A survey of current, stand-alone OWL Reasoners

Nicolas Matentzoglu, Jared Leo, Valentino Hudhra, Bijan Parsia, and Uli Sattler

The University of Manchester
Oxford Road, Manchester, M13 9PL, UK
{matentzn,jared.leo,hudhrav,bparsia,sattler}@manchester.ac.uk

Abstract. We present a survey of the current OWL reasoner landscape. Through literature and web search we have identified 35 OWL reasoners that are, at least to some degree, actively maintained. We conducted a survey directly addressing the respective developers, and collected 33 responses. We present an analysis of the survey, characterising all reasoners across a wide range of categories such as supported expressiveness and reasoning services. We will also provide some insight about ongoing research efforts and a rough categorisation of reasoner calculi.

Keywords: ontologies,reasoning,OWL

1 Introduction

The utility of reasoner surveys is twofold: the main purpose is to inform users about the systems they can potentially choose from, the secondary purpose is to give an overview of the reasoner landscape, both to reasoner developers and other related researchers. In this work, we focus on the second purpose. Instead of providing a detailed list of reasoners, we group them according to our classification scheme and report on commonly exhibited features. We discuss features such as logical services, licensing and supported OWL profiles. Rather than consulting potentially outdated literature, we decided to directly address reasoner developers to fill in a detailed survey involving questions about parallelism, mobile support and use of modules.

Our main contributions are (1) an exhaustive survey of stand-alone, current reasoners supporting reasoning with OWL or a fragment of it, (2) a discussion of relevant reasoner characteristics and (3) the foundations for a future knowledge base of OWL implementations.

2 Preliminaries

The reader should have a general familiarity with description logics and reasoning. A shallow understanding of description logic reasoning services, semantics and language expressivity should suffice.

3 Related work

There are two main types of reasoner surveys: (1) reasoner benchmarks and (2) reasoner meta-data surveys. Studies of the first type aim at understanding and comparing reasoner performance and generally involve a more or less systematic ontology gathering strategy, but will be, in practice, much more unsystematic with respect to the choice of reasoners to be evaluated. This choice is usually driven by practical considerations, such as the implementation of a particular interface (OWL API, ORE) and intuitions about the importance of the reasoners. For example, we might believe that Pellet, HermiT, ELK and FaCT++ are more important than other reasoners, because we have the impression that they are used more often, cited more often, or simply bundled with Protégé. Reasoner benchmarks attempt to establish the utility of a particular reasoning system for a particular set of cases, or test the effect of particular optimisations. The second type of surveys are reasoner meta-data surveys. Their aim is to list existing systems, provide an overview of the supported features and, ideally, organise the landscape. This work is of the second kind.

While reasoner benchmarks have recently received increasing attention, for example in the ORE reasoner competition [13], exhaustive meta-data surveys and reasoner listings are sparse and rarely comprehensive or up-to-date [29, 39]. The most comprehensive relatively recent study [22] provides an *historical account* of description logic reasoners from the early implementations (1975) until modern OWL reasoners such as HermiT. The authors analysed each reasoner with respect to four basic aspects: (1) Inference support (reasoning services, reasoner type), (2) basic components (supported languages, other features), (3) algorithm completeness and (4) programming languages. The accounts are relatively unstructured and do not lend themselves to automated comparisons, aggregations and as a basis for reasoner selections for users. The survey is supposed to provide a brief historical introduction into the subject matter, and constituted one of the starting points of our own investigation. Another study focuses on reasoners that the author deemed suitable for either Protégé or the NeOn toolkit [1]. The survey does describe some axes along which reasoners could be classified, and provides a comparative account across features such as expressivity, OWL API support or the ability to perform incremental classification. However, the study does not go into much detail, is not exhaustive (many relevant reasoners like JFact, jcel and Konclude are missing), and from a methodological perspective, there is a lack of explanation on how the data was obtained and validated. A nice study, focusing on reasoners that are able to deal with large EL ontologies, lists 8 reasoners and compares them in terms of supported expressivity, soundness, completeness, rule support and a few other features [7]. This study is a hybrid between a benchmark and a meta-data study, as it not only compares reasoner features, but also evaluates their performance against a handful of important biomedical ontologies. Its main contribution is the definition of relevant characteristics that guide reasoner choice of users. Our survey covers most of the dimensions they describe, except for rules and details with respect to ABox reasoning tasks.

