Description Logics in Ontology Applications

Ian Horrocks

School of Computer Science, University of Manchester
Oxford Road, Manchester M13 9PL, UK
horrocks@cs.man.ac.uk

Abstract. Description Logics (DLs) are a family of logic based knowledge representation formalisms. Although they have a range of applications (e.g., configuration and information integration), they are perhaps best known as the basis for widely used ontology languages such as OWL (now a W3C recommendation). This decision was motivated by a requirement that key inference problems be decidable, and that it should be possible to provide reasoning services to support ontology design and deployment. Such reasoning services are typically provided by highly optimised implementations of tableaux decision procedures; these have proved to be effective in applications in spite of the high worst case complexity of key inference problems. The increasing use of DL based ontologies in areas such as e-Science and the Semantic Web is, however, already stretching the capabilities of existing DL systems, and brings with it a range of research challenges.

1 Introduction

Description Logics (DLs) are a family of class (concept) based knowledge representation formalisms. They are characterised by the use of various constructors to build complex concepts from simpler ones, an emphasis on the decidability of key reasoning tasks, and by the provision of sound, complete and (empirically) tractable reasoning services.

Although they have a range of applications (e.g., reasoning with database schemas and queries [1–3]), DLs are perhaps best known as the basis for ontology languages such as OIL, DAML+OIL and OWL [4]. The decision to base these languages on DLs was motivated by a requirement not only that key inference problems (such as class satisfiability and subsumption) be decidable, but that "practical" decision procedures and "efficient" implemented systems also be available.

That DLs were able to meet the above requirements was the result of extensive research within the DL community over the course of the preceding 20 years or more. This research mapped out a complex landscape of languages, exploring a range of different language constructors, studying the effects of various combinations of these constructors on decidability and worst case complexity, and devising decision procedures, the latter often being tableaux based algorithms. At the same time, work on implementation and optimisation techniques demonstrated that, in spite of the high worst case complexity of key inference problems

(usually at least ExpTime), highly optimised DL systems were capable of providing practical reasoning support in the typical cases encountered in realistic applications.

With the added impetus provided by the OWL standardisation effort, DL systems are now being used to provide computational services for a rapidly expanding range of ontology tools and applications [5–10]. The increasing use of DL based ontologies in areas such as e-Science and the Semantic Web is, however, already stretching the capabilities of existing DL systems, and brings with it a range of research challenges.

2 Ontologies and Ontology Reasoning

In Computer Science, an ontology is usually taken to mean a conceptual model (of some domain), typically realised as a hierarchical vocabulary of terms, together with formal specifications of the meaning of each term. These specifications are often given with reference to other (simpler) terms in the ontology. For example, in a medical terminology ontology, the meaning of the term *Gastritis* might be specified as an *InflammatoryProcess* whose *outcome* is *InflammationOfStomach*, where *InflammatoryProcess*, *outcome* and *InflammationOfStomach* are all terms from the ontology. Such vocabularies may be used, e.g., to facilitate data sharing and reuse (often by annotating data using terms from a shared ontology), to structure data, or simply to explicate and investigate knowledge of a domain.

Ontologies play a major role in the Semantic Web (where they are used to annotate web resources) [11, 12], and are widely used in, e.g., knowledge management systems, e-Science, and bio-informatics and medical terminologies [13–16]. They are also of increasing importance in the Grid, where they may be used, e.g., to support the discovery, execution and monitoring of Grid services [17–19].

Given the formal and compositional nature of ontologies, it is natural to use logics as the basis for ontology languages—this allows for the precise definition of the meaning of compositional operators (such as "and" and "or"), and of relationships between terms (such as "subclass" and "instance"). The effective use of logic based ontology languages in applications will, however, critically depend on the provision of efficient reasoning support. On the one hand, such support is required by ontology engineers in order to help them to design and maintain sound, well-balanced ontologies [20]. On the other hand, such support is required by applications in order to exploit the formal specification of meaning captured in ontologies: querying ontologies and ontology structured data, is equivalent to computing logical entailments [21].

