Presenting TSTP Proofs with Inference Web Tools

Paolo Pinheiro da Silva[1], Geoff Sutcliffe[2], Cynthia Chang[3], Li Ding[3], Nick del Rio[1], and Deborah McGuinness[3]

[1]University of Texas at El Paso, [2]University of Miami, [3]Rensselaer Polytechnic Institute

Abstract. This paper describes the translation of proofs in the Thousands of Solutions from Theorem Provers (TSTP) solution library to the Proof Markup Language (PML), and the subsequent use of Inference Web (IW) tools to provide new presentations of the proofs. The translation enriches the TSTP proofs with proof provenance meta-data, and provides new possibilities for proof processing.

1 Introduction

The Thousands of Problems for Theorem Provers (TPTP)[1] problem library [12] and the Thousands of Solutions from Theorem Provers (TSTP)[2] solution library are large corpora of data for and produced by Automated Theorem Proving (ATP) systems and tools. In particular, the TSTP provides solutions to the TPTP problems, so that the main computation linking the two libraries is the execution of ATP systems on the TPTP problems to produce the TSTP solutions. Additionally, there are many other tools, mostly from the TPTPWorld [10], that are used on the corpora for other tasks such as problem analysis, problem transformation, proof analysis, proof verification, and proof presentation. The TPTP and the TSTP files are written using the TPTP language [11]. The common modus operandi of the tools is to work on one file (problem or solution) at a time, focussing on the first-order logical data therein.

The Inference Web (IW)[3] [5] is a semantic web based knowledge provenance infrastructure that supports interoperable explanations of sources, assumptions, learned information, and answers, as an enabler for trust. IW includes two components that are important for this work, the Proof Markup Language (PML) ontology [7] and the IW toolkit. PML is a semantic web based representation for exchanging explanations, including provenance information - annotating the sources of knowledge, justification information - annotating the steps for deriving the conclusions or executing workflows, and trust information - annotating trustworthiness assertions about knowledge and sources. The IW toolkit provides web-based and standalone tools that facilitate human users to browse, debug,

[1] http://www.tptp.org
[2] http://www.tptp.org/TSTP/
[3] http://inference-web.org/

explain, and abstract knowledge encoded in PML. In contrast to the TPTP, there is less focus on the logical data and the fine-grained reasoning processes - PML supports arbitrary logical data and inference steps including, e.g., extraction of data from non-logical sources, conversion to logical forms, clausification and first-order inferences, etc.

There are obvious parallels between the TPTP language/TPTPWorld and the PML language/IW toolkit. While the scope of the IW is broader than the logic-focussed TPTP/TSTP, there are obvious benefits to building bridges between the two. Principally, the TSTP offers a large corpora of data for testing and developing the IW, and the IW offers alternative views of the proofs in the TSTP. This paper describes the translation of TSTP files to PML format, and the presentation of the proofs using IW tools. The contribution of this work is to add value to TPTP proofs, by translation to PML and viewing the translated proofs with IW tools. Specific benefits include an XML proof format for TPTP proofs, links to provenance information maintained in the IW (e.g., information about ATP systems), structural search tools (rather than `grep`ing over the text form of TPTP proofs), and new views on TPTP proofs and proof nodes.

This paper is organized as follows: Section 2 provides the necessary background about the TPTP, TSTP, and PML. Section 3 describes the translation of TSTP files into PML format. Section 4 describes four IW tools' presentations of the proofs, demonstrating the value of these different views.

2 Background

2.1 About the TPTP and TSTP

The top level building blocks of TPTP and TSTP files are *annotated formulae*, *include directives*, and *comments*. An annotated formula has the form:

