TrOWL: Tractable OWL 2 Reasoning Infrastructure

Edward Thomas, Jeff Z. Pan, Yuan Ren

Department of Computing Science, University of Aberdeen, Aberdeen AB24 3UE, UK

Abstract. The Semantic Web movement has led to the publication of thousands of ontologies online. These ontologies present and mediate information and knowledge on the Semantic Web. Tools exist to reason over these ontologies and to answer queries over them, but there are no large scale infrastructures for storing, reasoning, and querying ontologies on a scale that would be useful for a large enterprise or research institution. We present the TrOWL infrastructure for transforming, reasoning, and querying OWL2 ontologies which uses novel techniques such as Quality Guaranteed Approximations and Forgetting to achieve this goal.

1 Introduction

Ontologies play a key role in the Semantic Web [3], where the W3C recommendation OWL [9] and its successor OWL2 [6] have become the de facto standards for publishing and sharing ontologies online. Increasingly these ontologies are being used by a variety of organisations, covering the definitions of a very wide range of subjects. While the number and variety of ontologies increases, the question of how to use these ontologies at an organisational level remains unresolved.

The reason why this is not a trivial problem is that OWL-DL language has a worst-case computational complexity of NExpTime, and 2NExpTime for OWL2-DL. This means that increasingly large ontologies may, in the worst case, require exponentially increasing computing resources to reason. Because of this, OWL2 also includes a number of tractable profiles which have combined complexity of PTIME-complete or better; however, these profiles all greatly restrict the expressive power of the language. As tool support for these profiles is still limited, it is also very easy for an ontology developer to accidentally exceed the complexity of their target profile by using a construct which is beyond the capability of that language fragment.

The approach of TrOWL is to offer support for all the expressive power of OWL2-DL, while maintaining tractability, by using language transformations. In particular, we utilise a Semantic Approximation [7] to transform OWL2-DL ontologies into OWL2-QL for conjunctive query answering, and a syntactic approximation from OWL2 to OWL2-EL for TBox reasoning. In addition, TrOWL contains a profile checker to detect which profile an ontology may already fit into, and it has support for heavyweight reasoning using a plug-in reasoner such as Fact++, Pellet, Hermit, or Racer.

2 Applications

The TrOWL reasoner was developed to support work on the MOST project[1] as well as provide reasoning support for large ontology-based knowledge bases.

Validating Process Refinements During software development, processes are modelled in the standard language, Business Process Modelling Notation (BPMN); these process models are then refined to produce progressively more detailed models. Several metrics exist to validate these refinements as being consistent with the earlier models, but no tools are available which can validate these refinements automatically across multiple refinements on large models. Our approach has been to translate this process model into an ontology and use ontology reasoning services to validate the model. With this approach, we can validate that the refinements are valid, or highlight the processes which are causing the problem.

The process refinement case study generates ontologies with general concept inclusions (GCIs) of particular patterns. At the time of developing REL, mainstream reasoners such as Pellet and FaCT++ failed to efficiently provide complete classification results on the generated ontologies. Via the syntactic approximation of TrOWL, the GCIs in these ontologies can be efficiently resolved and the reasoning results can be proved complete.

Software Engineering Guidance Ontology The physical device configuration case study uses ontologies to validate the consistency of the configuration of a network device. These devices are configured with several cards, and this configuration must be validated against a model which describes correct configurations.

The case study generates ontologies describing the configuration of network devices. These ontologies can sometimes be inconsistent, reflecting an invalid configuration of a physical device. To understand how this is manifested in the physical device and provide guidance on how it may be resolved, it is necessary to find justifications for the inconsistency, and isolate each axiom set which may be causing the inconsistency. Traditional tableaux reasoners usually terminate when an inconsistency is detected, making it difficult obtain all justifications. In this case, TrOWL can provide a more efficient and reliable service when used as a reasoning backend.

Linked Open Data We have also investigated using TrOWL for linked open data repositories. We used the RDF-DL reasoning component in the Billion Triple Challenge in ISWC 2009. We managed to successfully load and reason over the billion triple RDF data set, with full RDFS reasoning over class and property subsumption. The benefit of using TrOWL for linked open data is that it supports reasoning in all profiles of OWL, as well as using RDF-DL reasoning over RDFS data. Since conjunctive query answering is always reduced to OWL-QL query answering, this allows queries to be run over large heterogeneous ontologies with its characteristic AC_0 data complexity.

[1] http://www.most-project.eu

3 Technology

TrOWL is based around two primary technologies. Language transformations, and lightweight reasoners. The most important of these are outlined briefly here.

3.1 Language Transformations

TrOWL is the common interface to a number of reasoners. Quill provides reasoning services over RDF-DL and OWL-QL; REL provides reasoning over OWL-EL; and TrOWL can support full DL reasoning using a plug-in reasoner such as Pellet or Fact++. These reasoners and the languages which they support are optimised for certain applications, for example, OWL-QL has excellent ABox query answering performance but it lacks many constructors present in the more expressive flavours of OWL2.

The transformation from OWL2 to OWL-QL is based around Semantic Approximation from OWL-DL to DL-Lite which is described in [7]. Semantic Approximation uses a heavyweight reasoner to guarantee that every axiom in the approximated ontology is valid with respect to the source ontology. Because the semantics of OWL-QL are a subset of, and are hence compatible with, the direct semantics OWL2, this means that for all reasoning results against the approximated ontology are sound. In fact, it has been shown in that for conjunctive query answering, which is the strength of the QL language, results against the semantic approximation are also complete for a very large class of queries (those with no non-distinguished variables, or with non-distinguished variables in leaf nodes of the query).

