Semantics $\sqcap$ Scalability $\models \bot$?

Ian Horrocks

Oxford University Department of Computer Science
Oxford, UK

Abstract. So called "Semantic Technologies" are rapidly becoming mainstream technologies, with RDF and OWL now being deployed in diverse application domains, and with major technology vendors starting to augment their existing systems accordingly. This is, however, only the first step for Semantic Web research; we need to demonstrate that the Semantic Technologies we are developing can (be made to) exhibit robust scalability if deployments in large scale applications are to be successful. In this paper I will briefly review the evolution of Semantic Technologies to date, examine the scalability challenges arising from deployment in large scale applications, and discuss ongoing research aimed at addressing them.

1 Introduction

The Web Ontology Language (OWL) [11, 18] has been developed and standardised by the World Wide Web Consortium (W3C). It is one of the key technologies underpinning the Semantic Web, but its success has now spread far beyond the Web: it has become the ontology language of choice for applications in fields as diverse as biology [23], medicine [7], geography [8], astronomy [6], agriculture [25], and defence [14]. Moreover, ontologies are increasingly being used for "semantic data management", and DB technology vendors have already started to augment their existing software with ontological reasoning. For example, Oracle Inc. has recently enhanced its well-known database management system with modules that use ontologies to support 'semantic data management'. Their product brochure[1] lists numerous application areas that can benefit from this technology, including Enterprise Information Integration, Knowledge Mining, Finance, Compliance Management and Life Science Research.

The standardisation of OWL has brought with it many benefits. In the first place, OWL's basis in description logic has made it possible to exploit the results of more than twenty-five years of research and to directly transfer theoretical results and technologies to OWL. As a consequence, algorithms for computing OWL entailment are well known [17, 26, 10, 24], and the formal properties of the problem are well understood: it is known to be decidable, but to have high complexity (NExpTime-complete for OWL and 2NExpTime-complete for OWL 2 [19]).

[1] `http://www.oracle.com/technology/tech/semantic_technologies/pdf/oracle%20db%20semantics%20overview%2020080722.pdf`

Ontology	SNOMED CT	GALEN	FMA	GO
Logic	$\mathcal{EL}$	$\mathcal{EL}$	$\mathcal{EL}$	$\mathcal{EL}$
#classes	315,489	23,136	78,977	19,468
#properties	58	950	7	1
#axioms	430,844	36,547	121,712	28,897
#subsumptions	$> 10^{11}$	$> 10^{8}$	$> 10^{9}$	$> 10^{8}$
ELK (1 worker)	13.15	1.33	0.44	0.20
ELK (4 workers)	5.02	0.77	0.39	0.19

ontology:	Plant Anat.	SWEET-P	NCI-2	DOLCE-P
Logic	$\mathcal{SHIF}$	$\mathcal{SHOIN}$	$\mathcal{ALCH}$	$\mathcal{SHOIN}$
#classes	19,145	1,728	70,576	118
#properties	82	145	189	264
#axioms	35,770	2,419	100,304	265
#subsumptions	$> 10^{8}$	$> 10^{6}$	$> 10^{9}$	$> 10^{4}$
HermiT	11.2	11.2	—	105.1
Pellet	87.2	—	172.0	105.1
FaCT++	22.9	0.2	60.7	—

Fig. 1. Performance of OWL reasoners on large ontologies.

Year	Ontology size	Complete reasoning	Classification time (s)
1995	3,000	No	10^{5}
1998	3,000	Yes	300
2005	30,000	Yes	30
2010	400,000	Yes	5

Fig. 2. Evolution of reasoner performance over time.

