Semantic Web 1 (2011) 1–5
IOS Press

The RacerPro Knowledge Representation and Reasoning System [1]

Editor(s): Pascal Hitzler, Kno.e.sis Center, Wright State University, Dayton, Ohio, USA
Solicited review(s): Ulrike Sattler, University of Manchester, UK; Sergio Tessaris, Free University of Bozen-Bolzano, Italy; Zhisheng Huang, Vrije Universiteit Amsterdam, The Netherlands

Volker Haarslev [a], Kay Hidde [b], Ralf Möller [c,*] and Michael Wessel [d]
[a] *Concordia University, 1455 de Maisonneuve Blvd. W., Montreal, Quebec H3G 1M8, Canada*
E-mail: haarslev@cse.concordia.ca
[b] *Racer Systems GmbH & Co. KG, Blumenau 50, 22089 Hamburg, Germany*
E-mail: hidde@racer-systems.com
[c] *School of Electrical Engineering, Computer Science, and Mathematics, Hamburg University of Technology, Schwarzenbergstr. 95, 21073 Hamburg, Germany*
E-mail: r.f.moeller@tu-harburg.de
[d] *Racer Systems GmbH & Co. KG, Blumenau 50, 22089 Hamburg, Germany*
E-mail: wessel@racer-systems.com

Abstract. RacerPro is a software system for building applications based on ontologies. The backbone of RacerPro is a description logic reasoner. It provides inference services for terminological knowledge as well as for representations of knowledge about individuals. Based on new optimization techniques and techniques that have been developed in the research field of description logics throughout the years, a mature architecture for typical-case reasoning tasks is provided. The system has been used in hundreds of research projects and industrial contexts throughout the last twelve years. W3C standards as well as detailed feedback reports from numerous users have influenced the design of the system architecture in general, and have also shaped the RacerPro knowledge representation and interface languages. With its query and rule languages, RacerPro goes well beyond standard inference services provided by other OWL reasoners.

Keywords: Ontology Reasoning Systems, Description Logic Reasoning Systems, Deduction over Tboxes and Aboxes, Expressive Ontology-based Query Answering, Abox Abduction

1. Introduction

For all software systems, and in particular for a knowledge representation and reasoning engine, it holds that the system architecture is influenced by typical application areas for which it should be most effective. This is true also for the RacerPro system, a practical software system for building knowledge-based systems for demanding application scenarios ranging from autonomous agents on the semantic web to knowledge-based software engineering. We describe the main features of the system, in combination with a motivation for the design principles behind RacerPro.

On the one hand, the goal of the paper is to describe the features of a state-of-the-art description logic inference system (with support for syntax standards such as OWL [19]). On the other hand, the description contains a set of literature references such that interested researchers can find a comprehensive bibliography on terminological as well as assertional reasoning tech-

[1]The development of RacerPro was partially supported by DFG (Deutsche Forschungsgemeinschaft) and the European Commission under ICT frameworks FP6 and FP7.
[*]Corresponding author. E-mail:r.f.moeller@tu-harburg.de

nology. We hope to be able to stimulate the development of new, even better optimized reasoning architectures, such that even more powerful knowledge-based applications can be built in the future. The article is structured as follows. We first give an overview on the design principles of RacerPro, the description and query languages as well as on the overall system architecture. Afterwards, the article describes the interfaces, and it shortly refers to relevant use cases and application scenarios. In the last section, we conclude and present an outlook on future developments. We assume that the reader is familiar with description logics and logic programming. The presentation in this article refers to RacerPro 2.0. RacerPro is freely available for individuals participating at a degree-granting organization such as a universities or schools. More powerful network-supporting server versions can be licensed (e.g., for commercial purposes).

2. System Overview and Scientific Impact

2.1. Design Principles

RacerPro is available as a server version (RacerPro Server) or as a software library with API (RacerMaster for Common Lisp). In this system description we refer to RacerPro Server, and we will just use RacerPro as a name for the system. RacerPro communicates with client programs via various interfaces, either RacerPro-specific ones (maximum expressivity) or standardized ones (maximum portability). A powerful graphical interface is provided for manual interaction with the server, and for submitting ad-hoc server extensions and queries. See Figure 1 for an overview on the system architecture. It should be noted that RacerPro can be extended using a simple plugin mechanism. For the users' convenience, parts of the RacerPro code are open source, and can be used to extend the RacerPro reasoning server (see below for details).

In an ontology-based application, usually multiple representation languages are used for different purposes. The backbone is a description logic language for defining the terminological part (e.g. in OWL 2 syntax [29]), which is often extended with other logical languages for the assertional part, such as, for instance, logic programming rules, the region connection calculus for aspects of spatial reasoning, or Allen's interval algebra for aspects of temporal reasoning, just to name a few [79]. The RacerPro system is particularly tailored for supporting this kind of applications which mainly build on the exploitation of assertional reasoning (Abox reasoning). The main idea is that Aboxes are not static parts of the ontology, but are efficiently generated on the fly (referring to a shared Tbox which is "processed" only once). Tboxes (ontologies) and Aboxes are maintained using the RacerPro server system, which communicates with remote application programs using well-defined axiom manipulation languages or entailment query languages. In addition, a rule language (based on SWRL syntax) is used to conveniently extend Abox assertions stored on the reasoning server, i.e., rules that are transferred to the server can be used to extend the expressivity w.r.t. assertional reasoning and/or make implicit information explicit on the server. Server-side Aboxes can be remotely cloned and easily extended such that variants of assertional knowledge can be conveniently managed as lightweight objects while the Tbox part they refer to is shared.[1] Besides this approach for "lightweight Aboxes", the RacerPro architecture also supports large Aboxes stored in a triple store database (AllegroGraph, see below).

One of the main design principles of RacerPro is to automatically select applicable optimizations based on an analysis of the language of the input knowledge bases and the queries being processed.

2.2. Description languages

Ontologies are based on fragments of first-order logic for describing a shared conceptualization of a domain using concept and role descriptions (called classes and properties in OWL, respectively). The initial conceptual representation language of RacerPro was $\mathcal{ALCNH}_{R^+}(\mathcal{D})^-$ [31], and RacerPro was the first system which efficiently supported concrete domains for Tbox and Abox reasoning [40,41]. RacerPro was then extended to also support inverse roles and qualitative number restrictions [44] as part of the description logic (DL) $\mathcal{SHIQ}$ [50], a practically relevant subset of OWL. Since RacerPro supports concrete domains effectively, it was found that nominals (concepts representing single domain objects as defined in the DL $\mathcal{SHOIQ}$ [49]) were not of utmost importance for RacerPro users [43]. In many cases, in which users initially wanted nominals, strings were found to be sufficient. Furthermore, since multiple Aboxes should refer

[1] The Tbox to which an Abox refers can also be changed, but obviously, this requires complete reprocessing of the assertions in the Abox.

to a single Tbox (preprocessed and indexed offline before Abox query answering), Abox assertions should not introduce implicit subsumption relationships, a design principle that is, in general, broken if nominals are supported (the standard approximation of nominals [14] is provided though by RacerPro).

