Racer: An OWL Reasoning Agent for the Semantic Web

Volker Haarslev[†] and Ralf Möller[‡]
[†]Concordia University, Montreal, Canada (haarslev@cs.concordia.ca)
[‡]University of Applied Sciences, Wedel, Germany (rmoeller@fh-wedel.de)

Abstract

Racer, which can be considered as a core reasoning agent for the semantic web, is briefly described. Racer currently supports a wide range of inference services about ontologies specified in the Ontology Web Language (OWL). These services are made available to other agents via network based APIs. Racer is currently used by various clients such as ontology editors, ontology development and visualization tools, and a first web-based prototype for exploration and analysis of OWL ontologies.

1 Introduction

The Semantic Web initiative defines important challenges for knowledge representation and inference systems. Recently, several standards for representation languages have been proposed. One of the standards for the Semantic Web is the Resource Description Framework (RDF [1]). Since RDF is based on XML it shares its document-oriented view of grouping sets of declarations or statements. With RDF's triple-oriented style of data modeling, it provides means for expressing graph-structured data over multiple documents (whereas XML can only express graph structures within a specific document). As a design decision, RDF can talk about everything. Hence, in principle, statements in documents can also be referred to as resources. In particular, conceptual domain models can be represented as RDF resources. Conceptual domain models are referred to as "vocabularies" in RDF. Specific languages are provided for defining vocabularies (or ontologies). An extension of RDF for defining ontologies is RDF Schema (RDFS [2]) which only can express conceptual modeling notions such as generalization between concepts (aka classes) and roles (aka properties). For properties, domain and range restrictions can be specified. Thus, the expressiveness of RDFS is very limited. A much more expressive representation language is OWL (Ontology Web Language) [3]. Although still in a very weak way, based on XML-Schema, OWL also provides for means of dealing with data types known from programming languages.

The representation languages mentioned above are defined with a model-theoretic semantics. In particular, for the language OWL, a semantics was defined such that very large fragments of the language can be directly expressed using so-called description logics (see [4]). The fragment is called OWL DL. With some restrictions that are discussed below one can state that the logical basis of OWL can be characterized with the description logic $\mathcal{SHIQ}(\mathcal{D}_n)^-$ [5]. This means, with some restrictions, OWL documents can be automatically translated to $\mathcal{SHIQ}(\mathcal{D}_n)^-$ T-boxes. The RDF-Part of OWL documents can be translated to $\mathcal{SHIQ}(\mathcal{D}_n)^-$ A-boxes.

2 Racer: An OWL Reasoner

The logic $\mathcal{SHIQ}(\mathcal{D}_n)^-$ is interesting for practical applications because highly optimized inference systems are available (e.g., Racer [6]). Racer is freely available for research purposes and can be accessed by standard HTTP or TCP protocols (the Racer program is subsequently also called Racer server). Racer can read OWL knowledge bases either from local files or from remote Web servers (i.e., a Racer server is also a HTTP client). In turn, other client programs that need inference services can communicate with a Racer server via TCP-based protocols. OilEd [7] can be seen as a specific client that uses the DIG protocol [8] for communicating with a Racer server, whereas RICE [9] is another client that uses a TCP protocol providing extensive query facilities (see below).

The DIG protocol [8] is a XML- and HTTP-based standard for connecting client programs to description logic inference engines. DIG allows for the allocation of knowledge bases and enables clients to pose standard description logic queries. As a standard and a least common denominator it cannot encompass all possible forms of system-specific statements and queries. Let alone long term query processing instructions (e.g., exploitation of query subsumption, computation of indexes for certain kinds of queries etc., see [10]). Therefore, Racer also provides a TCP-based interface in order to receive instructions (statements) and queries. For interactive use, the language supported by Racer is not XML- or RDF-based. The advantage is that users can spontaneously type queries which can be directly sent to a Racer server. However, the Racer TCP interface can be very easily accessed from Java or C++ application programs as well. For both languages corresponding APIs are available.