Entailment is a very general reasoning task to which many other kinds of reasoning can be reduced. It is common to distinguish two general categories of reasoning tasks: those that are concerned primarily with classes, and those that are concerned primarily with individuals. Common tasks in the first category include checking class *(un-)satisfiability* (a class C is unsatisfiable w.r.t. an ontology $\mathcal{O}$ iff $\mathcal{O} \models C \equiv \bot$), *subsumption* (a class C subsumes a class D w.r.t. $\mathcal{O}$ iff $\mathcal{O} \models D \sqsubseteq C$), and *classification* (computing the subclass quasi-order for all the class names occurring in an ontology $\mathcal{O}$) [2]. The most common task in the second category is *query answering* (given an ontology $\mathcal{O}$ and a query q, compute the set of tuples of individuals $\{\overline{a} \mid \mathcal{O} \models q(\overline{a})\}$) [12].

2 OWL Reasoning

Notwithstanding the high worst case complexity of the underlying problem, highly optimised reasoners for class reasoning in OWL, including, e.g., ELK [13], FaCT++ [26], HermiT [17], Pellet [24] and Racer [9], are now extremely effective in practice (see, Figure 1 and Figure 2). There has also been significant progress on improving the performance of such systems w.r.t. query answering, but inherent limitations of the underlying algorithmic approach and the typical size of data sets means that many realistic problems that are still out of reach for such systems.

A range of different algorithmic approaches have been developed in an effort to address this issue, each with its own strengths and weaknesses. We will briefly review the most prominent approaches.

Query Rewriting

In an effort to improve scalability, systems such as Mastro[4], Quonto[2] and Requiem[3] employ a *query rewriting* approach in which the ontology is used to rewrite the query into a union of conjunctive queries (UCQ) whose evaluation can be delegated to a standard database system [3]. Unfortunately, the ontology language must be quite restricted in order to guarantee that such a rewriting exists [1], and such systems can support only the OWL 2 QL subset of OWL [19]. Using OWL 2 QL it is possible, for example, to model class hierarchies and (some) incomplete information, but not disjunctive information, relationship cardinalities or property chaining. Moreover, even with very restricted ontology languages, query rewriting algorithms can in theory produce very large rewritings, which cannot be (efficiently) evaluated by standard database systems [21], although optimised systems have been shown to work well in practice for OWL 2 QL ontologies [22].

Materialisation

RDF triple stores such as Sesame[4], OWLim[5], Minerva[6], WebPIE[7] and Oracle's Semantic Data Store[8] use a *materialisation* approach in order to improve the scalability of query answering. These systems are based on a relatively loose integration of ontology and database technologies, with the ontology being used in a preprocessing phase to materialise implied facts, after which queries are evaluated over the augmented data and without further reference to the ontology.

Materialisation based systems are now in quite widespread use, and have even been developed to run on mobile platforms such as the Samsung Galaxy II smartphone [16]. However, although the materialisation technique allows for efficient query evaluation, it also suffers from several drawbacks. Perhaps most important of these is that it can fully support only the OWL 2 RL subset of

[2] `www.dis.uniroma1.it/~quonto`
[3] `www.comlab.ox.ac.uk/projects/requiem`
[4] `www.openrdf.org`
[5] `www.ontotext.com/owlim`
[6] `www.alphaworks.ibm.com/tech/semanticstk`
[7] `www.few.vu.nl/~jui200/webpie.html`
[8] `www.oracle.com/technology/tech/semantic_technologies`

OWL [19]. Another drawback is that the augmented data set may be unfeasibly large, and even when it is of "reasonable" size, it can be very costly to compute. For example, one test data set required 90 hours to load into Oracle's Semantic Data Store (see `www.oracle.com/technology/tech/semantic_technologies/pdf/semtech09.pdf`), and although recent work on parallelisation has shown promising results, materialisation of large datasets can still take several hours even on large clusters [27]. Finally, the approach is applicable only if the information system fully controls and can modify the data, ruling it out in many practical application settings.

Combined Techniques

More recently, *combined techniques* have been developed in order to deal with ontology languages that are (slightly) more expressive than those used in pure query rewriting techniques while still allowing query evaluation to be delegated to a data repository [15]. Like triple stores, combined techniques use the ontology in a preprocessing phase, but they add only the information that is necessary to allow for subsequent query answering via query rewriting. However, the subset of OWL that can be fully supported is still relatively weak (OWL 2 EL), and excludes important features such as disjunctive information and relationship cardinalities. Moreover, the approach still suffers from some of the other drawbacks of materialisation techniques: preprocessing may be costly, and the information system must be able to modify the data.