The transformation from OWL2 to OWL-EL is based on the soundness preserving approximate reasoning approach presented in [8]. This is achieved by representing non-OWL-EL concept expressions with fresh named concepts, and maintaining non-OWL EL information, such as complementary relations, in separate data structures. In the reasoning stage, additional completion rules are plugged into the inference engine to restore the semantics of these information. The approximation is syntactic-based and can be performed in linear time. The additional completion rules retain the tractability of OWL2-EL. Thus the overall complexity for OWL2-DL ontologies can be reduced to PTime. Although known to be incomplete, our evaluation shows that, REL can classify existing benchmarks very efficiently with high recall (over 95%) [8].

Other transformation techniques used in TrOWL include forgetting [5, 11, 10].

3.2 Lightweight Reasoners

Quill The Quill reasoner has been implemented in Java using a novel and unique database schema for storing normalised representations of OWL2-QL ontologies. This allows us to rewrite any conjunctive query into a single, simple, SQL query over the underlying database, using the database itself to perform the transitive completion of class and property subsumption with an innovative exploitation

of the way database indices work. To support this we have developed new algorithms to replace those proposed in [4]. for query rewriting, and ontology normalisation. Initial testing across large knowledge bases with deep concept hierarchies, such as the DBPedia dataset and the Yago ontology, shows a significant performance improvement over other DL-Lite query engines. Using the standard query rewriting algorithm PerfectRef over a deep class or property hierarchy can result in a set of hundreds or thousands of conjunctive queries, where our method will only ever result in a single query. Quill supports all reasoning tasks for OWL2-QL, including consistency and satisfiability checking, and query answering, and by using an OWL-DL reasoner it can perform semantic approximation of more expressive ontologies.

REL The REL reasoner is a java implementation of an OWL-EL reasoner, in which an optimisation of the EL+ algorithm [2] has been extended with the completion rules for OWL-EL [1]. This allows REL to provide tractable TBox reasoning for OWL-EL ontologies and make up the core component of the soundness-preserving syntactic approximation. By this way, REL can provide soundness-guaranteed tractable TBox reasoning services for OWL2-DL ontologies. In additional, REL also consists of an OWL-EL conjunctive query engine [12], which allows queries over OWL-EL ontologies been answered more efficiently without semantic approximation.

4 Demonstration

Our demonstration of the TrOWL reasoner will focus on two scenarios. The first part of the demo will show how TrOWL differs from traditional reasoners, and give a comparison of the performance of TrOWL for different reasoning tasks. The second part of the demonstration will showcase the case studies from the MOST project and show how TrOWL is helping to solve these.

The proposed structure of the demonstration is:

- Reasoning Demonstration
  - TBox Reasoning
  - Query Answering
  - Comparison with traditional reasoners
- MOST Project Case Studies
  - Validating process refinement
  - Physical device configuration
  - Requirements modelling using ontologies
- Deploying TrOWL
  - As an embedded reasoner
  - As a SPARQL endpoint
  - As a web service

5 Acknowledgements

This research has been partially supported by the European Commission and by the Swiss Federal Office for Education and Science within the 7th Framework Programme project MOST number 216691 (cf. http://most-project.eu). We would also like to thank Stuart Taylor, Nophadol Jekjantuk and Yuting Zhao at the University of Aberdeen for their helpful discussion and other contributions.

References

1. F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ Envelope. In *Proc. of the 19th Int. Joint Conf. on Articial Intelligence (IJCAI 2005)*, 2005.
2. Franz Baader, Carsten Lutz, and Boontawee Suntisrivaraporn. Is tractable reasoning in extensions of the description logic el useful in practice? In *Proceedings of the 2005 International Workshop on Methods for Modalities (M4M-05)*, 2005.
3. T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, 2001.
4. Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The dl-lite family. *Journal of Automated Reasoning*, 39(3):385–429, 2007.
5. Thomas Eiter, Giovambattista Ianni, Roman Schindlauer, and Hans Tompits. Forgetting in managing rules and ontologies. In *In Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI 2006), Hongkong*, pages 411–419. IEEE Computer Society, 2006.
6. Boris Motik, Peter F. Patel-Schneider, and Bijan Parsia. OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax. http://www.w3.org/TR/2009/REC-owl2-syntax-20091027/, Oct 2009.
7. J. Z. Pan and E. Thomas. Approximating OWL-DL Ontologies. In *the Proc. of the 22nd National Conference on Artificial Intelligence (AAAI-07)*, pages 1434–1439, 2007.
8. Yuan Ren, Gerd Gröner, Jens Lemcke, Tirdad Rahmani, Andreas Friesen, Yuting Zhao, Jeff Z. Pan, and Steffen Staab. Validating process refinement with ontologies. In *Proceedings of the 22nd International Workshop on Description Logics (DL2009)*, 2009.
9. Michael K. Smith, Chris Welty, and Deborah L. McGuiness. http://www.w3.org/TR/owl-guide/. http://www.w3.org/TR/owl-guide/, Feb 2004.
10. Kewen Wang, Zhe Wang, Rodney W. Topor, Jeff Z. Pan, and Grigoris Antoniou. Concept and role forgetting in ALC ontologies. In *International Semantic Web Conference*, pages 666–681, 2009.
11. Zhe Wang, Kewen Wang, Rodney Topor, and Jeff Z. Pan. Forgetting in DL-Lite. In *the Proc. of the 5th European Semantic Web Conference 2008 (ESWC2008)*, 2008.
12. Yuting Zhao, Jeff Z. Pan, and Yuan Ren. Implementing and evaluating a rule-based approach to querying regular el+ ontologies. In *In Proc. of the International Conference on Hybrid Intelligent Systems (HIS2009)*, 2009.