language`(`*name*`,` *role*`,` *formula*`,` *source*`,` *useful_info*`).`

The *language*s currently supported are `fof` - formulae in full first order form, and `cnf` - formulae in clause normal form. The *role* gives the user semantics of the *formula*, e.g., `axiom`, `definition`, `lemma`, `conjecture`, which guides the use of the formula in an ATP system. The logical *formula*, in either FOF or CNF, uses a consistent and easily understood notation [13]. The forms of identifiers for uninterpreted functions, predicates, and variables follow Prolog conventions, i.e., functions and predicates start with a lowercase letter, variables start with an uppercase letter, and all contain only alphanumeric characters and underscore. The TPTP language also supports interpreted symbols, which either start with a `$`, e.g., `$true` and `$false`, or are composed of non-alphanumeric characters, e.g., `=` and `!=` for equality and inequality. The basic logical connectives are `!`, `?`,`~`, `|`, `&`, `=>`, `<=`, `<=>`, and `<~>`, for $\forall$, $\exists$,$\neg$, $\vee$, $\wedge$, $\Rightarrow$, $\Leftarrow$, $\Leftrightarrow$, and $\oplus$ respectively. Quantified variables follow the quantifier in square brackets, with a colon to separate the quantification from the logical formula. The *source* of an annotated formula describes where the formula came from, most commonly a `file` record or an `inference` record. A `file` record stores the name of the file from which the annotated formula was read, and optionally the name of the annotated formula

as it appears in the file. An `inference` record stores three items of information about an inferred formula: the name of the inference rule provided by the ATP system; a list of useful information items, e.g., the semantic `status` of the formula as an SZS ontology value [13]; and a list of the parents. The *useful_info* is a list of arbitrary useful information, as required for user applications. An example of a FOF formula, supplied from a file, is:

```
fof(formula_27,axiom,
    ! [X,Y] :
      ( subclass(X,Y) <=>
      ! [U] : ( member(U,X) => member(U,Y) )),
    file('SET005+0.ax',subclass_defn),
    [description('Definition of subclass'), relevance(0.9)]).
```

An example of an inferred CNF formula is:

```
cnf(175,lemma,
    ( rsymProp(ib,sk_c3) | sk_c4 = sk_c3 ),
    inference(factor_simp,[status(thm)],[
        inference(para_into,[status(thm)],[96,78,theory(equality)])]),
    [iquote('para_into,96.2.1,78.1.1,factor_simp')]).
```

Each problem file in the TPTP has a header section and a list of the annotated formulae that describe the problem. The header section contains information fields that provide context for the problem, including: the name and domain of the problem, short and long English descriptions of the problem, information about the source of the problem, the status of the problem in terms of the SZS ontology, and statistics about the problem. Each file in the TSTP has a header section and a list of the annotated formulae that describe the solution. The header section contains information fields that provide context for the solution, including: the name of the ATP system that produced the derivation, the name of the TPTP problem, the command line issued to run the ATP system, information about hardware and software resources used, the date and time the solution was produced, the result and output status in terms of the SZS ontology, and statistics about the solution.

At the time of writing this paper, the TPTP contains 11279 problems in 35 domains, and the TSTP contains the results of running 43 ATP systems and system variants on all the problems in the TPTP. The solution files are classified according to the TPTP problem domains, then by TPTP problem, and finally by the ATP systems – this information is visible in the directory hierarchy and solution file name.

2.2 About PML

PML is an interlingua for representing and sharing explanations generated by various intelligent systems such as hybrid web-based question answering systems, text analytic components, theorem provers, task processors, web services, rule engines, and machine learning components. PML is split into three modules – provenance, justification, and trust relations.

– The provenance ontology provides primitives for recording properties of entities that have been used or processed. Properties such as name, description, date and time of creation, authors, and owner, can be recorded. The IW Registry provides a public repository that allows users to register meta-data about entities.
– The justification ontology provides primitives for encoding justifications for derived conclusions. Some details are provided below.
– The trust relation ontology provides primitives for explaining belief assertions associated with information and trust assertions associated with sources.

PML classes are OWL [6] classes (they are subclasses of `owl:Class`), and PML data is therefore expressible in the RDF/XML syntax. PML is used to build OWL documents representing both proofs and proof provenance information. For this work, the representation of proofs is of primary interest. The two main constructs of proofs in PML are NodeSets and InferenceSteps. A NodeSet is used to host a set of alternative justifications for one conclusion. A NodeSet contains:

– A URI that is its unique identifier.
– The conclusion of the proof step.
– The expression language in which the conclusion is written.
– Any number of InferenceSteps, each of which represents an application of an inference rule that justifies the conclusion.