4 OWL Reasoner Characteristics

Not all reasoners are intended to support the entirety of OWL 2 DL. Reasoners such as HermiT and FaCT++ support all of OWL 2 DL, while many efficient reasoners exist that only deal with tractable profiles of OWL and OWL 2; ELK and CEL are examples of highly efficient reasoners that handle (most of) OWL 2 EL. There are also many reasoners that deal with extensions of classical DLs, such as fuzzyDL, a reasoner for fuzzy extensions to DLs. Amongst this variation are the reasoning services that the reasoners offer; some are very efficient at TBox queries such as classification (computing subsumption relations between named classes), whereas others are meant for query answering. From a utilitarian perspective, we believe that *supported reasoning services*, *expressivity levels* (including the degree of datatype support) and the *completeness* of the implemented algorithm are the most important characteristics to categorise a reasoner. A second way to classify the reasoner is by the primary calculus underlying its core reasoning service, for example tableau or consequence-based procedures. This distinction is primarily useful for researchers that are interested in finding more efficient ways to solve reasoning problems.

4.1 Utility

Modern OWL reasoners support a number of *primary logical services* such as consistency, classification, instance checking and entailment checking, and secondary services such as explanation for entailments and inconsistency, (semantic) module extraction and ontology based data access (OBDA). Explaining these services in detail is beyond the scope of this survey. It should suffice to say that some reasoners are particularly optimised to operate on concept level (TBox) knowledge, for example offering an efficient classification service, while other reasoners offer the user services for efficient instance retrieval (ABox level), most notably through conjunctive query answering. In our work, we attempted to determine key and ancillary reasoning services for all reasoners participating in our survey.

We define the *expressivity* of a language as the quality and conceptual breadth of the constructs which make up the language. The expressivity supported by a reasoner is the most expressive language for which it can provide a key reasoning service. As such, expressivity is not an attribute of the reasoner, but of a reasoning service: a reasoner might offer different sound reasoning services for varying levels of expressivity. There are many ways expressivity can be documented and communicated, for example: (1) The most expressive description logic fragment supported (such as ALC or SROIQ), (2) the set of OWL profiles supported and (3) the set of OWL profiles efficiently supported. To simplify things: (1) is usually communicated as a single language, however, since the expressivity levels of all known DL fragments can be organised into a partial order, the expressivity of a reasoner refers to a set of languages, (2) is usually used to indicate the expressivity of a reasoner based on the OWL profiles it is compatible with and (3) is an extension of (2) where efficiency is the main focus. As an example, consider the 2 reasoners HermiT and ELK. W.r.t (1), HermiT captures the set of DLs below and including $\mathcal{SROIQ}(\mathcal{D})$ whereas ELK similarly captures ($\mathcal{ELRO}$). W.r.t (2), the expressivity of HermiT is higher than ELK since HermiT captures OWL 2 DL and ELK only captures OWL 2 EL and finally, w.r.t (3) it is known that ELK performs better than HermiT w.r.t OWL 2 EL ontologies. In our survey, we obtained all three kinds of measurements for expressivity, but we will only report on (1) and (2). For the reasoner classification, we only consider known fragments of DL and standard OWL profiles respectively. As noted above, many reasoners are implemented to support extensions to these languages or profiles, for example probabilistic, temporal or fuzzy extensions. Although these differ considerably, they still use common calculi and are categorised accordingly.

In general, a *sound* algorithm will never answer YES if NO would be true and a *complete* algorithm will never answer NO if YES would be true. For example, given an algorithm P, an ontology $\mathcal{O}$, and an axiom α over $\mathcal{O}$ of the form $A \sqsubseteq B$ where $\mathcal{O} \models \alpha$, $P(\mathcal{O}, \alpha)$ says YES *iff* $\mathcal{O} \models \alpha$, where soundness is the $\rightarrow$ direction and completeness is the $\leftarrow$ direction. Both can be seen as important and required characteristics of a reasoner. To show why, a trivial implementation of a sound and incomplete algorithm answers NO to every question, where as a trivial implementation of a unsound but complete algorithm answers YES to every question - both can be seen as irrelevant if not taken together. While it would make sense to classify each reasoning service individually on whether the underlying algorithms are sound and complete, we simplify our measurement to determine what the algorithm underlying the key reasoning service is. In our survey, we distinguish broad categories of completeness, but assume that all procedures are generally sound.