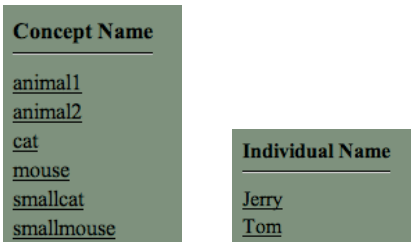

Figure 1: The lists of known concepts and individuals.

3 Some Supported Inference Services

In description logic terminology, a tuple consisting of a T-box and an A-box is referred to as a knowledge base. An individual is a specific named object. OWL also allows for individuals in concepts (and T-box axioms). For example, expressing the fact that all humans stem from a single human called ADAM requires to refer to an individual in a concept (and a T-box). Only part of the expressivity of individuals mentioned in concepts can be captured with A-boxes. However, a straightforward approximation exists (see [11]) such that in practice suitable $\mathcal{SHIQ}(\mathcal{D}_n)^-$ ontologies can be generated from an OWL document. Racer can directly read OWL documents and represent them as description logic knowledge bases (aka ontologies). In the following a selection of supported T-box queries is briefly introduced.

- Concept consistency: Is the set of objects described by a concept empty?
- Concept subsumption: Is there a subset relationship between the set of objects described by two concepts?
- Find all inconsistent concepts mentioned in a T-box. Inconsistent concepts might be the result of modeling errors.
- Determine the parents and children of a concept: The parents of a concept are the most specific concept names mentioned in a T-box which subsume the concept. The children of a concept are the most general concept names mentioned in a T-box that the concept subsumes.

Whenever a concept is needed as an argument for a query, not only predefined names are possible. If also an A-box is given, among others, the following types of A-box queries are possible:

- Check the consistency of an A-box: Are the restrictions given in an A-box w.r.t. a T-box too strong, i.e., do they contradict each other? Other queries are only possible w.r.t. consistent A-boxes.
- Instance testing: Is the object for which an individual stands a member of the set of objects described by a certain query concept? The individual is then called an instance of the query concept.
- Instance retrieval: Find all individuals from an A-box such that the objects they stand for can be proven to be a member of a set of objects described by a certain query concept.
- Computation of the direct types of an individual: Find the most specific concept names from a T-box of which a given individual is an instance.
- Computation of the fillers of a role with reference to an individual.

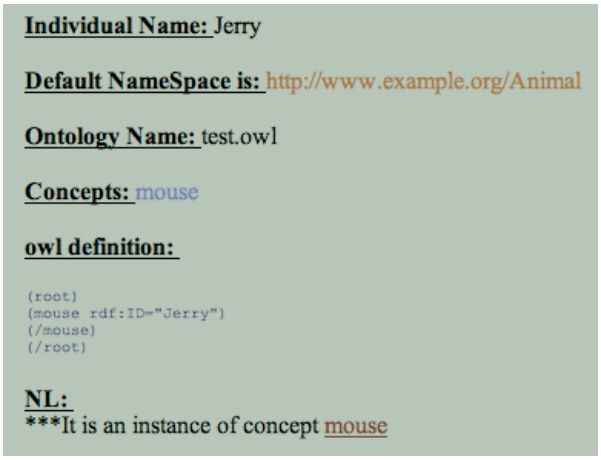

Figure 2: Information about the individual JERRY.

Given the background of description logics, many application papers demonstrate how these inference services can be used to solve actual problems with OWL knowledge bases. The query interface is extensively used by RICE and a tool for ontology exploration and analysis that is introduced in the next section.

4 Ontology Exploration and Analysis Tool

This section presents a first prototype for an ontology exploration and analysis tool designed for OWL. This tool parses OWL files and presents a "natural language" interface for exploring and analyzing ontologies. This is facilitated by using the inference services of Racer. We demonstrate this with a simple browsing scenario using a small ontology (for sake of brevity) about the cartoon characters Tom and Jerry.

