATPTP is available at

`www.tptp.org/cgi-bin/SystemQATPTP`

Testing

SPASS-XDB has been tested on a selection of problems crafted to illustrate the capabilities of the overall system, and the effective use of external axioms, in conjunction with SUMO and other internal axioms. The problems are:[10]

- `AbeMammal`, which proves that Abraham Lincoln was a mammal. The proof requires one external axiom that says he was a human, and eight internal SUMO axioms that define the taxonomy from humans to hominids to primates to mammals.
- `CloudyCapitals`, which finds capitals of two bordering European countries that have the same degree of cloud cover. The proof requires five external axioms about European countries, two about the capitals' weather, and one service axioms for printing the answer.
- `Composer18thCentury`, which finds a composer born in the first half of the 18th century. The proof requires four external axioms to find composers and check their birthdays, and two external service axioms for translation and printing the answer.
- `AlPacino`, which names an Al Pacino movie that contains the name "Glen" in it twice. The proof requires an external axiom to find the movie, and two external service axioms to check the regular expression `'Glen.*Glen'` and print the answer.
- `CuriePrizes`, which names the Nobel prizes won by members of the Curie family. The variant of the PrintTTY source is used to print the answers, so that SPASS-XDB finds all possible answers. Nine external axioms about the Curie's and their prizes are used.
- `ViennaTravel`, which proves it is possible to travel by land from Vienna to Budapest. The proof requires five external axioms about the countries, and six internal axioms that define properties of geographic regions and land paths.
- `CapitalLevelMoscow`, which finds an OECD country whose capital is at the same latitude as Moscow, rounded to the nearest degree. The proof requires four external axioms about OECD countries, two external axioms for real arithmetic, and four external service axioms for translation, distinguishing Moscow from the capital, and printing the answer.
- `FloodingCapital`, which extends `CapitalLevelMoscow` to require that the capital can be flooded. The proof additionally requires an external axiom for the coastal capital, and eight internal SUMO axioms about flooding, bodies of water, and coastal cities.

Table 1 shows the results. The columns are the number of requests queued for external sources, the number of requests sent, the number of external axioms delivered, the number of external axioms used in the proof, the CPU time taken by SPASS-XDB and the mediators (the times taken by the external sources cannot be measured), and the wall clock time taken. SPASS-XDB and the mediators were run on a 2.8GHz computer with 1.5GB memory, running Linux 2.6. The default control settings were used.

Problem	Queued	Sent	Del'd	Used	Der'd	CPU	WC
`AbeMammal`	24	5	110	1	8090	24s	59s
`Cloudy`	2951	204	196	8	5427	158s	257s
`Composer`	222	172	233	6	198	2s	98s
`AlPacino`	63	36	61	3	95	1s	21s
`Curies`	12	12	11	9	10	2s	18s
`Vienna`	66	58	53	5	510	12s	60s
`Capital`	66	45	64	10	69	2s	23s
`FloodingC`	148	146	92	11	7861	59s	221s

Table 1: SPASS-XDB testing results

The results show that rather complex problems can be solved using a few external axioms, supported by a few general ontological axioms.[11] In problems whose solutions require deep reasoning using ontological axioms, e.g., `FloodingCapital`, the interaction between the external axioms and the ontological axioms can cause a large search space if it is not carefully controlled. When solving `FloodingCapital`, the `-ESOS` flag causes only the conjecture to be placed in the set of support. This leads to goal directed reasoning, and provides a strong initial focus for SPASS-XDB's search. The first literal in the conjecture determines which countries are members of the OECD, and is the first selected because of the `-Select` flag. The subsequent request results in 27 axioms being delivered, and the search space branches into 27 parts that are developed concurrently. For each country the capital city is retrieved. Thanks to the `-EAsk` flag the requests for capital cities are the only ones made at that stage, avoiding other requests, e.g., for latitudes, that would compound the branching of the search space. Each city is found to look different from "Moscow", and only then are the latitudes retrieved. While the external axioms are being (comparatively slowly) retrieved, the search continues trying to prove that one of the capitals is subject to flooding. This grows the search space as each of the 27 parts interacts with the general ontological axioms, but thanks to the `-EResurrect` flag the growth is incremental. When the external axioms have been delivered the search becomes refocused, because only Copenhagen is at the same latitude as Moscow. One of the strengths of SPASS-XDB is the ability to do deep reasoning while asynchronously retrieving external axioms. The asynchronous activity is evident in Table 1 from the differences between the numbers of requests queued and the numbers sent. The larger wall clock times confirm that, as expected, the external sources are comparatively slow.

The use of general ontological axioms can alone lead to a large search space. For example, the ontological axioms used in `FloodingCapital` provide an indirect specifica-

[10] Available at `http://www.tptp.org/ATPSystems/SPASS-XDB/`, and solvable using SystemOnTPTP.