4.2 Calculus

We identified 3 main categories of calculi: consequence-, model construction- and rewriting-based. In general, *consequence-based* reasoners rely on adding logical consequences (entailments) to a knowledge base (KB) without the need to check possible consequences that are not entailed by the knowledge base. This category covers similar techniques such as resolution-based techniques, rule-based procedures and completing algorithms. These techniques are most commonly employed for reasoners that deal only with the fragments of OWL 2 DL such as EL+ and EL++, and are used by some popular reasoners such as ELK and CEL. *Model construction* techniques are based on building models based on the KB and checking for KB consistency. These are generally utilised for quite highly expressive fragments of OWL 2 DL, i.e. those extending ALC. This category includes those reasoners that use automata-based approaches, tableau and hypertableau techniques. *Rewriting* is a technique used to expand a KB by rewriting the facts of the KB, to be used for a specific task, e.g. query answering, making the reasoner less reliant on the terminological aspect and more involved in the data aspect. This technique is most commonly used for query rewriting and catalogue rewriting but can also be used for structural transformation of a KB.

While we attempted to determine which calculus was primarily employed by the reasoner developers, there seems to be a trend, at least for the general purpose reasoners that cover the entirety of OWL 2 DL reasoning, to employ hybrid techniques, for example combining saturation-based or consequence-based techniques with tableau-style techniques (Pellet, Konclude, WSClassifier and others).

5 Materials and Methods

The subjects of this survey are current, stand-alone reasoners that can deal with OWL. We defined *current* as either being updated or published over since January 2012. Being able to *deal with OWL* means that at least one of the standard reasoning services above was supported on a defined fragment of the OWL. *Stand-alone* means that the reasoner is meant to be used by itself, rather than being a dependent component of a bigger system, such as the various inference engines as part of triple stores.

This survey emerged as a by-product of a systematic review on DL reasoning system evaluation quality. It is however by itself not strictly systematic. The initial list of reasoners was created from publicly maintained lists [29, 39]. More reasoners were found through respective Google searches. This knowledge directly informed the search strategy for the systematic review mentioned before, which again yielded a large range of system description papers that lead to amending the full list of reasoners. After the search was completed, our list included 68 description logic and OWL reasoners, with another two being added later as part of a feedback round from reasoner developers.

As a first step, we attempted to find evidence of use of each reasoner by searching Google and Google Scholar for web pages that led to version control systems or other kinds of download pages as well as academic papers. We recorded the most up to date evidence we could find (for example, the latest commit in a version control, the publication date of the most recent paper). If a reasonable attempt at finding evidence failed (repeated searches with varying keyword combinations, references in perhaps older system description papers), we excluded the reasoner from the survey. Apart from the 35 reasoners we mention in this survey, the original list contained the following reasoners: COROR, RacerPro, *SAT, BACK, CB, Cerebra Engine, CICLOP, CLASSIC, Condor, CRACK, DLP, Fact, FLEX, HAM-ALC, K-REP, Bossam, KRIS, LOOM, MSPASS, QuOnto, SHER, YAK, OWLGres, Pronto, DLEJena, F-OWL, Fresg, OWLer, OntoMinD, Screech, REQUIEM, YARR!, Kaon2, Elly and SoftFacts. Most of these reasoners are quite old description logic reasoners, some reasoners, such as RacerPro, are superseded by other reasoners (Racer). For none of these reasoners we were able to find proof that they were used or published about since January 2012. The YARR! reasoner did have a workshop paper in 2013 [30], but we decided to exclude it from the survey because we were unable to find a more substantial proof of existence (like a web page). We sent our survey to the developers of the remaining 35 reasoners.

The survey itself was implemented as a series of questions in SurveyMonkey[1], as a joint effort of a team of experts including reasoner developers, logicians and

[1] `https://www.surveymonkey.com/r/6F9K89X`

empirical researchers. The results were used to update the popular reasoner listing[2] at Manchester.

6 Survey Results and Discussion

Out of the 35 surveys sent out to reasoner developers, we collected 33 complete answers. Surveys for OwlOntDB [10] and Deslog [40] were not completed at the time of this writing. An overview of all participating systems can be found in Table 1 (see appendix). As discussed in the introduction, we do not aim to provide a discussion of the individual reasoners. Additional information can be found on the Manchester University pages mentioned in the previous section.