Let us assume a corresponding ontology has been loaded. We start browsing the lists of all concept and individual names declared in this ontology (see Figure 1). We are interested in the individual JERRY (shown in Figure 2) and learn that JERRY is an instance of the class MOUSE. We know that cats usually eat mice, so we decide to inspect the description of CAT (see Figure 3) by clicking on the corresponding hyperlink in Figure 1.

Figure 3 shows a description of the class CAT. This description displays results from the inference services of Racer and consists of the following information.

Figure 3: Description of class CAT.

- Concept (class) name
- Default name space
- Ontology filename
- The names of the ancestor classes (TOP, which is a synonym for THING, and ANIMAL1, ANIMAL2)
- The names of the descendent classes (BOTTOM, which is a synonym for NOTHING, and SMALLCAT)
- The parents (ANIMAL1 and ANIMAL2)
- The children (SMALLCAT)
- The names of the roles (properties) mentioned in this class definition (EAT-MOUSE)
- The individual names that are instances of this class (TOM)
- The OWL definition (for debugging purposes)
- A description of the class declaration in a formalized "natural language"

We learn that a cat has to be in the relationship (role) EAT-MOUSE with at least one individual. After clicking on the corresponding hyperlink, the description EAT-MOUSE is displayed in Figure 4.

Roles may be part of a role hierarchy. For instance, for this example we discover that EAT-MOUSE is defined as a child of role EAT-ANIMAL and a parent of role EAT-SMALL-MOUSE.

Most readers will agree that this kind of information is better readable and helpful in understanding ontologies than just reading the OWL specification. The final version of this web-based tool will offer more support for the exploration and analysis of (unknown) OWL ontologies with the help of the OWL reasoning agent Racer. It is planned to provide a more ad-

Figure 4: Description of property EAT-MOUSE.

vanced query support and better cross-referencing. The tool is implemented as a web server and can be used with any web browser. Currently the tool is designed as a reactive agent for understanding OWL ontologies. However, we also envision a more proactive version of this tool that would automatically notify users or agents if interesting information about ontologies becomes available. In the following section we describe a general interface supporting the proactive behavior of such a type of agents.

5 Accessing Retrieval Inference Services

The main examples for the Semantic Web use information retrieval applications involving one or more agents. In a full-fledged information retrieval scenario, an agent might consult a document management system provided by an agent host environment. The agent can ask for documents that match a certain query in a similar way as discussed above. This scenario can also be realized with Racer if documents are annotated with meta data formalized with RDF [12]. Information about documents can be represented using A-boxes. RDF annotations for documents are read by Racer and corresponding assertions are added to an A-box. Data types and values play an important role for describing documents (e.g., year, ISBN number etc.). Agents can retrieve documents by posing retrieval queries to A-boxes w.r.t. to specific T-boxes in the way exemplified above.

5.1 Publish/Subscribe Interface

If we consider an instance retrieval query Q w.r.t. an A-box A, then it is clear that the solution set for Q could be extended if more information is added to A over time (whoever is responsible for that, another agent or the agent host environment). It would be a waste of resources to frequently poll the host environment with the same query (and repeated migration operations). Therefore, Racer supports the registration of queries at some server w.r.t. to some A-box (Publish/Subscribe Interface). With the registration, the agent specifies an IP address and a port number. The corresponding Racer Server passes a message to the agent if the solution set of a previously registered instance retrieval query is extended. The message specifies the new individuals found to be instances of the query concept Q. We call the registration of a query, a subscription to a channel on which Racer informs applications about new query results. For details see the Racer manual [11].

Rather than considering a single query in isolation, a prac-

tical system should be able to consider query sets (as database systems do in many applications). With the publish/subscribe interface, multiple queries can be optimized by Racer. Instance retrieval queries can be answered in a faster way if the set of candidates can be reduced. In a similar way as for databases, the idea is to exploit results computed for previous instance retrieval queries by considering query subsumption (which is decidable in the case of the query language that Racer supports). However, this requires computing index structures for the T-box (the process is known as T-box classification) and, therefore, query subsumption is enabled on demand only. On the one hand, there are some applications, in which A-boxes are generated on the fly with few queries referring to a single A-box. On the other hand, there are applications which pose many queries to more or less "static" T-boxes and A-boxes (which are maybe part of the agent host environment). The Racer Server supports both application scenarios. As a design decision, Racer computes answers for queries with as few resources as possible. Nevertheless, a Racer Server can be instructed to compute index structures in advance if appropriate to support multiple queries.

