RuQAR : Reasoning Framework for OWL 2 RL Ontologies

Jaroslaw Bak, Maciej Nowak, and Czeslaw Jedrzejek

Institute of Control and Information Engineering,
Poznan University of Technology,
M. Sklodowskiej-Curie Sqr. 5, 60-965 Poznan, Poland
`{jaroslaw.bak,czeslaw.jedrzejek}@put.poznan.pl`

Abstract. This paper addresses the first release of the Rule-based Query Answering and Reasoning framework (RuQAR). The tool provides the ABox reasoning and query answering with OWL 2 RL ontologies executed by forward chaining rule reasoners. We describe current implementation and an experimental evaluation of RuQAR by performing reasoning on the number of benchmark ontologies. Additionally, we compare obtained results with inferences provided by HermiT and Pellet. The evaluation shows that we can perform the ABox reasoning with considerably better performance than DL-based reasoners.

1 Introduction and Motivation

Ontologies in information systems are becoming more and more popular in various fields, such as web technologies, database integration, multi agent systems, natural language processing, etc. However, in order to utilize all features that an ontology provides we need to apply a reasoning engine. Moreover, we can use different engines with ontologies expressed in different OWL 2 Profiles[1] (as well as in different fragments of OWL 1.1[2], eg. Horn-$\mathcal{SHIQ}$). One of the most interesting profile is OWL 2 RL which enables the implementation of polynomial time reasoning algorithms in a standard rule engine. Nonetheless, a naive implementation of OWL 2 RL reasoner is known to perform poorly with large ABoxes [2]. Moreover, since the official list[3] of OWL 2 reasoners supporting OWL 2 RL is limited, we are motivated to provide such a tool. We intent to apply OWL 2 RL reasoning in a rule-based system in an efficient way. Description logic-based reasoners handle the TBox entailments better than the ABox ones. However, the ABox reasoning can be performed more efficiently by a rule engine [3]). Nevertheless, we do not limit ourselves to one particular engine or implementation. Instead, we aim at providing easy-to-use framework for performing the ABox reasoning with OWL 2 RL ontologies in any forward chaining rule engine which will be applicable in many rule-based applications. Thus, we have

[1] `http://www.w3.org/TR/owl2-profiles/`
[2] `http://www.w3.org/Submission/owl11-overview/`
[3] `http://www.w3.org/2001/sw/wiki/OWL/Implementations`

developed the Abstract Syntax of Rules and Facts (ASRF) which can be easily applied in different rule engines. Moreover, since the interoperability with the widely-used Semantic Web Rule Language (SWRL) is desired in many practical applications, we included it in the ASRF syntax. Moreover, SWRL Built-ins are also supported.

In this paper, we describe the RuQAR (Rule-based Query Answering and Reasoning) framework. The main goal of the tool is to support query answering and reasoning with semantic data stored in a relational database. However, current implementation enables ontology-based reasoning using a standard forward chaining rule engine. We present the reasoning features that are already applicable in any application. Those features will be used in future development of RuQAR. Current implementation supports the OWL 2 RL Profile and DL-safe SWRL rules which are crucial in many Semantic Web applications.

2 The RuQAR Framework

Application of a rule engine to an ontology-based reasoning requires a transformation method of an ontology into a set of rules. According to this RuQAR implements a method of transforming an OWL 2 ontology into a set of rules and a set of facts expressed in ASRF. The transformation schema is presented in Figure 1. Firstly, an OWL 2 ontology is loaded into the HermiT[4] engine. Then, the TBox reasoning is executed. Finally, the inferred ontology is transformed into two sets: one of rules and one of facts. Both are expressed in the ASRF notation which enables easy translation into the language of a rule engine.

In ontology-to-ASRF transformation we translate each logical ontology axiom into its equivalent rule. For example, if the *prp-symp* axiom of OWL 2 RL defines a symmetric property P, then the axiom can be expressed as the following rule: $triple(?x, P, ?y) \rightarrow triple(?y, P, ?x)$, where $?x$ and $?y$ are variables. Such a rule operates on the ABox part only. As a result the transformation materializes the semantics of a given ontology in the set of Datalog-like rules (we consider it as a non-naive translation).

The reasoning process is divided into two sub-processes: for the TBox reasoning and for the ABox reasoning. The TBox reasoning is performed by HermiT during the transformation, while the ABox reasoning is performed by a rule engine. By loading a translated and inferred ontology, produced by HermiT, into the rule engine we can produce more entailments during the ABox reasoning than those supported by OWL 2 RL. However, it depends on an applied ontology (whether or not it uses constructs that are beyond the OWL 2 RL Profile).

Using ASRF sets produced by RuQAR one can apply it in any forward chaining rule engine by implementing mappings between ASRF and the language of the engine. Nevertheless, RuQAR implements translation into two rule engines: Jess[5] and Drools[6]. Both translations can be performed automatically. Moreover,

[4] `http://www.hermit-reasoner.com/`
[5] `http://jessrules.com/`
[6] `http://www.jboss.org/drools/`

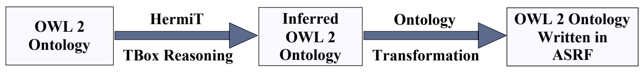

Fig. 1. OWL 2 ontology transformation schema.