6.1 General information

As can be seen in Figure 1, the majority of reasoners were developed in the last five years (23 out of 33). However, 3 reasoners have been around for more than ten years, all of which have been updated during the last two years. 23 are actively developed, 8 are merely maintained (bugfixes etc.) and two reasoners, CEL and HermiT, are not maintained at all anymore. In terms of implementation, most reasoners are based on the Java programming language. This is not surprising, given that the dominant frameworks supporting OWL, such as the OWL API [16] and Jena, are implemented in Java. As can be seen in Figure 2 there are, interestingly enough, a good number or reasoners implemented in other languages though, such as the more efficient C/C++ or Prolog. More than two thirds of the reasoners in the set implement the OWLReasoner interface of the OWL API. Like other questions in the survey this one was optional, thus merely establishing a lower bound, but it already demonstrates the ubiquity of the framework. The majority of reasoners are accessible via the command line, or are readily available for integration with the Protégé editor, also Figure 2. The dominant underlying primary calculus is tableau (10 out of 33), but only just. It is inherently difficult to classify modern reasoners accurately. Konclude for example uses a hybrid approach that involves tableau and some nested saturation-based techniques. Many full fledged OWL 2 reasoners come with an efficient consequence-based delegate reasoner for EL ontologies, such as Pellet, FaCT++ or WSClassifier. For this survey, we grouped the different techniques into the categories described in Section 4.2. As can be see in Figure 2, most reasoners are primarily based on an approach that involves model construction.

Few reasoner developers report to make *use of modularity*, perhaps surprisingly. Only four reasoners, including full fledged modular reasoners such as MORe and Chainsaw, report to make use of modularity for classification, and only three reasoners (Pellet, FaCT++ and DistEL) are reported to make use of modularity for incremental reasoning. Only 6 reasoners are known to draw on parallelism to improve classification performance, and only 4 mobilise it during pre-processing. Up until today, little is known whether parallel techniques like

[2] `http://owl.cs.manchester.ac.uk/tools/list-of-reasoners/`

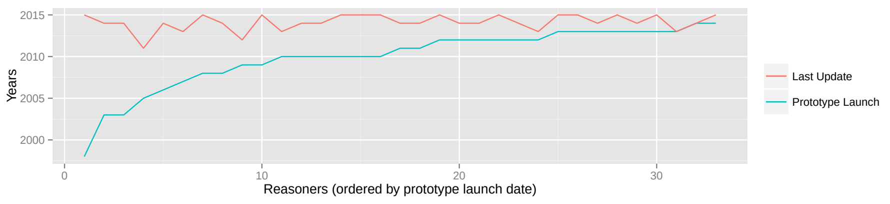

Fig. 1. Timeline of OWL reasoners.

parallel or-branching or parallel module classification actually do much good, especially for tableau-style reasoning. Three reasoners, CEL, snorocket and WS-Classifier have been reported to be tuned to particular ontologies (SNOMED CT the two former, a version of FMA the latter). Given the growing interest for mobile technology, we also asked whether the reasoner has been run on a mobile device. 5 reasoners where run on Android (ELK, JFact, jcel, LIFr, BaseVIsor), 24 where never run on a mobile device and for 4 reasoners the developers did not give an answer.

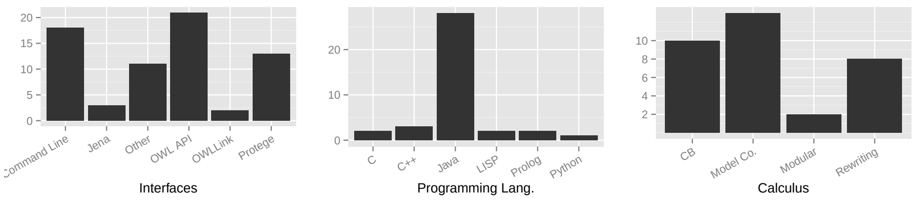

Fig. 2. Main programming language the reasoner is written in, interfaces provided and the main category of calculi underpinning the reasoner.

6.2 Expressivity and services

Some reasoners do not merely provide reasoning services for standard description logics. A minimal support for the most important extension, datatypes, is provided by many reasoners. In our survey however, 16 reasoner developers did not give any indication for datatype support. Full support of all datatypes on the OWL 2 datatype map is reported for 6 reasoners. The question was optional, however, so the numbers have to be interpreted cautiously. Other prominent extensions are fuzzy with 3 reasoners, probabilistic with 4 and distributed reasoning with 2 reasoners in the survey. Multiple extensions to standard description logic (for example distributed-fuzzy) reasoning do not seem to occur at all.