[11] In case you're wondering, the answers that SPASS-XDB found for the "question" style problems are: Tirana in Albania and Podgorica in Montenegro (both with few clouds when the testing was done), Carl Philipp Emanuel Bach (born 1714), the movie "Glengarry Glen Ross", Marie Curie – Physics and Chemistry prizes, and Pierre Curie – Physic prize, and Copenhagen in Denmark (latitude 55.68 north, Moscow is at 55.76 north).

tion of the "isa" hierarchy: Each class of entities, e.g., seas, bodies of water, etc., is a member of the meta-class of "sets and classes". The "isa" hierarchy is then specified by an axiom that says if two classes are members of the meta-class, one of them is a subclass of the other, and some object is a member of the subclass, then that object is a member of the superclass. This specification allows for reasoning about objects and classes in the same framework, but increases the search space for an ATP system. The general power of ATP systems is necessary to deal with this search. When solving `FloodingCapital`, after it has been established that Copenhagen is the city at the same latitude as Moscow, the search proceeds to use these ontological axioms to establish that Copenhagen might be flooded, as described in the Introduction. This part of the search is done last, again thanks to the `-Select` flag causing the corresponding conjecture literal to be dealt with later.

SPASS-XDB's default control settings were established through earlier experimentation, and examination of logs from successful and failed proof attempts. In order to confirm the general optimality of the default settings, SPASS-XDB was run over the test problems using all combinations of flag values. The results confirmed all settings except for `-Select`, where one of SPASS' original possible values (not the default) produced better results on the harder problems `CapitalLevelMoscow` and `FloodingCapital`. The reason for this effect is currently being investigated.

Conclusion

This paper has presented practical progress made in the development of the SPASS-XDB automated reasoning system, which is able to retrieve world knowledge from external sources, asynchronously on demand during its reasoning process. Features that focus and control the reasoning process, and a range of sources of world knowledge, allow SPASS-XDB to solve complex problems. New user interfaces have provided new ways for users to interact with SPASS-XDB and the sources of world knowledge.

The availability of large numbers of external axioms, coupled with general ontological axioms, e.g., transitivity of subclass membership, naturally produces a large search space. Ignorance of what external axioms will be delivered means that traditional ATP techniques for constraining the search space lead to incompleteness. Therefore new techniques have to be developed, and SPASS-XDB has made progress towards that goal. A further necessary step will be to integrate techniques for automatically selecting the few ontological axioms (from a large rich ontology) necessary for solving a given problem – the development of such techniques has already been the topic of some successful research, e.g., (Urban and others 2008) and Krystof Hoder's successful SInE framework (Sutcliffe 2009). An embedding of SPASS-XDB into SInE is currently underway.

There are many theoretical and practical aspects where this work might be improved. On the theoretical side, it is necessary to investigate and formalize the properties of an ATP system that starts without knowing what external axioms will be delivered while it is reasoning. Aspects related to reasoning with equality also need to be considered. On the practical side, automatic configuration of control features according to the problem characteristics – an *automode*, will improve performance. Finally, a natural language interface is being developed, based on a specialized translation from Attempto Controlled English (Fuchs, Kaljurand, and Kuhn 2008) to TPTP format first-order logic.

Acknowledgements: The original development of SPASS-XDB was done in the Automation of Logic group at the Max-Planck-Institut für Informatik, Germany. Martin Suda was supported by the Grant Agency of Charles University, grant 9828/2009.

References

Bauer, A., and Clarke, E. 1998. Analytica - An Experiment in Combining Theorem Proving and Symbolic Computation. *Journal of Automated Reasoning* 21(3):295–325.

Fuchs, N.; Kaljurand, K.; and Kuhn, T. 2008. Attempto Controlled English for Knowledge Representation. In *Proceedings of Reasoning Web: 4th International Summer School*, number 5224 in LNCS, 104–124.

Lenat, D. 1995. CYC: a Large-scale Investment in Knowledge Infrastructure. *Communications of the ACM* 38(11):33–38.

Niles, I., and Pease, A. 2001. Towards A Standard Upper Ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems*, 2–9.

Suchanek, F.; Kasneci, G.; and Weikum, G. 2007. YAGO: A Core of Semantic Knowledge. In *Proceedings of the 16th international World Wide Web Conference*, 697–706.

Suda, M., et al. 2009. External Sources of Axioms in Automated Theorem Proving. In *Proceedings of the 32nd Annual Conference on Artificial Intelligence*, number 5803 in LNAI, 281–288.

Sutcliffe, G., et al. 2006. Using the TPTP Language for Writing Derivations and Finite Interpretations. In *Proceedings of the 3rd International Joint Conference on Automated Reasoning*, number 4130 in LNAI, 67–81.

Sutcliffe, G. 2008. The SZS Ontologies for Automated Reasoning Software. In *Proceedings of the 7th International Workshop on the Implementation of Logics*, number 418 in CEUR, 38–49.

Sutcliffe, G. 2009. The 4th IJCAR Automated Theorem Proving System Competition - CASC-J4. *AI Communications* 22(1):59–72.

Urban, J., et al. 2008. MaLARea SG1: Machine Learner for Automated Reasoning with Semantic Guidance. In *Proceedings of the 4th International Joint Conference on Automated Reasoning*, number 5195 in LNAI, 441–456.

Weidenbach, C., et al. 2009. SPASS Version 3.5. In *Proceedings of the 22nd International Conference on Automated Deduction*, number 5663 in LNAI, 140–145.

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