An InferenceStep contains:

– The inference rule that was applied to produce the conclusion.
– The antecedent NodeSets of the inference step.
– Bindings for variables in the inference.
– Any number of discharged assumptions.
– The original sources upon which the conclusion depends.
– The inference engine that performed the inference step.
– A time stamp recording when the inference step was performed.

A proof consists of a collection of NodeSets, with a root NodeSet as the final goal, linked recursively to its antecedent NodeSets.

3 TPTP to PML Translation

The translation of a TSTP proof into PML is done by parsing the TSTP file using the TPTP-parser[4], and extracting the necessary information into PML object instances. The proof is translated into a PML NodeSet collection, with each formula in the solution being translated as singleton member of the collection (but see Section 5 for hints about future work which will aggregate proofs, so

[4] A parser for the TPTP language written in Java by Andrei Tchaltsev at ITC-irst, available from http://www.freewebs.com/andrei_ch/

that NodeSets may have multiple elements). Additionally, the conjecture of the corresponding TPTP problem is translated into a PML Query, and the `English` header field of the problem into a PML Question. The Query contains a pointer to the Question and to all NodeSet collections (from different ATP systems) that provide a solution. The Query thus provides a starting point for accessing all the proofs for that problem.

To translate a TPTP formula into a PML NodeSet, the translator needs to determine the following:

- The language of the formula, either `fof` or `cnf`. Both `fof` and `cnf` have corresponding PML provenance elements registered in the IW registry.
- The TPTP role. This is used to help determine the inference rule of the formula.
- The logical formula. The formula text is used as the NodeSet conclusion string.
- The inference engine (ATP system) that produced the proof. The translator looks in the header of the TSTP file to find the engine name. Each engine is registered in the IW registry. For example, EP has an URI of http://inference-web.org/registry/IE/EP.owl#EP.
- The inference rule used to derive the formula. Leaves of proofs that have an `axiom` role are considered to have been derived by "direct assertion". Leaves of proofs that have an `assumption` role are considered to have been derived by "assumption". For internal nodes that have an `inference` record, the translator extracts the inference rule from the record.
- The antecedent list (for inferred formulae). The members of the parent list in the `inference` record are used to form the antecedent list of the InferenceStep.
- The external source. The source is used to form the source usage of a NodeSet's inference step to describe where the conclusion originated from.
- Date and time. The translator obtains the date and time from the header of the TSTP file, to record when the proof was created.

When all information is gathered from a TSTP formula, the translator creates a NodeSet instance, and adds it to the collection forming the proof. For example, the following node from EP 0.999's proof of `PUZ001+1` ...

```
cnf(57,plain,
    ( hates(butler,X1)
    | ~ killed(X1,agatha) ),
    inference(spm,[status(thm)],[36,45,theory(equality)])).
```