As was discussed in Section 4.1, we will report the supported language expressivity in two ways: Supported OWL profile and the main underlying description logic family. Both can be seen in Figure 3. For readability, we have grouped the DL languages into 3 rough categories: Horn-style languages includes EL, RL, Horn-SHIQ and similar. Horn logics are those that provide no means to express any form of disjunction, either directly or indirectly. They do not require any form of "reasoning by case", often have nice canonical models, and are generally suited for consequence-based reasoning. To the right of the figure, we summarize the degree of completeness of the underlying algorithms. SC stands for soundness/completeness, SI stands for sound and incomplete, SC/Profile means that the reasoner is sound and complete for one of the profiles, SC/OWL1 sound and complete for OWL 1 and so on. Only six reasoners in the set are incomplete, and the majority of reasoners are complete at least for one of the polytime OWL profiles. This is also consistent with the supported profiles: The majority of reasoners support OWL 2 EL, and at least 12 reasoners are directly recommended for the OWL 2 EL profile (see questionnaire).

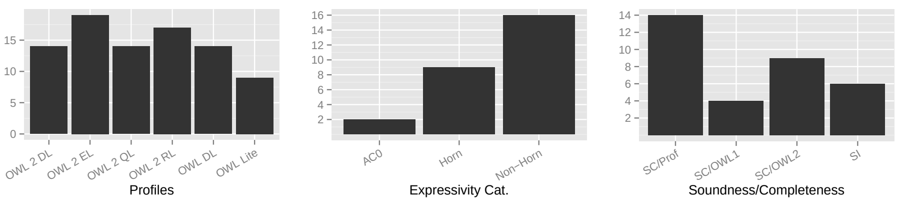

Fig. 3. Supported levels of expressivity and completeness.

Most reasoners support simple satisfiability checking (22 out of 33), knowledge base consistency (24), entailments checking (19) and classification (21). Fewer reasoners are asserted to support ABox related tasks, such as realisation (15) and conjunctive query answering (11).

6.3 Ongoing Research and future directions

Beyond the obvious need for improved evaluations and support for increased expressive languages, many current developments focus on optimisations related to parallelism and concurrency. At least 6 reasoner developers are working to integrate or improve some aspect of the reasoning with parallel techniques. Another working topic at the moment appears to be incremental reasoning. 4 reasoner developers report to be actively working on integrating support for incremental reasoning. Other topics of interest include, but are not limited to: Reasoning with large scale datasets, meta-modeling and meta-reasoning, conditional completeness-guaranteed of approximation results, proof-based explanations, integration of a novel approach to module extraction and support of extra interfaces to allow client code to access the tableaux structure.

7 Conclusions and Future Work

We hope that our work will inform reasoner developers of other relevant projects in their field, and support them to get a sense of the coverage and directions of current reasoner development trends. We tried to get an idea of (intended) use for concurrent or parallel technologies, as well as modules. The survey we conducted covers many more aspects than we could report on within the limits of this paper. We asked for supported datatypes individually, employed reasoner optimisations and recommended usage. The resulting data will form the starting point for a knowledge base about reasoning systems and ontologies. We have already started working on an OWL reasoner ontology (ORO) that attempts to model language expressivity, calculi and reasoner optimisations in a fine grained fashion, to serve as a schema for the knowledge base. ORO is directly informed by the outcomes of this survey and is, in its very prototypical first version, available online[3].