While interesting optimization techniques for nominals have been developed [66], reasoning with nominals is known as hard not only from a theoretical point of view [49] but also from a practical point of view (i.e., hard also for typical-case input). RacerPro can be extended with nominals, however, once optimization techniques for reasoning with nominals get mature enough such that RacerPro applications can effectively exploit this feature (see, e.g., [61,22,23] for first results).

Role axioms ($\mathcal{SROIQ}$ [48]) are another language construct that could be integrated into RacerPro such that the full expressivity of the latest W3C standard for ontology languages (OWL 2) is not only syntactically supported but also w.r.t. intensional reasoning.

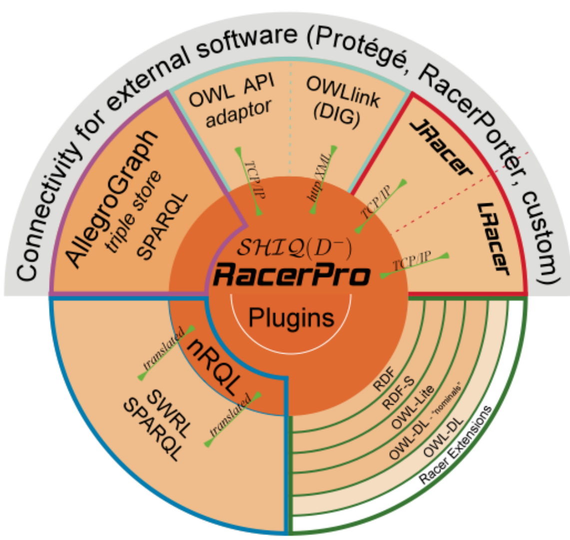

Fig. 1. RacerPro System Architecture

There are various usage scenarios of DL systems for which RacerPro is optimized. In one scenario, Tboxes are usually rather large, and Aboxes are small (< 100 individuals), but many (variants of) Aboxes need to be handled. In the other scenario, Tboxes are rather small, and one large Abox (> 100.000 individuals) is referred to in queries.

2.3. *Query Languages*

Inference services for concept subsumption and the taxonomy of a Tbox have been part of description logic reasoning systems right from the beginning in the eighties (Tbox classification). Classification is supported in RacerPro with specific optimization techniques [32,33,36,87,86] that are based on or are integrated with results obtained in other projects [24,25] as well as techniques implemented in mature predecessor DL systems such as $\mathcal{KRIS}$ [8,6,7] and FaCT [46,47,71]. Still, Tbox classification is a very fruitful research area, and new techniques are being integrated into RacerPro. Interestingly, for dealing with an $\mathcal{ELH}$ [5] version of the Snomed/CT knowledge base, a very old structural subsumption technique being integrated into RacerPro provided for classification in the range of minutes for this very large Tbox [39], with the additional advantage that (small) parts of the Tbox can indeed use more expressive language fragments.

Inference services for Aboxes are influenced by many earlier DL systems, for instance, CLASSIC [13, 14]. In these systems, the query language for finding individuals is based on concept descriptions, and, thus, rather limited (see [11,12]). In addition, in order to effective answer queries, in CLASSIC the most-specific concept names of which individuals are instances are computed in advance (Abox realization). In contrast, RacerPro was designed in such a way that the user can decide whether to compute index structures in advance or on the fly [37]. Research on RacerPro has focused on concept-based instance retrieval [34] as well as on a more expressive form of queries, namely grounded conjunctive queries [42,43].

The new Racer Query Language (nRQL, pronounced "niracle" and to be heard as "miracle") was one of the first expressive query languages for DL systems providing conjunctive queries [18] with variables ranging over named domain objects, negation as failure, a projection operator, as well as group-by and aggregation operators (the latter two features were added recently). With negation as failure and projection, one can also represent universal quantification in queries.

The formal semantics of nRQL was described in [82,83]. Interestingly, much later, a query language semantics based on a different viewpoint was described in [15]. The nRQL language can nowadays be seen as an implementation of the approach proposed in [15]. It should be emphasized that RacerPro supports query answering with the pull mode (clients send queries and retrieve result sets incrementally from the server) or with the push mode (clients subscribe conjunctive queries and receive notifications about elements in the result set incrementally) [35].

Optimization techniques for instance retrieval are analyzed in [43,58,38]. A little later, also the influential Pellet reasoning system provided support for conjunctive queries [67,68], and this emphasizes the enormous practical relevance that the RacerPro Abox reasoning work had at that time. The KAON 2 system [60] and the DL-Lite system [16] provided highly regarded transformation techniques for conjunctive queries w.r.t. Tboxes. A variant of the transformation approach for a sublanguage of OWL has also been investigated with RacerPro (see [58] for details).

Based on the query language, RacerPro was extended with a rule language such that application programs can transfer rules to the server in order to test whether certain conditions hold in order to then establish new assertions [30]. Based on practical use cases (see below), the rule language is designed to conveniently manipulate Abox assertions. It is not designed as a declarative knowledge representation language.

Also based on the query language, an abductive reasoning component was integrated into RacerPro [17]. In the abductive mode, instantiated atoms of a complex conjunctive query which cannot be proven to hold are returned as part of the query answer. The space of abducibles can be defined using named queries. Due to the use of disjunction in the formulation of these queries, multiple explanations are possible, and a ranking measure is built into the system [21]. Abduction for Abox queries is a unique feature of RacerPro.

3. Interfaces

RacerPro can be used as a server application in a network-wide context. In addition to raw TCP communication interfaces with APIs for Java (JRacer), Common Lisp (LRacer), and also C, RacerPro supports the OWLlink communication interface [55,63] (a successor of DIG [10]). RacerPro also supports various OWL 2 syntaxes, namely RDF/XML as well as OWL Functional syntax. See again Figure 1 for an overview of the RacerPro system architecture. A standardized RDF query language for RacerPro is SPARQL [65]. For a an important subset of SPARQL, RacerPro offers ontology-based reasoning. RacerPro also supports many extensions to SPARQL in a KRSS-like syntax (e.g., the full nRQL query language, publish/subscribe interface etc.). RacerPro implements the OWL API [45] such that a plugin for Protégé 4 is available [62] (Figure 2.3).