5.2 Additional Features of the Racer System

Optimizations: Various optimization techniques for ontology-based query answering with respect to T-boxes, A-boxes, and concrete values have been developed, implemented, and investigated with the Racer System. One of the design goals of Racer is to automatically select state of the art optimization techniques that are applicable to the current input.

Persistency: In a similar way as in database systems, for query answering w.r.t. T-boxes and A-boxes complex data structures are computed and used internally by Racer. Internal structures of T-boxes and A-boxes being processed for query answering can be saved to disk for quick access and later reuse if the Racer Server is restarted.

Multi-User Support, Thread Safeness, Locking, Load Balancing: In a distributed systems context, there can be multiple agents connecting to a server at the same time. If they refer to the same A-boxes and T-boxes, requests must be synchronized. Thus similar problems as with databases such as thread safeness, locking, and load balancing have to be dealt with. For instance, if multiple Racer Servers are started, queries can be automatically directed to "free" Racer Servers. These problems are tackled by the Racer Proxy, which is supplied as part of the Racer System distribution.

6 Conclusion

This paper briefly described the OWL reasoning agent Racer and its services and demonstrated that Racer can cooperate with an ontology exploration and analysis tool. Description logic systems freely available for research purposes now provide for industry-oriented software integration and begin to ensure stable access in multi-user environments as can be expected in the context of the semantic web with its XML-based representation languages (RDF, OWL).

Acknowledgements

We gratefully acknowledge the work of Ying Lu, who is developing the ontology exploration and analysis tool.

References

[1] O. Lassila and R.R. Swick, "Resource description framework (RDF) model and syntax specification. recommendation, W3C, february 1999. http://www.w3.org/tr/1999/rec-rdf-syntax-19990222", 1999.

[2] D. Brickley and R.V. Guha, "RDF vocabulary description language 1.0: RDF Schema, http://www.w3.org/tr/2002/wd-rdf-schema-20020430/", 2002.

[3] F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein, "OWL web ontology language reference", 2003.

[4] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, *The Description Logic Handbook*, Cambridge University Press, 2003.

[5] F. Baader, I. Horrocks, and U. Sattler, "Description logics as ontology languages for the semantic web", in *Festschrift in honor of Jörg Siekmann*, D. Hutter and W. Stephan, Eds. 2003, LNAI. Springer-Verlag.

[6] V. Haarslev and R. Möller, "Racer system description", in *International Joint Conference on Automated Reasoning, IJCAR'2001, June 18-23, 2001, Siena, Italy.*, 2001.

[7] S. Bechhofer, I. Horrocks, and C. Goble, "OilEd: a reason-able ontology editor for the semantic web", in *Proceedings of KI2001, Joint German/Austrian conference on Artificial Intelligence, September 19-21, Vienna*. LNAI Vol. 2174, 2001, Springer-Verlag.

[8] S. Bechhofer, R. Möller, and P. Crowther, "The DIG description interface", in *Proc. International Workshop on Description Logics – DL'03*, 2003.

[9] R. Möller, R. Cornet, and V. Haarslev, "Graphical interfaces for Racer: querying DAML+OIL and RDF documents", in *Proc. International Workshop on Description Logics – DL'03*, 2003.

[10] V. Haarslev and R. Möller, "Optimization stategies for instance retrieval", in *Proc. International Workshop on Description Logics – DL'02*, 2002.

[11] V. Haarslev and R. Möller, "The Racer user's guide and reference manual", 2003.

[12] Adobe Systems Inc., "Embedding XMP metadata in application files", 2002.