References

1. S. Abburu. A Survey on Ontology Reasoners and Comparison. *International Journal of Computer Applications*, 57(17):33–39, Nov. 2012. Full text available.
2. F. Baader, C. Lutz, and B. Suntisrivaraporn. CEL - A Polynomial-Time Reasoner for Life Science Ontologies. In *Automated Reasoning, Third International Joint Conference, IJCAR 2006, Seattle, WA, USA, August 17-20, 2006, Proceedings*, pages 287–291, 2006.
3. J. Bak, M. Nowak, and C. Jedrzejek. RuQAR: Reasoning Framework for OWL 2 RL Ontologies. In *The Semantic Web: ESWC 2014 Satellite Events - ESWC 2014 Satellite Events, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 195–198, 2014.
4. F. Bobillo, M. Delgado, and J. Gmez-Romero. DeLorean: A reasoner for fuzzy OWL 2. *Expert Syst. Appl.*, 39(1):258–272, 2012.
5. F. Bobillo and U. Straccia. fuzzyDL: An expressive fuzzy description logic reasoner. In *FUZZ-IEEE 2008, IEEE International Conference on Fuzzy Systems, Hong Kong, China, 1-6 June, 2008, Proceedings*, pages 923–930, 2008.
6. D. Calvanese, G. D. Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodriguez-Muro, R. Rosati, M. Ruzzi, and D. F. Savo. The MASTRO system for ontology-based data access. *Semantic Web*, 2(1):43–53, 2011.
7. K. Dentler, R. Cornet, A. t. Teije, and N. d. Keizer. Comparison of reasoners for large ontologies in the OWL 2 EL profile. *Semantic Web*, 2(2):71–87, 2011.
8. C. L. Duc, M. Lamolle, A. Zimmermann, and O. Cur. DRAOn: A Distributed Reasoner for Aligned Ontologies. In *Informal Proceedings of the 2nd International Workshop on OWL Reasoner Evaluation (ORE-2013), Ulm, Germany, July 22, 2013*, pages 81–86, 2013.
9. T. Eiter, M. Ortiz, M. Simkus, T.-K. Tran, and G. Xiao. Query Rewriting for Horn-SHIQ Plus Rules. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada.*, 2012.
10. R. U. Faruqui and W. MacCaull. O wl O nt DB: A Scalable Reasoning System for OWL 2 RL Ontologies with Large ABoxes. In *Foundations of Health Information Engineering and Systems - Second International Symposium, FHIES 2012, Paris, France, August 27-28, 2012. Revised Selected Papers*, pages 105–123, 2012.
11. M. d. M. R. Garca and J. F. A. Montes. Evaluating DBOWL: A Non-materializing OWL Reasoner based on Relational Database Technology. In *Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE-2012), Manchester, UK, July 1st, 2012*, 2012.

[3] `https://github.com/matentzn/owlreasonerontology`

12. B. Glimm, I. Horrocks, B. Motik, G. Stoilos, and Z. Wang. HermiT: An OWL 2 Reasoner. *J. Autom. Reasoning*, 53(3):245–269, 2014.
13. R. S. Gonalves, S. Bail, E. Jimnez-Ruiz, N. Matentzoglu, B. Parsia, B. Glimm, and Y. Kazakov. OWL Reasoner Evaluation (ORE) Workshop 2013 Results: Short Report. In *Informal Proceedings of the 2nd International Workshop on OWL Reasoner Evaluation (ORE-2013), Ulm, Germany, July 22, 2013*, pages 1–18, 2013.
14. A. V. Grigorev and A. G. Ivashko. TReasoner: System Description. In *Informal Proceedings of the 2nd International Workshop on OWL Reasoner Evaluation (ORE-2013), Ulm, Germany, July 22, 2013*, pages 26–31, 2013.
15. V. Haarslev, K. Hidde, R. Mller, and M. Wessel. The RacerPro knowledge representation and reasoning system. *Semantic Web*, 3(3):267–277, 2012.
16. M. Horridge and S. Bechhofer. The OWL API: A Java API for OWL ontologies. *Semantic Web*, 2(1):11–21, 2011.
17. Y. Kazakov, M. Krtzsch, and F. Simancik. The Incredible ELK - From Polynomial Procedures to Efficient Reasoning with EL Ontologies. *J. Autom. Reasoning*, 53(1):1–61, 2014.
18. R. Kontchakov, M. Rezk, M. Rodriguez-Muro, G. Xiao, and M. Zakharyaschev. Answering SPARQL Queries over Databases under OWL 2 QL Entailment Regime. In *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*, pages 552–567, 2014.
19. C. J. Matheus, K. Baclawski, and M. M. Kokar. BaseVISor: A Triples-Based Inference Engine Outfitted to Process RuleML and R-Entailment Rules. In *Rules and Rule Markup Languages for the Semantic Web, Second International Conference, RuleML 2006, Athens, Georgia, USA, November 10-11, 2006, Proceedings*, pages 67–74, 2006.
20. J. Mendez. jcel: A Modular Rule-based Reasoner. In *Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE-2012), Manchester, UK, July 1st, 2012*, 2012.
21. A. Metke-Jimenez and M. Lawley. Snorocket 2.0: Concrete Domains and Concurrent Classification. In *Informal Proceedings of the 2nd International Workshop on OWL Reasoner Evaluation (ORE-2013), Ulm, Germany, July 22, 2013*, pages 32–38, 2013.
22. R. B. Mishra and S. Kumar. Semantic web reasoners and languages. *Artif. Intell. Rev.*, 35(4):339–368, 2011.
23. B. Motik, Y. Nenov, R. Piro, I. Horrocks, and D. Olteanu. Parallel Materialisation of Datalog Programs in Centralised, Main-Memory RDF Systems. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Qubec City, Qubec, Canada.*, pages 129–137, 2014.
24. R. Mutharaju, P. Hitzler, P. Mateti, and F. Lcu. Distributed and Scalable OWL EL Reasoning. In *Proceedings of the 12th Extended Semantic Web Conference, Portoroz, Slovenia*, To Appear, 2015.
25. M. Niepert, J. Noessner, and H. Stuckenschmidt. Log-Linear Description Logics. In *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pages 2153–2158, 2011.
26. I. Palmisano. JFact repository, 2015.
27. F. Riguzzi, E. Bellodi, E. Lamma, and R. Zese. BUNDLE: A Reasoner for Probabilistic Ontologies. In *Web Reasoning and Rule Systems - 7th International Conference, RR 2013, Mannheim, Germany, July 27-29, 2013. Proceedings*, pages 183–197, 2013.