... is represented by the following PML ...

```
<rdf:RDF
    xmlns:pmlp="http://inference-web.org/2.0/pml-provenance.owl#"
    xmlns:ds="http://inference-web.org/2.0/ds.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns="http://inference-web.org/2.0/pml-justification.owl#"
    xmlns:daml="http://www.daml.org/2001/03/daml+oil#">
```

```
  <NodeSet rdf:about="http://inference-web.org/proofs/tptp/Solutions/
PUZ/PUZ001+1/EP---0.999/answer.owl#ns_57">
    <pmlp:hasCreationDateTime rdf:datatype="http://www.w3.org/2001/XMLSchema#dateTime">
2008-05-01T17:11:39-04:00</pmlp:hasCreationDateTime>
    <hasConclusion>
      <pmlp:Information>
        <pmlp:hasRawString rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
hates(butler,X1) | ~ killed(X1,agatha)</pmlp:hasRawString>
        <pmlp:hasLanguage rdf:resource="http://inference-web.org/registry/LG/
TPTPCNF.owl#TPTPCNF"/>
        <pmlp:hasDescription>
          <pmlp:Information>
            <pmlp:hasRawString rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
cnf(57,plain,
    ( hates(butler,X1)
    | ~ killed(X1,agatha) ),
    inference(spm,[status(thm)],[36,45,theory(equality)])).
</pmlp:hasRawString>
            <pmlp:hasLanguage rdf:resource="http://inference-web.org/registry/LG/
TPTPCNF.owl#TPTPCNF"/>
            <pmlp:hasName rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
TPTP Formula</pmlp:hasName>
            <pmlp:hasURL rdf:datatype="http://www.w3.org/2001/XMLSchema#anyURI">
http://www.cs.miami.edu/~tptp/cgi-bin/DVTPTP2WWW/view_file.pl?Category=Solutions&amp;
Domain=PUZ&amp;File=PUZ001+1&amp;System=EP---0.999.THM-CRf.s#57</pmlp:hasURL>
          </pmlp:Information>
        </pmlp:hasDescription>
      </pmlp:Information>
    </hasConclusion>
    <isConsequentOf>
      <InferenceStep>
        <hasIndex rdf:datatype="http://www.w3.org/2001/XMLSchema#int">0</hasIndex>
        <fromAnswer rdf:resource="http://inference-web.org/proofs/tptp/Solutions/PUZ/
PUZ001+1/EP---0.999/answer.owl#answer"/>
        <hasInferenceRule rdf:resource="http://inference-web.org/registry/DPR/
EP0.999Spm.owl#EP0.999Spm"/>
        <hasAntecedentList>
          <NodeSetList>
            <ds:first rdf:resource="http://inference-web.org/proofs/tptp/Solutions/PUZ/
PUZ001+1/EP---0.999/answer.owl#ns_36"/>
            <ds:rest>
              <NodeSetList>
                <ds:first rdf:resource="http://inference-web.org/proofs/tptp/Solutions/PUZ/
PUZ001+1/EP---0.999/answer.owl#ns_45"/>
              </NodeSetList>
            </ds:rest>
          </NodeSetList>
        </hasAntecedentList>
        <hasInferenceEngine rdf:resource="http://inference-web.org/registry/IE/EP.owl#EP"/>
      </InferenceStep>
    </isConsequentOf>
  </NodeSet>
</rdf:RDF>
```

As an aside, the reverse translation from PML to TPTP is trivially possible for proofs translated from TPTP to PML, because the `hasConclusion` element of a NodeSet contains the original TPTP node as plain text. However, reconstruction of the TPTP node from the other NodeSet elements is not always completely possible because some minor items of information are not captured in the PML form. For example, the fact that an inference used the theory of equality, recorded by the `theory(equality)` parent of the TPTP node, is not captured in the PML form. NodeSets that come from sources other than translation from the TPTP are unlikely to be translatable to TPTP form, due to

different items of data being recorded, and different data formats being used. In particular, the PML form records the logical formula of a proof node as a text string in the `hasConclusion` element, and does not parse the formula into a representative structure. Thus if the logical formula is in a non-TPTP language, e.g., KIF [3] or DFG [4], there is no capability within IW to convert that to TPTP form.

4 Presentations

Given the PML encoded proofs from the TSTP, it becomes possible to use IW tools to process the proofs. Four examples are described in this section.

4.1 IW NodeSet Browser

The IW NodeSet browser allows the user to traverse the NodeSets of a proof. The presentation of a NodeSet provides:

- the conclusion, with a control to display its metadata (which contains provenance information);
- the antecedent formula and links to the NodeSets that justify (by inference) this conclusion;
- links to the leaf (evidence) nodes upon which this node depends;
- links to information about the ATP system and the inference rule used;
- the inferred formulae and links to the NodeSets that this conclusion is used to infer, with an option to show the sibling formulae used in each case;
- the formula and a link to the NodeSet finally concluded with the help of this conclusion;
- the query and question answered.

Figure 1 shows a NodeSet from EP's [8] proof for the TPTP problem `PUZ001+1`. The conclusion of the NodeSet is

```
hates(butler,X1) | ~ killed(X1,agatha)
```

The two justifying antecedents are

```
~ richer(X1,X2) | ~ killed(X1,X2)
hates(butler,X1) | richer(X1,agatha)
```

The single inferred formula is

```
hates(butler,esk1_0)
```

and the final conclusion is

```
$false
```

corresponding to the end of the proof by refutation. Figure 2 shows the provenance information obtained for the conclusion by expanding its `show metadata` control, and also the provenance information for EP 0.999 obtained by clicking on its link in the display.