Ontologies can be read from files, or can be retrieved from the web as well as from an RDF triple store managed by the built-in AllegroGraph system (version 3) from Franz Inc. [1]. AllegroGraph can be used to store materialized inferences and also provides for a powerful query language based on SPARQL syntax [1]. AllegroGraph can store very large Aboxes for which users need nRQL query answering.

The RacerPro inference server can be programmed in a functional language called miniLisp. For instance, query results can be postprocessed by the miniLisp interpreter running small functional programs being sent to the server such that query results can be sent in application-specific XML formats to client programs via the built-in RacerPro web server/client. The language miniLisp is designed in such a way that termination of miniLisp programs is guaranteed, and miniLisp can be used to specify rather complex queries and server extensions while the reasoning server is running (see Figure 5). The functional language miniLisp can be extended by application programs (e.g., in the same sense that application programs can generate Javascript programs on the fly and send them to a web browser). In combination with a miniLisp program for manipulating query results on the server, nRQL queries ensure that it is not necessary to transfer large sets of Abox assertions from the inference server to application programs.

For more complex extensions, RacerPro supports a plugin interface. Plugins can be developed with Allegro Common Lisp Free Edition (from Franz Inc.). Thus, compiled programs can encode arbitrary algorithms to be executed on the RacerPro server. Plugins have been used, for instance, for developing nonstandard inference algorithms [72].

Using the open source library OntoLisp[2], large parts of the RacerPro code for syntactically processing ontologies are publicly available. So, for instance, one could extend RacerPro with an open source reasoner such as, e.g., CEL [9] if specific tasks require $\mathcal{EL}^{++}$ reasoning [5], while still exploiting, say, the RacerPro interface services and miniLisp as server-side algorithmic language. Ontolisp can be used for developing new reasoners as well (various Common Lisp systems are supported).

[2] See http://sourceforge.net/projects/ontolisp/.

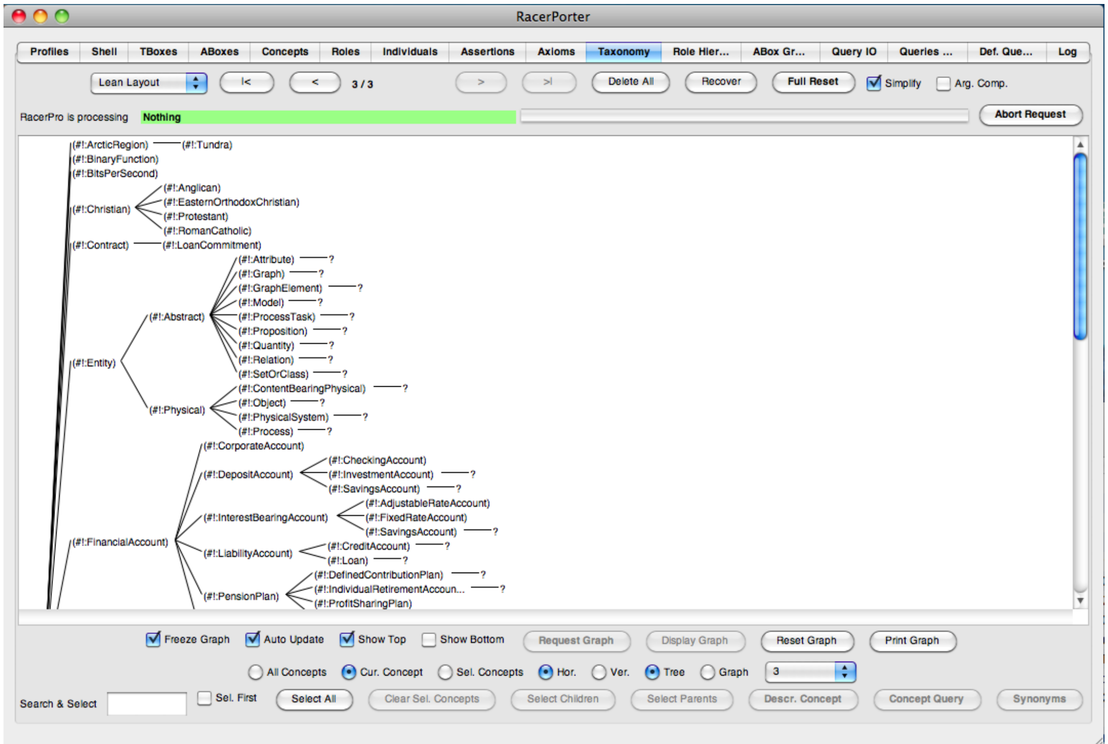

Fig. 2. RacerPorter, a graphical user interface for RacerPro. The taxonomy of the CYC knowledge base is shown. The taxonomy can be interactively unfolded.

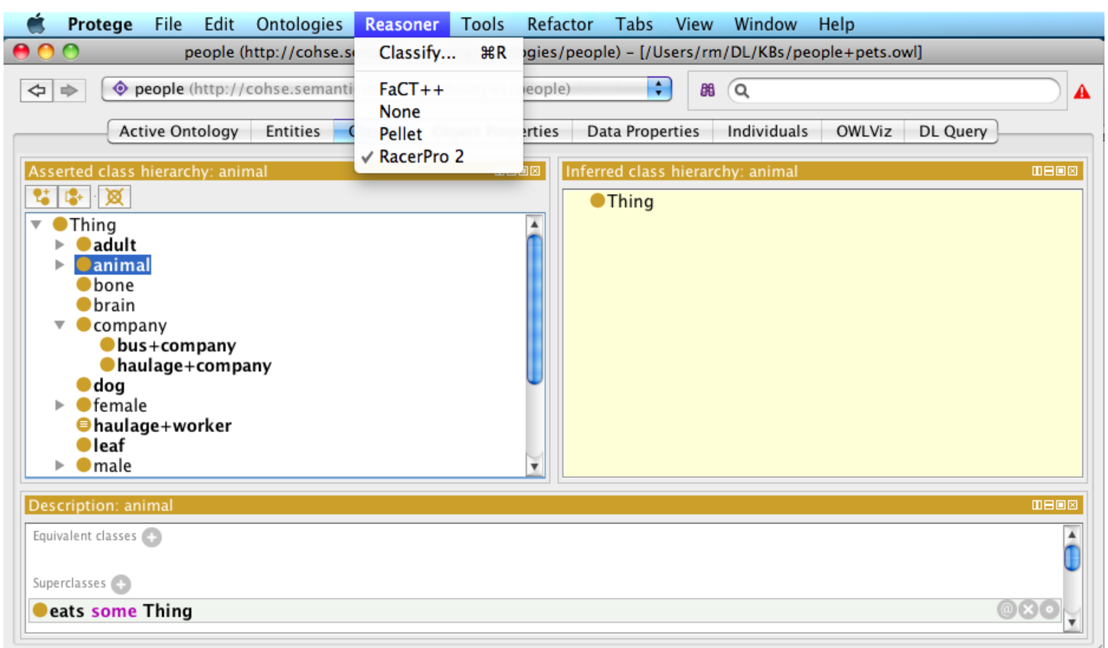

Fig. 3. Protégé 4 with RacerPro selected as the reasoner.