28. A. A. Romero, B. C. Grau, and I. Horrocks. MORe: Modular Combination of OWL Reasoners for Ontology Classification. In *The Semantic Web - ISWC 2012 - 11th International Semantic Web Conference, Boston, MA, USA, November 11-15, 2012, Proceedings, Part I*, pages 1–16, 2012.
29. U. Sattler and N. Matentzoglu. List of Reasoners (owl.cs). Modified: 01/09/2014.
30. J. Schoenfisch and J. Ortmann. Yarr!: Yet another rewriting reasoner. In *Informal Proceedings of the 2nd International Workshop on OWL Reasoner Evaluation (ORE-2013), Ulm, Germany, July 22, 2013*, pages 19–25, 2013.
31. B. Sertkaya. The ELepHant Reasoner System Description. In *Informal Proceedings of the 2nd International Workshop on OWL Reasoner Evaluation (ORE-2013), Ulm, Germany, July 22, 2013*, pages 87–93, 2013.
32. E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *J. Web Sem.*, 5(2):51–53, 2007.
33. W. Song, B. Spencer, and W. Du. WSReasoner: A Prototype Hybrid Reasoner for ALCHOI Ontology Classification using a Weakening and Strengthening Approach. In *Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE-2012), Manchester, UK, July 1st, 2012*, 2012.
34. A. Steigmiller, T. Liebig, and B. Glimm. Konclude: System description. *J. Web Sem.*, 27:78–85, 2014.
35. E. Thomas, J. Z. Pan, and Y. Ren. TrOWL: Tractable OWL 2 Reasoning Infrastructure. In *The Semantic Web: Research and Applications, 7th Extended Semantic Web Conference, ESWC 2010, Heraklion, Crete, Greece, May 30 - June 3, 2010, Proceedings, Part II*, pages 431–435, 2010.
36. D. Tsarkov and I. Horrocks. FaCT++ Description Logic Reasoner: System Description. In *Automated Reasoning, Third International Joint Conference, IJCAR 2006, Seattle, WA, USA, August 17-20, 2006, Proceedings*, pages 292–297, 2006.
37. D. Tsarkov and I. Palmisano. Chainsaw: a Metareasoner for Large Ontologies. In *Proceedings of the 1st International Workshop on OWL Reasoner Evaluation (ORE-2012), Manchester, UK, July 1st, 2012*, 2012.
38. D. Tsatsou, S. Dasiopoulou, I. Kompatsiaris, and V. Mezaris. LiFR: A Lightweight Fuzzy DL Reasoner. In *The Semantic Web: ESWC 2014 Satellite Events - ESWC 2014 Satellite Events, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 263–267, 2014.
39. W3C. OWL/Implementations. Modified: 11 December 2013.
40. K. Wu and V. Haarslev. A Parallel Reasoner for the Description Logic ALC. In *Proceedings of the 2012 International Workshop on Description Logics, DL-2012, Rome, Italy, June 7-10, 2012*, 2012.
41. G. Xiao and T. Eiter. Inline Evaluation of Hybrid Knowledge Bases. In *Web Reasoning and Rule Systems - 5th International Conference, RR 2011, Galway, Ireland, August 29-30, 2011. Proceedings*, pages 300–305, 2011.
42. R. Zese, E. Bellodi, E. Lamma, and F. Riguzzi. A Description Logics Tableau Reasoner in Prolog. In *Proceedings of the 28th Italian Conference on Computational Logic, Catania, Italy, September 25-27, 2013.*, pages 33–47, 2013.
43. R. Zese, E. Bellodi, E. Lamma, F. Riguzzi, and F. Aguiari. Semantics and Inference for Probabilistic Description Logics. In *Uncertainty Reasoning for the Semantic Web III - ISWC International Workshops, URSW 2011-2013, Revised Selected Papers*, pages 79–99, 2014.