Fig. 1. IW NodeSet browser presentation from EP's proof of PUZ001+1

Fig. 2. Provenance information in the IW NodeSet browser

4.2 IWBrowser

The Inference Web Browser (IWBrowser)[5] provides a graphical rendering of a PML proof, with links to the underlying provenance information stored in the PML. The presentation also provides options to focus in on the current path to the root of the proof, and to hide nodes in the proof. Figure 3 shows an extract from EP's proof for the TPTP problem `PUZ001+1`, including the example from Section 4.1. The various boxed links provide the access to provenance information and rendering options.

4.3 Probe-It!

Probe-It![6] [1] is a browser suited to graphically rendering PML based provenance associated with results derived from inference engines and workflows. Probe-It! consists of three primary views to accommodate the different kinds of provenance information: results, justifications, and provenance, which refer to final and intermediate data, descriptions of the generation process (i.e., execution traces) and information about the sources respectively. Figure 4 shows the Probe-It! rendering of SNARK's [9] proof for the TPTP problem `GEO053-2`. Each node of

[5] http://iw.stanford.edu/iwbrowser/. http://browser.inference-web.org/tptppml/ provides access to the PML translations of the TSTP files.

[6] http://trust.cs.utep.edu/probeit/

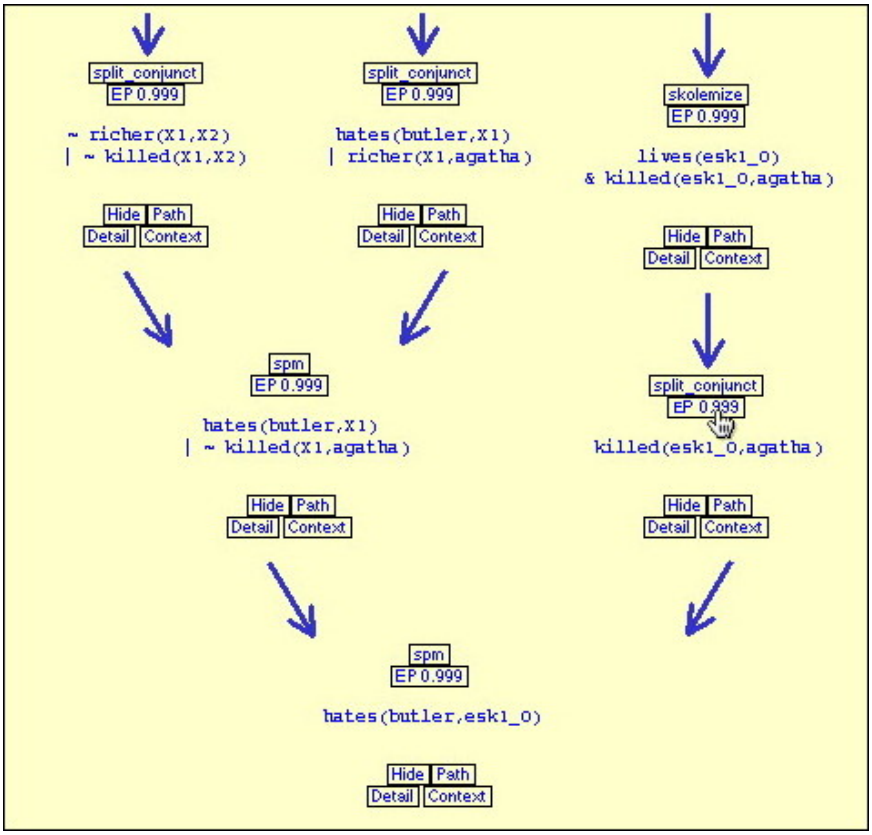

Fig. 3. IWBrowser presentation of an extract from EP's proof of `PUZ001+1`

the proof is drawn as a square, with orange squares being leaves of the proof and blue squares derived. Provenance information - the inference rule and ATP system name - is given in the upper pane of each square. The logical formula is given in the lower content pane of the square. The "panner" window in the lower left allows the user to move around the proof, while the zoom buttons provide more and less detailed views.