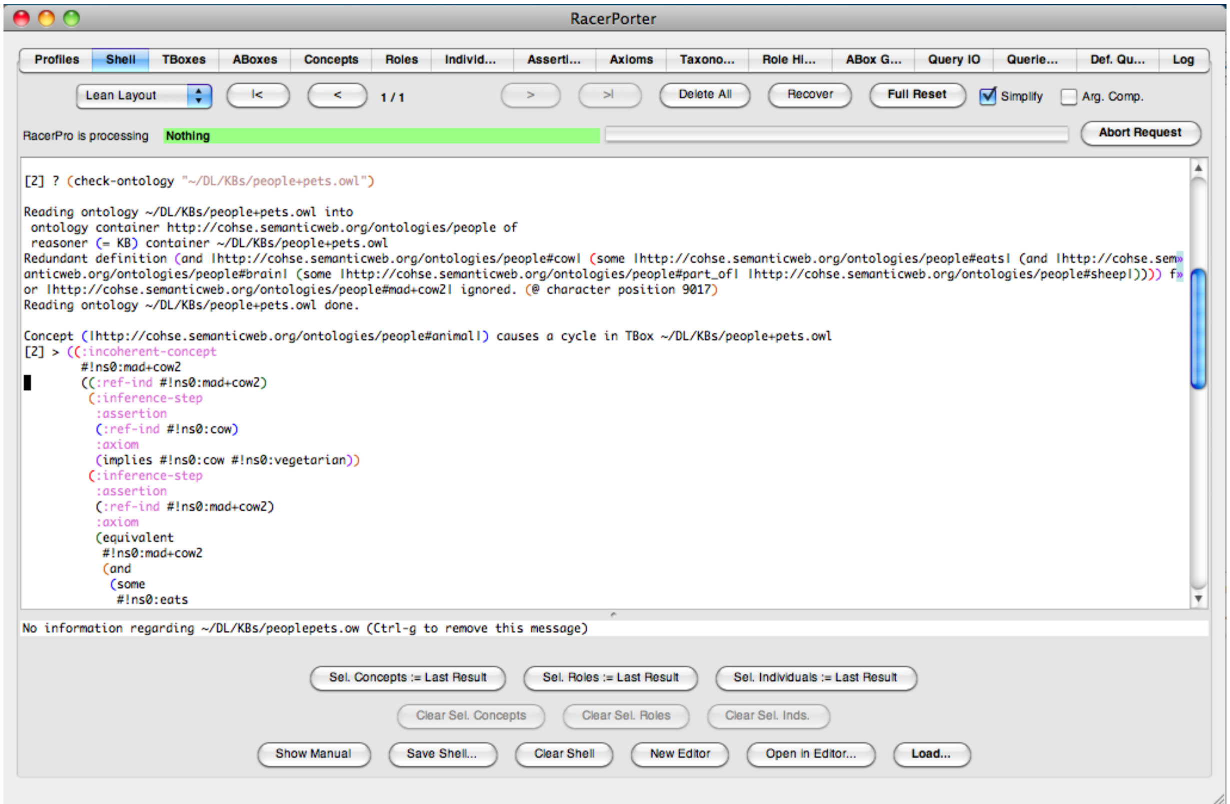

Fig. 4. RacerPorter Shell showing explanation output in a read-eval-print loop.

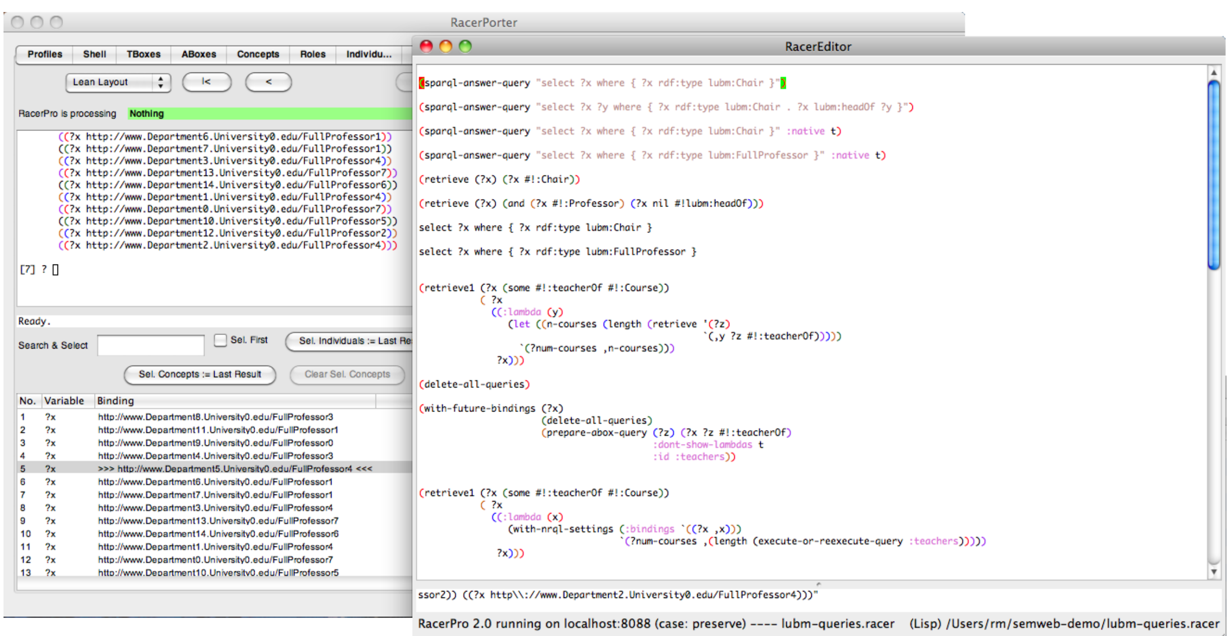

Fig. 5. RacerPorter Editor with commands that can be sent to the server.