Table 1. Overview of the participating reasoning systems. SC stands for soundness/completeness, P for profile, O1 for OWL 1 and O2 for OWL 2. CALC is the main underlying calculus, EXP the highest expressive language supported. ACT indicates whether there is active development (B=Bugfixes, D=Active development, N=No development)

Name	Institution	SC	ACT	CALC	EXP
BaseVISor[19]	VIStology, Inc.	P	B	Rete Network	NA
BUNDLE[27]	Univ. of Ferrara	O2	D	Tableaux	SROIQ
CEL[2]	Technische Universitt Dresden	P	N	Consequence-based	EL+
Chainsaw[37]	Univ. of Manchester	O2	D	Modular Reasoner	SROIQ
Clipper[9]	Vienna Univ. of Technology	P	B	Query Rewriting	Horn-SHIQ
DBOWL[11]	Univ. of Malaga	O1	D	Relational Algebra and fixed-point iterations	SHOIN
DeLorean[4]	Not given	O2	D	Fuzzy	NA
DistEL[24]	Wright State Univ.	P	D	Consequence-based	NA
DRAOn[8]	Univ. of Paris 8, IUT of Montreuil	O1	D	Compressed models	NA
DReW[41]	Vienna Univ. of Technology	P	B	Datalog Rewriting	EL++
ELepHant[31]	Not given	I	D	Consequence-based	EL++
ELK[17]	Univ. of Ulm, Germany	I	D	Consequence-based	EL+
ELOG[25]	Not given	P	B	Integer Linear Programming	NA
FaCT++[36]	Univ. of Manchester	O2	D	Tableaux	SROIQ
fuzzyDL[5]	ISTI - CNR	I	D	Tableaux	SHIF
HermiT[12]	Univ. of Oxford	O2	N	Hypertableaux	SROIQ
jcel[20]	Technische Universitt Dresden	P	B	Consequence-based	EL+
JFact[26]	Univ. of Manchester	I	D	Tableaux	SROIQ
Konclude[34]	Univ. of Ulm, derivo GmbH	O2	D	Hybrid	SROIQV
LiFR[38]	Centre for Research and Technology Hellas (CERTH)	P	D	Hypertableaux	OWL DLP
Mastro[6]	Sapienza Univ. of Rome	P	D	Query Rewriting	DL-LiteA
MORe[28]	Univ. of Oxford	O2	D	Modular Reasoner	SROIQ
ontop[18]	Free Univ. of Bozen-Bolzano	P	D	Query Rewriting	OWL 2 QL
Pellet[32]	Clark & Parsia, LLC	O2	D	Tableaux	SROIQ
Racer[15]	Concordia Univ., Montreal, Canada; Univ. of Luebeck, Germany;	P	B	Tableaux	SRIQ
RDFox[23]	Univ. of Oxford	P	D	Datalog Rewriting	OWL 2 RL
RuQAR[3]	Poznan Univ. of Technology	P	D	Datalog Rewriting	
Snorocket[21]	CSIRO	I	D	Consequence-based	EL++
TReasoner[14]	Tyumen State Univ.	O2	B	Tableaux	SROIQ
TRILL[42]	Univ. of Ferrara	O1	D	Tableaux	SHOQ
TRILLP[43]	Univ. of Ferrara	O1	D	Tableaux	ALC
TrOWL[35]	Univ. of Aberdeen	P	D	Consequence-based	SROIQ
WSClassifier[33]	Univ. of New Brunswick, Canada	I	B	Hybrid	ALCHOI