Reasoning Services for an OWL Authoring Tool: An Experience Report

Thorsten Liebig & Holger Pfeifer & Friedrich von Henke
Dept. of Artificial Intelligence
University of Ulm, Germany
{liebig|pfeifer|vhenke}@informatik.uni-ulm.de

1 Background

OWL has been designed to be a formal language for representing ontologies in the Semantic Web. In short, OWL is the result of combining an expressive Description Logic (DL) with techniques and standards of the Web. DLs have been well studied in the field of knowledge representation over the last decades. As one result, some highly optimized DL reasoners have been implemented, which provide an excellent starting point for building a sound and complete OWL DL/Lite reasoner. However, having a traditional DL system with standard functionality is not enough in the current context. So far, DL systems have been used by KR experts mainly in isolated application domains. Now, in order to make the Semantic Web happen far more flexible and interactive DL-based tools are needed for building, maintaining, linking, and applying ontologies even for non-experienced users. The importance of so-called non-standard inference services that support building and maintaining knowledge bases has been pointed out recently [1, 2]. We argue that the availability of those inference services is a fundamental premise for upcoming real-world Semantic Web systems and applications. Our experience in the course of developing the graphical ontology editor OntoTrack is a prime example here.

2 OntoTrack: a Novel Ontology Editor

OntoTrack [7] is a new browsing and editing "in-one-view" ontology authoring tool for OWL Lite that combines a sophisticated graphical layout with mouse-enabled editing features and instant reasoning feedback using an external DL reasoner. More precisely, all user changes after each editing step are send to the RACER [6] reasoner via a TCP-based client interface. The reasoner will then make all modeling consequences explicitly available. OntoTrack will hand over relevant consequences (e.g. new subsumption relationships, equivalent or unsatisfiable classes) to the user by providing appropriate graphical feedback. However, implementing this feedback functionality turned out to become difficult and even impossible for some language statements (e.g. the deletion of global domain and range restrictions of properties couldn't be implemented due to a missing retraction functionality).

Currently, DL reasoners only provide some kind of batch-oriented enter and query interface. Because of lack of algorithms for appropriately handling incremental additions to a knowledge base [9] complete reclassification after each user interaction is necessary. Furthermore, in order to become aware of a new subsumption relationship due to a just added property restriction for example, OntoTrack needs to query the reasoner about direct super classes for almost all classes of the ontology in turn. One could of course narrow this set to those classes that also have an explicit or inherited restriction on that particular property or a sub-property thereof. But this requires to have explicit knowledge about inherited restrictions or sub-properties, which in turn may result in additional queries. Deletion of, or changes within, classes and properties or even fractions thereof is an analogous problem. However, using an optimized tableaux-style reasoner for a language with an expressivity comparable to that of OWL, retraction and changing of definitions (e. g. GCIs) may be of high complexity because of optimization techniques like absorption.

3 Desirable Reasoning Services

Based on our experiences in the course of developing OntoTrack, we briefly summarize our application requirements with respect to DL reasoners for supporting ontology editing.

Instead of querying for all possible changes with respect to a specific consequence (most notably the direct subsumption relationship) after each editing step we would like to have an event-triggered notification model on the reasoner side. This mechanism should only publish the set of differences in conclusions with respect to the previous state. This would correspond to a likewise TBox technique of RACERs ABox publish-subscribe mechanism.

Another desirable feature is incremental reasoning and retraction of definitions. As long as partial class definitions are concerned, additive incremental reasoning can be done with help of additional GCIs and reclassification. However, adding a global domain or range restriction will then result in GCIs which are not absorbable. Incremental reasoning with complete class definitions requires to retract the original definition before adding a new restriction in combination with the original one. As mentioned before, retraction of definitions or statements may be of high cost but is a prerequisite for interactive ontology tools. A solution could consist of a reasoner heuristic that analyzes the retraction statement and decides about on-the-fly deletion or reclassification. A related problem is how to detect and deal with statements explicitly or implicitly affected by a retraction process e. g. due to references.

A serious issue of each ontology authoring tool is concerned with debugging of ontologies. Here standard inference services provide no help to resolve inconsistencies in logical incoherent ontologies. In [12] a new reasoning service for pinpointing logical contradictions of $\mathcal{ALC}$ ontologies has been developed.

Methods for explaining unsatisfiability of classes and class subsumption have also been developed for $\mathcal{ALC}$ [11, 5]. Sophisticated debugging or explanation services in combination with an appropriate graphical user interface would obviously make ontology authoring much more efficient. An on-demand generation of an ABox model for a selected class may also be helpful for explanation.