3 Mitigating Incompleteness

As discussed above, existing materialisation based procedures can fully handle only a relatively week subset of OWL, but they can still be used in a sound but possibly incomplete way with an ontology that is outside the relevant subset. In such cases the set of forward chaining inference rules can no longer guarantee to materialise all implied facts, a problem that is inherent to this technique as the materialisation needs to be both deterministic (it must generate a single data set over which queries can subsequently be evaluated) and of bounded size (or it will not terminate). This means that query answers can be incomplete when the ontology contains, e.g., disjunction or existential quantification. In such cases there is usually no way to know if the answer to a query is complete or not, and if incomplete how incomplete.

In order to address this issue, techniques have recently been investigated for measuring and potentially "repairing" causes of incompleteness. This work has shown that for many ontologies and queries it is possible to identify all relevant causes of incompleteness, and that in many realistic cases it is possible to prove that answers to specific queries are complete even if the query answering system being used is incomplete in general [5].

As far as ontology repair is concerned, one simple way to eliminate some causes of incompleteness is to use a complete reasoner to add implicit subsumptions to the ontology, an idea that has already been used in the DLDB [20] and

PelletDB[9] systems. This technique is, however, limited to adding simple atomic subsumption axioms, and more sophisticated techniques might make it possible to repair ontologies by adding more complex axioms [5].

4 Discussion

Reasoning tools are vital for ontology engineering and to support ontology based systems and applications. In the former case, the focus is mainly on class reasoning, and highly effective reasoners are already available. In the latter case, the focus is mainly on query answering, and although great progress has been made, challenges still remain. This is a very active research area, with many different techniques being developed and investigated. Given the size of this research effort, and our ever deepening understanding of both theoretical and practical issues, it is reasonable to expect that the future performance improvements in query answering systems will be even more spectacular those achieved in the past by class reasoning systems.

References

1. A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The DL-Lite family and relatives. *J. of Artificial Intelligence Research*, 36:1–69, 2009.
2. F. Baader, I. Horrocks, and U. Sattler. Description Logics. In F. van Harmelen, V. Lifschitz, and B. Porter, editors, *Handbook of Knowledge Representation*, chapter 3, pages 135–180. Elsevier, 2008.
3. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *J. of Automated Reasoning*, 39(3):385–429, 2007.
4. D. Calvanese, G. De Giacomo, M. Lenzerini, D. Lembo, A. Poggi, and R. Rosati. MASTRO-I: Efficient integration of relational data through DL ontologies. In *Proc. of the 2007 Description Logic Workshop (DL 2007)*, volume 250 of *CEUR (http://ceur-ws.org/)*, 2007.
5. B. Cuenca Grau, B. Motik, G. Stoilos, and I. Horrocks. Completeness guarantees for incomplete ontology reasoners: Theory and practice. *J. of Artificial Intelligence Research*, 2011.
6. S. Derriere, A. Richard, and A. Preite-Martinez. An ontology of astronomical object types for the virtual observatory. *Proc. of Special Session 3 of the 26th meeting of the IAU: Virtual Observatory in Action: New Science, New Technology, and Next Generation Facilities*, 2006.
7. C. Golbreich, S. Zhang, and O. Bodenreider. The foundational model of anatomy in OWL: Experience and perspectives. *J. of Web Semantics*, 4(3):181–195, 2006.
8. J. Goodwin. Experiences of using OWL at the ordnance survey. In *Proc. of the First Int. Workshop on OWL Experiences and Directions (OWLED 2005)*, volume 188 of *CEUR Workshop Proceedings*. CEUR (http://ceur-ws.org/), 2005.
9. V. Haarslev and R. Möller. RACER system description. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2001)*, volume 2083 of *Lecture Notes in Artificial Intelligence*, pages 701–705. Springer, 2001.