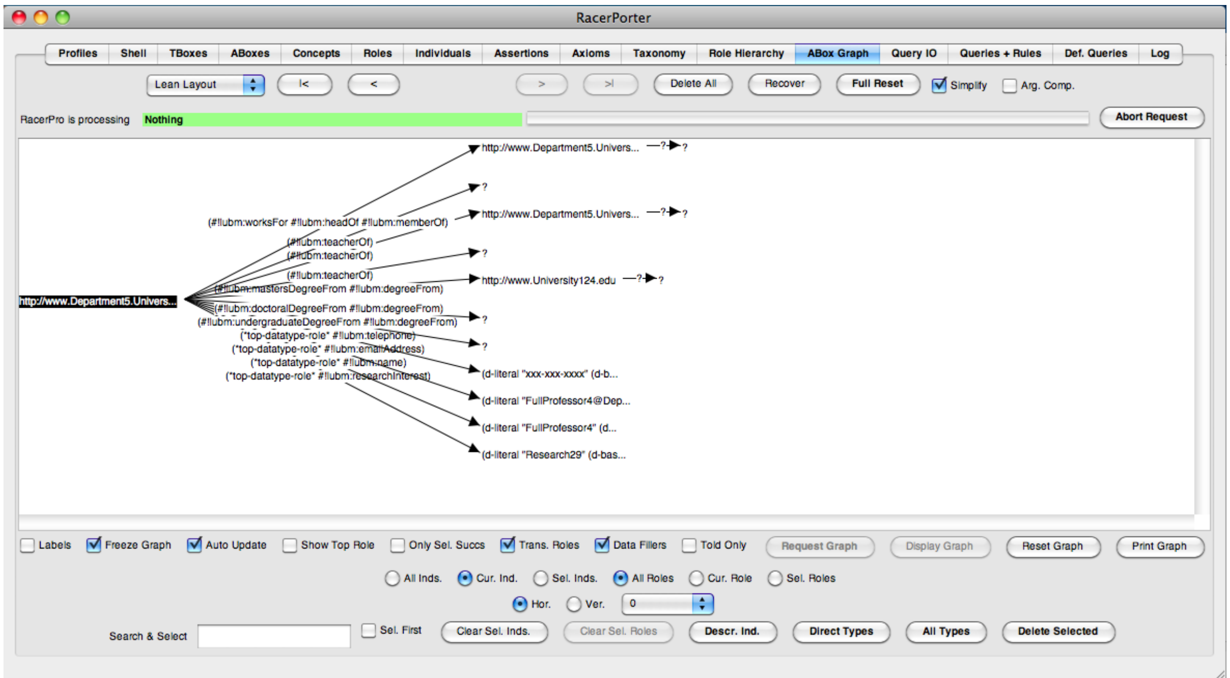

Fig. 6. RacerPro Abox inspector.

The graphical interface RacerPorter [84] can be used to explore axioms and terminologies (see Figure 2.3). It provides a read-eval-print loop (shell) for interactively extending and querying one or more RacerPro servers (see Figure 4), or for inspecting, for instance, explanation results for certain inferences. For the user's convenience, a complete Emacs-style text editor is provided to edit nRQL queries or miniLisp function definitions (see Figure 5). Query results (bindings for variables) can be interactively inspected and Aboxes can interactively explored in the Abox inspector showing told and inferred assertions (see Figure 6).

4. Use Cases

We now discuss some use cases of RacerPro in order to characterize some of the main application areas of RacerPro. Ontology development support is still the most-important application area of description logic reasoning systems. RacerPro provides all standard inference services (subsumption checking, coherence checking, classification). See Figure 2.3 for an example taxonomy shown in RacerPro.

The explanation facility of RacerPro has been used to support the development of the OWL translator for the SUMO ontology [64]. Explanation features for inconsistent concepts are available for knowledge bases as large as Snomed CT (the RacerPro explanation facility uses built-in data structures of the tableau reasoner). It should be noted that Abox reasoning services can be used for problem solving. For instance, in [78], an application of Abox realization for computing solutions to Sudoku problems is presented (note that nominals are not required for this purpose [70]).

Another application area of RacerPro is software engineering. In [69] nRQL has been used to represent integrity constraints as queries which must return an empty result set. This early use case also has shaped the functionalities provided by nRQL. Rob Lemmens has used Racer for investigating the semantic interoperability of distributed geo-services [54].

Abox reasoning (consistency, direct types, etc.) has been used to formalize scene interpretation problems [51]. In particular, it was shown that complete reasoning is necessary for efficiently integrating different "clues" obtained from sensors into a coherent whole. Furthermore, in [51] deduction proved useful to really find interesting object classifications as well as interesting events for making decisions.

Event recognition was also explored in the BOEMIE project [59,17,21] as well as in the ContextWatcher project [81,80]. In the former approach, RacerPro was extended with CLP(R)-like techniques for dealing with quantitative information where in the latter approach the expressivity of nRQL is explored for qualitative event recognition. Another very interesting approach in this context is the use of nRQL as a target language for compiling linear temporal logic (LTL) event descriptions, and using assertional reasoning provided by RacerPro for solving the actual event recognition problem [4].

5. Summary and Outlook

Twelve years of development for RacerPro have passed, and much has been achieved. Although independent benchmarking has revealed that RacerPro is not always the fastest system [77], RacerPro's reputation w.r.t. correctness is very good [56], and the set of features provided with the RacerPro server makes it a unique milestone.

The research perspective behind RacerPro has been to build industrial-strength systems, not just prototypes in order to achieve a symbiosis between practical and theoretical computer science.

Recent research results allow for new areas to be explored, and hence the RacerPro system will be extended in the near future in the following respects:

- Abox modularization, together with sound and complete approximation for implementing query answering for very large Aboxes [76,74,75,52]. This work extends summarization techniques investigated with the SHER system [20],
- Stream-based reasoning [73,26,27],
- Cognitive Agent Framework [28], e.g. for building a Media Interpretation Agent in CASAM,
- Support for parallel reasoning using symmetric multiprocessing (SMP), e.g., for parallel classification as presented in [3],
- Development of software abstractions for building adaptive and flexible reasoning engines using a compositional approach [85].

Acknowledgements

We would like to thank all customers and users of RacerPro for their valuable feedback and support. Special thanks go to Ragnhild van der Straeten, Britta Hummel, Thorsten Liebig, and Marko Luther for testing RacerPro in demanding applications. We also would like thank Olaf Noppens for providing the Protégé 4 plugin for RacerPro. An earlier interface for Racer called RICE was built by Ronald Cornet.

RacerPro is built with Allegro Common Lisp, AllegroGraph, and AllegroServe [2] (it is also possible to build the system using Lispworks Common Lisp and CL-HTTP [57]). RacerPro uses the Wilbur Semantic Web Toolkit [53] for XML and RDF processing. RacerPorter is built with Lispworks Common Lisp.