Other novel inference services intended to support building an ontology have been developed (see sec. 6.3 in [3] for a summary). One interesting service consists of matching of class patterns against class descriptions in order to find already defined classes with a similar structure. Another approach tries to create class definitions by generalizing one or more user given ABox assertions. Other non-standard inference services like least common subsumer or most specific concept are also relevant during authoring of ontologies.

Unfortunately, only some of these non-standard reasoning services have been implemented[1] and only a few are found in state of the art reasoning systems today. First this is due to the fact that some of them only make sense if used for DLs less expressive than OWL. Second, approaches for solving these services are usually based on structural subsumption algorithms known to be not appropriate for languages like OWL. However, only some ontologies use all available language constructs. A large fraction is within restricted clusters of less expressive sublanguages [13]. We therefore hope to see some of these reasoning features (even for sublanguages of OWL) integrated into reasoners in the future.

Technical requirements with respect to interfaces and communication are also an important issue for building a successful application. A state of the art architecture should support multiple clients via standard protocols. Notably, some of the most recent reasoner developments are either not network-aware or without any interface documentation. Support of standard formats (e. g. KRSS [10], DIG [4]) and native OWL import either from file or via HTTP is also a desired quality.

4 Conclusion

The development of sophisticated and adequate Semantic Web tools for end users strongly depends on sufficiently broad reasoning services and appropriate interfaces of its core technology, namely DL reasoners. It is worth mentioning that our experiences are not specific to our choice for RACER as external DL reasoner. An internal evaluation of some DL based reasoning systems potentially capable of handling portions of OWL (FaCT, FaCT++, RACER, Pellet, BOR) identified general deficits with respect to our requirements mentioned above. We therefore argue that not only correctness, efficiency, and language conformity are important, service diversity and interactive capabilites should become an issue for research and criteria of future reasoner evaluations.

References

[1] Franz Baader, Ian Horrocks, and Ulrike Sattler. Description Logics as Ontology Languages for the Semantic Web. In *Festschrift in honor of Jörg Siekmann.* Springer, 2003.

[1] The Classic system is a notable exception here [8].

[2] Franz Baader, Ian Horrocks, and Ulrike Sattler. *Handbook on Ontologies*, chapter Description Logics, pages 3–28. International Handbooks on Information Systems. Springer Verlag, 2004.

[3] Franz Baader, Ralf Küsters, and Frank Wolter. *The Description Logic Handbook*, chapter Extensions to Description Logics. Cambridge University Press, 2003.

[4] Sean Bechhofer, Ralf Möller, and Peter Crowther. The DIG Description Logic Interface. In *Proc. of International Workshop on Description Logics (DL2003)*, Rome, Italy, 2003.

[5] Alex Borgida, Enrico Franconi, and Ian Horrocks. Explaining $\mathcal{ALC}$ subsumption. In *Proc. of International Workshop on Description Logics (DL1999)*, Linköping, Sweden, 1999.

[6] Volker Haarslev and Ralf Möller. RACER System Description. In *Proc. of the International Joint Conference on Automated Reasoning (IJCAR'2001)*, volume 2083 of *LNAI*, pages 701–706, Siena, Italy, 2001.

[7] Thorsten Liebig and Olaf Noppens. OntoTrack: Fast Browsing and Easy Editing of Large Ontologies. In *Proc. of The 2nd Int. Workshop on Evaluation of Ontology-based Tools (EON2003)*, Sanibel Island, USA, 2003.

[8] Deborah L. McGuinness and Alex Borgida. Explaining Subsumption in Description Logics. Technical Report LCSR-TR-228, Dept. of Computer Sciences, Rutgers University, September 1994.

[9] Ralf Möller and Volker Haarslev. *The Description Logic Handbook*, chapter Description Logics Systems. Cambridge University Press, 2003.

[10] Peter F. Patel-Schneider and Bill Swartout. Description Logic Specification from the KRSS Effort. Working version (draft), 1993.

[11] Stefan Schlobach and Ronald Cornet. Explanation of Terminological Reasoning A Preliminary Report. In *Proc. of International Workshop on Description Logics (DL2003)*, Rome, Italy, 2003.

[12] Stefan Schlobach and Ronald Cornet. Non-Standard Reasoning Services for the Debugging of Description Logic Terminologies. In *Proc. of the Belgian-Dutch Conference on AI (BNAI03)*, Nijmegen, The Netherlands, 2003.

[13] Christoph Tempich and Raphael Volz. Towards a benchmark for Semantic Web reasoners - an analysis of the DAML ontology library. In *Proc. of The 2nd Int. Workshop on Evaluation of Ontology-based Tools (EON2003)*, Sanibel Island, USA, 2003.