[9] http://clarkparsia.com/pelletdb

10. V. Haarslev, R. Möller, and M. Wessel. Querying the Semantic Web with Racer + nRQL. In *Proc. of the KI-2004 Intl. Workshop on Applications of Description Logics (ADL'04)*, 2004.
11. I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *J. of Web Semantics*, 1(1):7–26, 2003.
12. I. Horrocks and S. Tessaris. Querying the semantic web: a formal approach. In I. Horrocks and J. Hendler, editors, *Proc. of the 1st International Semantic Web Conference (ISWC 2002)*, volume 2342 of *Lecture Notes in Computer Science*, pages 177–191. Springer-Verlag, 2002.
13. Y. Kazakov, M. Krötzsch, and F. Simančík. Concurrent classification of EL ontologies. In *Proc. of the 10th International Semantic Web Conference (ISWC 2011)*, volume 7031 of *Lecture Notes in Computer Science*, pages 305–320. Springer, 2011.
14. L. Lacy, G. Aviles, K. Fraser, W. Gerber, A. Mulvehill, and R. Gaskill. Experiences using OWL in military applications. In *Proc. of the First Int. Workshop on OWL Experiences and Directions (OWLED 2005)*, volume 188 of *CEUR Workshop Proceedings*. CEUR (`http://ceur-ws.org/`), 2005.
15. C. Lutz, D. Toman, and F. Wolter. Conjunctive query answering in the description logic el using a relational database system. In *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI 2009)*, pages 2070–2075, 2009.
16. B. Motik, I. Horrocks, and S. Kim. Delta-reasoner: a semantic web reasoner for an intelligent mobile platform. In *Proc. of the Twentyfirst International World Wide Web Conference (WWW 2012)*. ACM, 2012.
17. B. Motik, R. Shearer, and I. Horrocks. Hypertableau reasoning for description logics. *J. of Artificial Intelligence Research*, 36:165–228, 2009.
18. OWL 2 Web Ontology Language Overview. W3C Recommendation, 27 October 2009. Available at `http://www.w3.org/TR/owl2-overview/`.
19. OWL 2 Web Ontology Language Profiles. W3C Recommendation, 27 October 2009. Available at `http://www.w3.org/TR/owl2-profiles/`.
20. Z. Pan and J. Heflin. DLDB: Extending relational databases to support semantic web queries. In *Proc. of ISWC 2003 Workshop on Practical and Scaleable Semantic Web Systems*, pages 109–113, 2003.
21. H. Pérez-Urbina, I. Horrocks, and B. Motik. Efficient query answering for OWL 2. In *Proc. of the 8th International Semantic Web Conference (ISWC 2009)*, volume 5823 of *Lecture Notes in Computer Science*, pages 489–504. Springer, 2009.
22. M. Rodriguez-Muro and D. Calvanese. Dependencies: Making ontology based data access work. In *Proceedings of the 5th Alberto Mendelzon International Workshop on Foundations of Data Management*, volume 749 of *CEUR (http://ceur-ws.org/)*, 2011.
23. A. Sidhu, T. Dillon, E. Chang, and B. S. Sidhu. Protein ontology development using OWL. In *Proc. of the First Int. Workshop on OWL Experiences and Directions (OWLED 2005)*, volume 188 of *CEUR Workshop Proceedings*. CEUR (`http://ceur-ws.org/`), 2005.
24. E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *J. of Web Semantics*, 5(2):51–53, 2007.
25. D. Soergel, B. Lauser, A. Liang, F. Fisseha, J. Keizer, and S. Katz. Reengineering thesauri for new applications: The AGROVOC example. *J. of Digital Information*, 4(4), 2004.
26. D. Tsarkov and I. Horrocks. FaCT++ description logic reasoner: System description. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006)*,

volume 4130 of *Lecture Notes in Artificial Intelligence*, pages 292–297. Springer, 2006.

27. J. Urbani, S. Kotoulas, J. Maassen, F. van Harmelen, and H. Bal. WebPIE: A Web-scale parallel inference engine using MapReduce. *J. of Web Semantics*, 10(1), 2012.