References

[1] AllegroGraph User Guide. `http://www.franz.com/agraph/allegrograph/`.

[2] Allegroserve. `http://allegroserve.sourceforge.net/`.

[3] M. Aslani and V. Haarslev. Parallel TBox classification in description logics - first experimental results. In *Proceedings of the 19th European Conference on Artificial Intelligence - ECAI 2010, Lisbon, Portugal, Aug. 16-20*, pages 485–490, 2010.

[4] F. Baader, A. Bauer, P. Baumgartner, A. Cregan, A. Gabaldon, K. Ji, K. Lee, D. Rajaratnam, and R. Schwitter. A novel architecture for situation awareness systems. In Martin Giese and Arild Waaler, editors, *Proceedings of the 18th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (Tableaux 2009)*, volume 5607 of *Lecture Notes in Computer Science*, pages 77–92. Springer-Verlag, 2009.

[5] F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence IJCAI-05*, Edinburgh, UK, 2005. Morgan-Kaufmann Publishers.

[6] F. Baader, E. Franconi, B. Hollunder, B. Nebel, and H.-J. Profitlich. An empirical analysis of optimization techniques for terminological representation systems, or: Making $\mathcal{KRIS}$ get a move on. In *Proc. of the 3rd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'92)*, pages 270–281, 1992.

[7] F. Baader, E. Franconi, B. Hollunder, B. Nebel, and H.-J. Profitlich. An empirical analysis of optimization techniques for terminological representation systems or: Making $\mathcal{KRIS}$ get a move on. *Applied Artificial Intelligence. Special Issue on Knowledge Base Management*, 4:109–132, 1994.

[8] F. Baader and B. Hollunder. $\mathcal{KRIS}$: $\mathcal{K}$nowledge $\mathcal{R}$epresentation and $\mathcal{I}$nference $\mathcal{S}$ystem. *SIGART Bull.*, 2(3):8–14, 1991.

[9] F. Baader, C. Lutz, and B. Suntisrivaraporn. CEL—a polynomial-time reasoner for life science ontologies. In U. Furbach and N. Shankar, editors, *Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR'06)*, volume 4130 of *Lecture Notes in Artificial Intelligence*, pages 287–291. Springer-Verlag, 2006. `http://lat.inf.tu-dresden.de/systems/cel/`.

[10] S. Bechhofer, R. Möller, and P. Crowther. The DIG description logic interface. In *Proceedings of the International Workshop on Description Logics (DL-2003), Rome, Italy, September 5-7*, 2003.

[11] A. Borgida. On the relationship between description logics and predicate logic queries. Technical Report LCSR-TR-295-A, Rutgers University, New Brunswick (NJ, USA), 1992.

[12] A. Borgida. Description logics in data management. *IEEE Trans. on Knowledge and Data Engineering*, 7(5):671–682, 1995.

[13] A. Borgida, R. J. Brachman, D. L. McGuinness, and L. Alperin Resnick. CLASSIC: A structural data model for objects. In *Proc. of the ACM SIGMOD Int. Conf. on Management of Data*, pages 59–67, 1989.

[14] A. Borgida and P.F. Patel-Schneider. A semantics and complete algorithm for subsumption in the CLASSIC description logic. *J. of Artificial Intelligence Research*, 1:277–308, 1994.

[15] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Eql-lite: effective first-order query processing in description logics. In *IJCAI'07: Proceedings of the 20th International Joint Conference on Artifical intelligence*, pages 274–279, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[16] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Dl-lite: Tractable description logics for ontologies. In *Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005)*, pages 602–607, 2005.

[17] S. Castano, S. Espinosa, A. Ferrara, V. Karkaletsis, A. Kaya, R. Möller, S. Montanelli, G. Petasis, and M. Wessel. Multimedia interpretation for dynamic ontology evolution. *Journal of Logic and Computation*, 19(5):859–897, 2008.

[18] A.K. Chandra. Theory of database queries. In *Proc. of the 7th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'88)*, pages 1–9, 1988.

[19] M. Dean and G. Schreiber. OWL Web Ontology Language Reference. W3C Recommendation, 10 February 2004. http://www.w3.org/TR/2004/REC-owl-ref-20040210/.

[20] J. Dolby, A. Fokoue, A. Kalyanpur, E. Schonberg, and K. Srinivas. Scalable highly expressive reasoner (SHER). *Web Semant.*, 7(4):357–361, 2009.

[21] S. Espinosa, A. Kaya, and R. Möller. Formalizing multimedia interpretation based on abduction over description logic Aboxes. In *Proc. of the 2009 International Workshop on Description Logics DL- 2009, 27 to 30 July 2009, Oxford, United Kingdom*, 2009. CEUR Workshop Proceedings (Vol. 477).

[22] J. Faddoul and V. Haarslev. Algebraic tableau reasoning for the description logic $\mathcal{SHOQ}$. In *Journal of Applied Logic, Special Issue on Hybrid Logics*, page 31 pages, 2010. Accepted for publication.

[23] J. Faddoul and V. Haarslev. Optimizing algebraic tableau reasoning for $\mathcal{SHOQ}$: First experimental results. In *Proceedings of the 2010 International Workshop on Description Logics (DL-2010), Waterloo, Canada, May 4-7*, pages 161–172., 2010.

[24] J. W. Freeman. *Improvements to Propositional Satisfiability Search Algorithms*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania, 1995.

[25] J. W. Freeman. Hard random 3-SAT problems and the Davis-Putnam procedure. *Artificial Intelligence*, 81:183–198, 1996.

[26] O. Gries, R. Möller, A. Nafissi, M. Rosenfeld, K. Sokolski, and S. Wandelt. Dealing efficiently with ontology-enhanced linked data for multimedia. In *The 5th International Conference on Semantic and Digital Media Technologies (SAMT 2010)*, Lecture Notes in Computer Science. Springer, 2010.

[27] O. Gries, R. Möller, A. Nafissi, M. Rosenfeld, K. Sokolski, and M. Wessel. Media interpretation and companion feedback for multimedia annotation. In *The 5th International Conference on Semantic and Digital Media Technologies (SAMT 2010)*, Lecture Notes in Computer Science. Springer, 2010.

[28] O. Gries, R. Möller, A. Nafissi, M. Rosenfeld, K. Sokolski, and M. Wessel. A probabilistic abduction engine for media interpretation. In J. Alferes, P. Hitzler, and Th. Lukasiewicz, editors, *Proc. International Conference on Web Reasoning and Rule Systems (RR-2010)*, 2010.