ASRF is similar to syntaxes of well-known rule engines like Jess or Clips. More information about RuQAR and ASRF can be found at RuQAR's web page[7].

3 Evaluation

We evaluated RuQAR using test ontologies taken from the KAON2 website[8]: Vicodi - an ontology about European history, Semintec - an ontology about financial domain and LUBM - an ontology benchmark about organizational structures of universities. We used different datasets of each ontology (Semintec_0, Semintec_1, etc.) where the higher number means bigger ABox set. Our tests were performed on a Windows 7 desktop machine with Java 1.7 update 25 while the maximum heap space was set to 1GB.

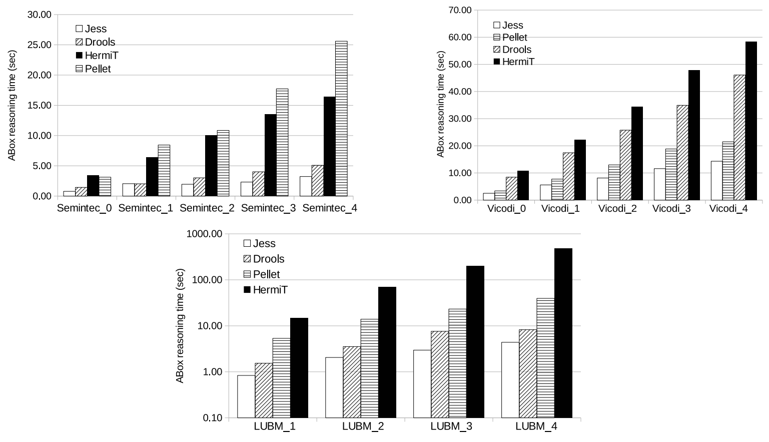

Fig. 2. The ABox reasoning times of the tested ontologies.

Evaluation schema for each ontology was the following. Firstly, we performed the TBox reasoning using HermiT. Then, an inferred ontology was loaded into each tested engine and the ABox reasoning was executed. In each case we recorded the reasoning time and counted the size of the resulting ABox. We performed the ABox reasoning with the following engines: Jess, Drools, HermiT and Pellet[9]. We verified that the reasoners produced identical results (a similar

[7] `http://etacar.put.poznan.pl/jaroslaw.bak/RuQAR.php`
[8] `http://kaon2.semanticweb.org/`
[9] `http://clarkparsia.com/pellet/`

empirical approach is applied in [1] and [4] in order to compare their OWL 2 RL reasoners with Pellet/RacerPro and HermiT, respectively). However, HermiT and Pellet provided more reasoning results in the LUBM case. It is correct, since only Vicodi is within the OWL 2 RL Profile. Nevertheless, all results inferred by Jess and Drools were among the results produced by HermiT/Pellet. Generally, we obtained better performance in the ABox reasoning with Jess/Drools than with HermiT or Pellet (see Figure 2). However, in the Vicodi case Pellet was on the second place. The reason for that is that Vicodi contains large number of classes - which means that the reasoning produces many new triples. Drools performs slower than Jess in creating new triples (or checking if a triple exists in the working memory) since it uses pure Java classes (while Jess uses its own classes). Nevertheless, the obtained results confirm that RuQAR increases the performance of the ABox reasoning in comparison to the DL-based reasoners.

4 Conclusions and Future Work

In this paper we presented RuQAR which is the first release of a reasoning framework for OWL 2 RL ontologies. This tool enables ontology transformation into rules and facts expressed in the ASRF syntax. The translation from ASRF into Jess and Drools is also provided. Moreover, we described the reasoning schema, preliminary implementation as well as performed experiments. To the best of our knowledge presented work is the first not-naive implementation of the OWL 2 RL reasoning in Drools and Jess which can be applied in any application requiring efficient ABox reasoning (except the work presented in [4], where these engines are used to infer with naive implementation of rules from the OWL 2 RL Profile specification). In the next release of RuQAR we will support relational database interface as well as optimized query processing.

Acknowledgements. The work presented in this paper was supported by UMO-2011/03/N/ST6/01602 grant.

References

1. RokanUddin Faruqui and Wendy MacCaull. Owlontdb: A scalable reasoning system for owl 2 rl ontologies with large aboxes. In Jens Weber and Isabelle Perseil, editors, *Foundations of Health Information Engineering and Systems*, volume 7789 of *Lecture Notes in Computer Science*, pages 105–123. Springer Berlin Heidelberg, 2013.
2. Aidan Hogan and Stefan Decker. On the ostensibly silent w in owl 2 rl. In Axel Polleres and Terrance Swift, editors, *Web Reasoning and Rule Systems*, volume 5837 of *Lecture Notes in Computer Science*, pages 118–134. Springer Berlin Heidelberg, 2009.
3. Georgios Meditskos and Nick Bassiliades. Combining a dl reasoner and a rule engine for improving entailment-based owl reasoning. In *Proceedings of the 7th International Conference on The Semantic Web*, ISWC '08, pages 277–292, Berlin, Heidelberg, 2008. Springer-Verlag.
4. Martin J. O'Connor and Amar Das. A pair of owl 2 rl reasoners. In Pavel Klinov and Matthew Horridge, editors, *OWLED*, volume 849 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.