4.4 IWSearch

IWSearch[7] is a service in inference web architecture. It aims to discover, index, and search for PML objects available on the web. IWSearch consists of three groups of services: (i) the discovery services, which utilize Swoogle [2] search results and a focused crawler to discover URLs of PML documents on the web; (ii) the index services, which use an indexer to parse the submitted PML documents and prepare meta-data about PML objects for future search, and use a searcher

[7] http://onto.rpi.edu/iwsearch/

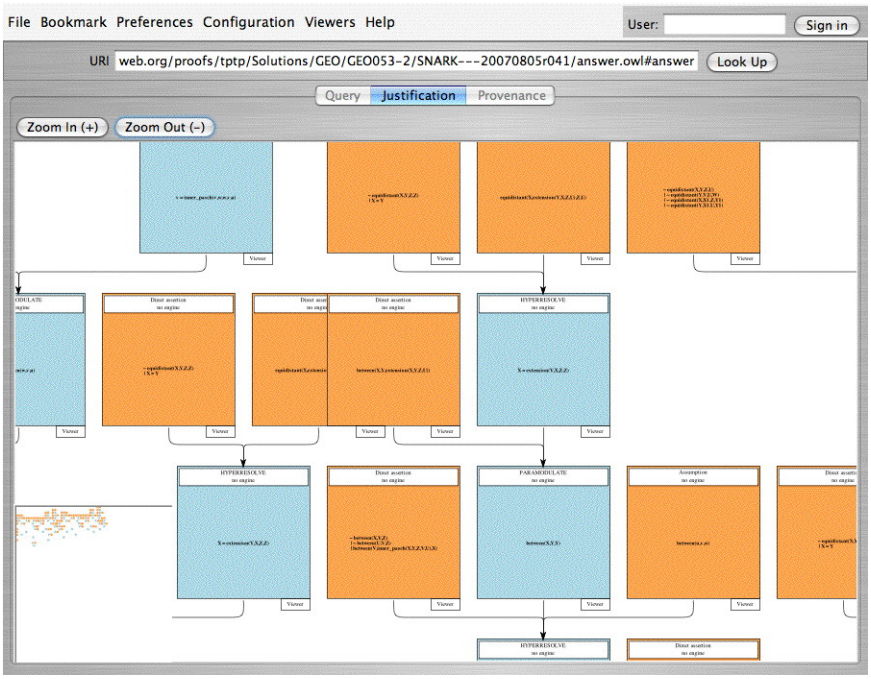

Fig. 4. Probe-It! presentation of SNARK's proof of `GEO053-2`

1 - 100 of total **456** results for **agatha** in **0.9380000233650208** seconds

label	type	more	source
agatha != butler	NodeSet	browse	http://inference-we
agatha != butler	NodeSet	browse	http://inference-we
aunt_agatha != butler	NodeSet	browse	http://inference-we
agatha != butler	NodeSet	browse	http://inference-we
! [X] : (hates(agatha,X) => hates(butler,X))	NodeSet	browse	http://inference-we
agatha != butler	NodeSet	browse	http://inference-we
! [X] : (~ richer(X,agatha) => hates(butler,X))	NodeSet	browse	http://inference-we
! [X] : (hates(agatha,X) => hates(butler,X))	NodeSet	browse	http://inference-we
! [X1] : (lives(X1) => (X1 = agatha | X1 = butler | X1 = charles))	NodeSet	browse	http://inference-we
agatha != butler	NodeSet	browse	http://inference-we
agatha = butler | hates(agatha,agatha)	NodeSet	browse	http://inference-we
agatha_hates_agatha	Document	browse	http://inference-we
! [X0] : (lives(X0) => (X0 = agatha | X0 = butler | X0 = charles))	NodeSet	browse	http://inference-we
agatha_hates_charles	Document	browse	http://inference-we
! [X0] : (~ lives(X0) | X0 = charles | X0 = butler | X0 = agatha)	NodeSet	browse	http://inference-we
butler != agatha	NodeSet	browse	http://inference-we
agatha_hates_agatha	Document	browse	http://inference-we
aunt_agatha != butler	NodeSet	browse	http://inference-we
! [X] : (hates(agatha,X) => ~ hates(charles,X))	NodeSet	browse	http://inference-we