[29] W3C OWL Working Group. OWL 2 Web Ontology Language, Document Overview. W3C Recommendation, October 2009. http://www.w3.org/TR/owl2-overview/.

[30] V. Haarslev, K. Hidde, R. Möller, and M. Wessel. RacerPro User's Guide and Reference Manual. Racer Systems GmbH & Co. KG.

[31] V. Haarslev and R. Möller. Expressive ABox reasoning with number restrictions, role hierarchies, and transitively closed roles. In *Proc. of the 7th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2000)*, pages 273–284. Morgan Kaufmann, 2000.

[32] V. Haarslev and R. Möller. High performance reasoning with very large knowledge bases: A practical case study. In B. Nebel, editor, *Proceedings of Seventeenth International JointÃŁ Conference on Artificial Intelligence, IJCAI-01*, pages 161–166, 2001.

[33] V. Haarslev and R. Möller. Optimizing reasoning in description logics with qualified number restrictions. In *Proceedings International Workshop on Description Logics (DL-2001), Stanford, USA, 1.-3. August*, pages 142–151, 2001.

[34] V. Haarslev and R. Möller. Optimization strategies for instance retrieval. In *Proc. of the 2002 Description Logic Workshop (DL 2002)*. CEUR Electronic Workshop Proceedings, http://ceur-ws.org/Vol-53/, 2002.

[35] V. Haarslev and R. Möller. Incremental query answering for implementing document retrieval services. In *Proceedings of the International Workshop on Description Logics (DL-2003), Rome, Italy, September 5-7*, pages 85–94, 2003.

[36] V. Haarslev and R. Möller. Optimization techniques for retrieving resources described in OWL/RDF documents: First results. In *Proc. of the 9th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2004)*, 2004.

[37] V. Haarslev and R. Möller. Optimization techniques for retrieving resources described in OWL/RDF documents: First results. In *Ninth International Conference on the Principles of Knowledge Representation and Reasoning, KR 2004, Whistler, BC, Canada, June 2-5*, pages 163–173, 2004.

[38] V. Haarslev and R. Möller. On the scalability of description logic instance retrieval. *Journal of Automated Reasoning*, 41(2):99–142, 2008.

[39] V. Haarslev and R. Möller. The revival of structural subsumption in tableau-based description logic reasoners. In *Proceedings of the 2008 International Workshop on Description Logics (DL-2008)*, 2008.

[40] V. Haarslev, R. Möller, and A.Y. Turhan. Exploiting pseudo models for TBox and ABox reasoning in expressive description logics. In R. Goré, A. Leitsch, and T. Nipkow, editors, *International Joint Conference on Automated Reasoning, IJCAR'2001, June 18-23, Siena, Italy*, pages 29–44. Springer-Verlag, 2001.

[41] V. Haarslev, R. Möller, and M. Wessel. The description logic alcnhr+ extended with concrete domains: A practically moti-

vated approach. In R. Goré, A. Leitsch, and T. Nipkow, editors, *International Joint Conference on Automated Reasoning, IJCAR'2001, June 18-23, Siena, Italy*, pages 29–44. Springer-Verlag, 2001.

[42] V. Haarslev, R. Möller, and M. Wessel. Querying the semantic web with Racer + nRQL. In *Proceedings of the KI-2004 International Workshop on Applications of Description Logics (ADL'04), Ulm, Germany, September 24*, 2004.

[43] V. Haarslev, R. Möller, and M. Wessel. Description logic inference technology: Lessons learned in the trenches. In I. Horrocks, U. Sattler, and F. Wolter, editors, *Proc. International Workshop on Description Logics*, 2005.

[44] B. Hollunder and F. Baader. Qualifying number restrictions in concept languages. In *Proc. of the 2nd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'91)*, pages 335–346, 1991.

[45] M. Horridge and S. Bechhofer. The OWL API: A Java API for Working with OWL 2 Ontologies. In *OWLED 2009, 6th OWL Experienced and Directions Workshop, Chantilly, Virginia*, 2009.

[46] I. Horrocks. *Optimising Tableaux Decision Procedures for Description Logics*. PhD thesis, University of Manchester, 1997.

[47] I. Horrocks. Using an expressive description logic: FaCT or fiction? In *Proc. of the 6th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'98)*, pages 636–647, 1998.

[48] I. Horrocks, O. Kutz, and U. Sattler. The even more irresistible $\mathcal{SROIQ}$. In *Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 57–67. AAAI Press, 2006.

[49] I. Horrocks and U. Sattler. A tableaux decision procedure for $\mathcal{SHOIQ}$. In *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005)*, Edinburgh (UK), 2005. Morgan Kaufmann.

[50] I. Horrocks, U. Sattler, and St. Tobies. Reasoning with individuals for the description logic $\mathcal{SHIQ}$. In David McAllester, editor, *Proc. of the 17th Int. Conf. on Automated Deduction (CADE 2000)*, volume 1831 of *Lecture Notes in Computer Science*, pages 482–496. Springer, 2000.

[51] B. Hummel. *Description Logic for Scene Understanding at the example of Urban Road Intersections*. Südwestdeutscher Verlag für Hochschulschriften, 2010.

[52] A. Kaplunova, R. Möller, S. Wandelt, and M. Wessel. Towards scalable instance retrieval over ontologies. In Bi Yaxin and Williams Mary-Anne, editors, *Knowledge Science, Engineering and Management, Fourth International Conference, KSEM 2010, Proceedings*, volume 6291 of *Lecture Notes in Computer Science*. Springer, 2010.

[53] O. Lassila. Wilbur semantic web toolkit for CLOS. `http://wilbur-rdf.sourceforge.net/`.

[54] R.L.G. Lemmens. *Semantic interoperability of distributed geo-services*. PhD thesis, International Institute for Geo-Information Science and Earth Observations (CT), Delft, Netherlands Geodetic Commission NCG, 2006.

[55] T. Liebig, M. Luther, O. Noppens, M. Rodrigues, D. Calvanesez, M. Wessel, R. Möller, M. Horridge, S. Bechhofer, D. Tsarkov, and E. Sirin. OWLlink: DIG for OWL 2. In *Fifth International Workshop on OWL: Experiences and Directions (OWLED08), Karlsruhe, Germany*, 2008.

[56] M. Luther, T. Liebig, S. Böhm, and O. Noppens. Who the heck is the father of bob? - A survey of the owl reasoning infrastructure for expressive real-world applications. In *Proceedings of the European Semantic Web Conference (ESWC 2009), Heraklion, Greece*. Springer Verlag, 2009.

[57] J. Mallery. A Common LISP Hypermedia Server. In *In Proc. First International World-Wide Web Conference*, pages 239–247, 1994. `http://www.cl-http.org:8001/`.

[58] R. Möller, V. Haarslev, and M. Wessel. On the scalability of description logic instance retrieval. In Chr. Freksa and M. Kohlhase, editors, *29. Deutsche Jahrestagung für Künstliche Intelligenz*, Lecture Notes in Artificial Intelligence. Springer Verlag, 2006.

[59] R. Möller and B. Neumann. Ontology-based Reasoning Techniques for Multimedia Interpretation and Retrieval. In *Semantic Multimedia and Ontologies : Theory and Applications*. Springer, 2008.

[60] B. Motik. *Reasoning in Description Logics using Resolution and Deductive Databases*. PhD thesis, Univesität Karlsruhe (TH), Karlsruhe, Germany, January 2006.

[61] B. Motik, R. Shearer, and I. Horrocks. Hypertableau reasoning for description logics. *J. of Artificial Intelligence Research*, 36:165–228, 2009.

[62] O. Noppens. RACER reasoning support for OWLAPI and Protégé. `http://www.uni-ulm.de/in/ki/semantics/owltools`.

[63] O. Noppens, M. Luther, and Th. Liebig. The OWLlink protocol: Infrastructure for interfacing and managing OWL 2 reasoning systems. In *OWLED 2009, 6th OWL Experienced and Directions Workshop, Chantilly, Virginia, October 2009*, volume 529 of *CEUR Workshop Proceedings*, 2009.

[64] A. Pease and I. Niles. IEEE Standard Upper Ontology: A progress report. *Knowledge Engineering Review, Special Issue on Ontologies and Agents*, 17:65–70, 2002. `http://www.ontologyportal.org/`.

[65] E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF. W3C Recommendation, January 2008. http://www.w3.org/TR/rdf-sparql-query/.

[66] E. Sirin, B. Cuenca Grau, and B. Parsia. From wine to water: Optimizing description logic reasoning for nominals. In P. Doherty, J. Mylopoulos, and Chr. A. Welty, editors, *KR*, pages 90–99. AAAI Press, 2006.

[67] E. Sirin and B. Parsia. Optimizations for answering conjunctive ABox queries. In *Proc. of the 2006 International Workshop on Description Logics DL'06*, 2006.

[68] E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *Web Semant.*, 5(2):51–53, 2007.

[69] R. Van Der Straeten. *Inconsistency Management in Model-Driven Engineering: An Approach using Description Logics*. PhD thesis, Vrije Universiteit Brussel, 2005.

[70] Sudoku in OWL. `http://www.mindswap.org/~aditkal/sudoku`.

[71] D. Tsarkov, I. Horrocks, and P. F. Patel-Schneider. Optimizing terminological reasoning for expressive description logics. *J. of Automated Reasoning*, 39(3):277–316, 2007.

[72] A.-Y. Turhan. *On the Computation of Common Subsumers in Description Logics*. PhD thesis, TU Dresden, Institute for Theoretical Computer Science, 2008.

[73] S. Wandelt and R. Möller. Updatable island reasoning over alchi ontologies. In *Conference on Knowledge Engineering and Ontology Development (KEOD)*, 2009. CEUR Workshop Proceedings (Vol. 477).

[74] S. Wandelt and R. Möller. Distributed island-based query answering for expressive ontologies. In V. Haarslev, D. Toman, and G. Weddell, editors, *Proceedings of the 2010 International Workshop on Description Logics (DL2010)*, volume 573 of *CEUR-WS*, pages 185–196, 2010.

[75] S. Wandelt and R. Möller. Distributed island-based query answering for expressive ontologies. In *Advances in Grid and Pervasive Computing, 5th International Conference, GPC 2010, Hualien, Taiwan, May 10-13, 2010. Proceedings*, volume 6104 of *Lecture Notes in Computer Science*, pages 461–470. Springer, 2010.

[76] S. Wandelt and R. Möller. Sound summarizations for alchi ontologies - how to speedup instance checking and instance retrieval. In *Second International Conference on Agents and Artificial Intelligence (ICAART)*. INSTICC Press, 2010.

[77] T. Weithöner, Th. Liebig, M. Luther, , and S. Böhm. What's Wrong with OWL Benchmarks? In *Proc. of the Second Int. Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS 2006)*, pages 101–114, Athens, GA, USA, November 2006.

[78] M. Wessel. Solving Sudoku problems using Abox reasoning. `http://www.sts.tu-harburg.de/people/mi.wessel/papers/sudoku.racer`.

[79] M. Wessel. *Flexible und konfigurierbare Software-Architekturen für datenintensive ontologiebasierte Informationssysteme*. PhD thesis, Hamburg University of Technology, 2008. ISBN 978-3-8325-2162-2.

[80] M. Wessel, M. Luther, and R. Möller. What happened to bob? semantic data mining of context histories. In *Proc. of the 2009 International Workshop on Description Logics DL-2009, Oxford, United Kingdom*, volume 477 of *CEUR Workshop Proceedings*, 2009.

[81] M. Wessel, M. Luther, and M. Wagner. The difference a day makes - recognizing important events in daily context logs. In *Proceedings of the International Workshop on Contexts and Ontologies: Representation and Reasoning (C&O:RR),*, volume 298 of *CEUR Workshop Proceedings*, 2007.

[82] M. Wessel and R. Möller. A high performance semantic web query answering engine. In I. Horrocks, U. Sattler, and F. Wolter, editors, *Proc. International Workshop on Description Logics*, 2005.

[83] M. Wessel and R. Möller. A flexible dl-based architecture for deductive information systems. In G. Sutcliffe, R. Schmidt, and S. Schulz, editors, *Proc. IJCAR-06 Workshop on Empirically Successful Computerized Reasoning (ESCoR)*, pages 92–111, 2006.

[84] M. Wessel and R. Möller. Design principles and realization techniques for user friendly, interactive, and scalable ontology browsing and inspection tools. In *International Workshop on OWL: Experiences and Directions (OWLED 2007)*, 2007.

[85] M. Wessel and R. Möller. Software abstractions for description logic systems. In *Proceedings of the 5th European Lisp Workshop (ELW'08)*, 2008.

[86] J. Wu and V. Haarslev. Planning of axiom absorption. In *Proceedings of the 2008 International Workshop on Description Logics (DL-2008), Dresden, Germany*, 2008.

[87] M. Zuo and V. Haarslev. High performance absorption algorithms for terminological reasoning. In *Proceedings of the 2006 International Workshop on Description Logics (DL-2006), Lake District, UK*, pages 159–166, 2006.