Fig. 5. IWSearch results for the query `agatha`

to serve query calls invoked by users; (iii) the user interface services, which offer keyword search and a categorical browse interface for human or machine users. Figure 5 shows the first results returned from the query "`agatha`", after indexing the PML translations of the TSTP files. The `label` gives the raw string content of the object, the `type` is the class in the PML ontology, the `more` link provides access to that node in the IW NodeSet browser, and the `source` is the URL of the PML document containing this node.

5 Conclusion

The translation of TSTP proofs into PML, and their presentation using IW tools, changes the strict focus on logical aspects of the proof to one that encompasses proof provenance. This type of presentation is necessary for applications that demand justification or explanation of the reasoning performed. This work therefore adds value to the proofs produced by ATP systems, and makes the ATP system more suitable as tools in hybrid reasoning applications.

Work on the translation of TSTP files to PML is ongoing. Improvements and new features will be made in the near future. For example, an IW tool for combining proofs will be used to aggregate proofs from different ATP systems proofs for a given problem. This in turn will make it possible to produce new proofs with preferred characteristics, e.g., with minimal use of certain types of reasoning. The TPTP language has recently been extended to include typed higher-order logic formulae (the "THF" format), and proofs that use this language will automatically be accomodated by the translation to PML, due to the text format used for the logic formulae.

References

1. N. Del Rio and P. Pinheiro da Silva. Probe-it! Visualization Support for Provenance. In G. Bebis, R. Boyle, B. Parvin, D. Koracin, N. Paragios, T. Syeda-Mahmood, T. Ju, Z. Liu, S. Coquillart, C. Cruz-Neira, T. Muller, and T. Malzbender, editors, *Proceedings of the 2nd International Symposium on Visual Computing*, number 4842 in Lecture Notes in Computer Science, pages 732–741, 2007.
2. L. Ding, T. Finin, A. Joshi, R. Pan, R.S. Cost, Y. Peng, P. Reddivari, V. Doshi, and J. Sachs. Swoogle: A Search and Metadata Engine for the Semantic Web. In L. Gravano, C. Zhai, O. Herzog, and D. Evans, editors, *Proceedings of the 13th ACM Conference on Information and Knowledge Management*, pages 652–659. ACM Press, 2004.
3. 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.
4. R. Hähnle, M. Kerber, and C. Weidenbach. Common Syntax of the DFG-Schwerpunktprogramm Deduction. Technical Report TR 10/96, Fakultät für Informatik, Universät Karlsruhe, Karlsruhe, Germany, 1996.
5. D. McGuinness and P. Pinheiro da Silva. Explaining Answers from the Semantic Web: The Inference Web Approach. *Journal of Web Semantics*, 1(4):397–413, 2004.

6. D. McGuinness and F. van Harmelen. OWL Web Ontology Language Overview. Technical report, 2004. World Wide Web Consortium (W3C) Recommendation.
7. P. Pinheiro da Silva, D.L. McGuinness, and R. Fikes. A Proof Markup Language for Semantic Web Services. *Information Systems*, 31(4-5):381–395, 2006.
8. S. Schulz. E: A Brainiac Theorem Prover. *AI Communications*, 15(2-3):111–126, 2002.
9. M.E. Stickel. SNARK - SRI's New Automated Reasoning Kit. http://www.ai.sri.com/ stickel/snark.html.
10. G. Sutcliffe. TPTP, TSTP, CASC, etc. In V. Diekert, M. Volkov, and A. Voronkov, editors, *Proceedings of the 2nd International Computer Science Symposium in Russia*, number 4649 in Lecture Notes in Computer Science, pages 7–23. Springer-Verlag, 2007.
11. 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.
12. G. Sutcliffe and C.B. Suttner. The TPTP Problem Library: CNF Release v1.2.1. *Journal of Automated Reasoning*, 21(2):177–203, 1998.
13. 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.