3 Ontology Languages and Description Logics

The OWL recommendation actually consists of three languages of increasing expressive power: OWL Lite, OWL DL and OWL Full. Like OWL's predecessor DAML+OIL, OWL Lite and OWL DL are basically very expressive description

logics with an RDF syntax. OWL Full provides a more complete integration with RDF, but its formal properties are less well understood, and key inference problems would certainly be *much* harder to compute.[1] For these reasons, OWL Full will not be considered here.

More precisely, OWL DL is based on the $\mathcal{SHOIQ}$ DL [23]; it restricts the form of number restrictions to be unqualified (see [24]), and adds a simple form of Datatypes (often called concrete domains in DLs [25]). Following the usual DL naming conventions, the resulting logic is called $\mathcal{SHOIN}(\mathbf{D})$, with the different letters in the name standing for (sets of) constructors available in the language: $\mathcal{S}$ stands for the basic $\mathcal{ALC}$ DL (equivalent to the propositional modal logic $\mathbf{K}_{(\mathbf{m})}$) extended with transitive roles [22], $\mathcal{H}$ stands for role hierarchies (equivalently, inclusion axioms between roles), $\mathcal{O}$ stands for nominals (classes whose extension is a single individual) [26], $\mathcal{N}$ stands for unqualified number restrictions and ($\mathbf{D}$) stands for datatypes) [27]. OWL Lite is equivalent to the slightly simpler $\mathcal{SHIF}(\mathbf{D})$ DL (i.e., $\mathcal{SHOIQ}$ without nominals, and with only functional number restrictions).

These equivalences allow OWL to exploit the considerable existing body of description logic research, e.g.:

- to define the semantics of the language and to understand its formal properties, in particular the decidability and complexity of key inference problems [28];
- as a source of sound and complete algorithms and optimised implementation techniques for deciding key inference problems [29, 22, 27];
- to use implemented DL systems in order to provide (partial) reasoning support [30–32].

3.1 $\mathcal{SHOIN}$ Syntax and Semantics

The syntax and semantics of $\mathcal{SHOIN}$ are briefly introduced here (we will ignore datatypes, as adding a datatype component would complicate the presentation and has little affect on reasoning [33]).

Definition 1. *Let* $\mathbf{R}$ *be a set of* role names *with both transitive and normal role names* $\mathbf{R}_+ \cup \mathbf{R}_\mathsf{P} = \mathbf{R}$*, where* $\mathbf{R}_\mathsf{P} \cap \mathbf{R}_+ = \emptyset$*. The set of* $\mathcal{SHOIN}$-roles *(or* roles *for short) is* $\mathbf{R} \cup \{R^- \mid R \in \mathbf{R}\}$*. A* role inclusion axiom *is of the form* $R \sqsubseteq S$*, for two roles* R *and* S*. A* role hierarchy *is a finite set of role inclusion axioms.*

An interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ *consists of a non-empty set* $\Delta^\mathcal{I}$*, called the* domain *of* $\mathcal{I}$*, and a function* $\cdot^\mathcal{I}$ *which maps every role to a subset of* $\Delta^\mathcal{I} \times \Delta^\mathcal{I}$ *such that, for* $P \in \mathbf{R}$ *and* $R \in \mathbf{R}_+$*,*

$$\langle x, y\rangle \in P^\mathcal{I} \text{ iff } \langle y, x\rangle \in P^{-\mathcal{I}},$$
$$\text{and if } \langle x, y\rangle \in R^\mathcal{I} \text{ and } \langle y, z\rangle \in R^\mathcal{I}, \text{ then } \langle x, z\rangle \in R^\mathcal{I}.$$

[1] Inference in OWL Full is clearly undecidable as OWL Full does not include restrictions on the use of transitive properties which are required in order to maintain decidability [22].

An interpretation $\mathcal{I}$ satisfies a role hierarchy $\mathcal{R}$ *iff* $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$ *for each* $R \sqsubseteq S \in \mathcal{R}$*; such an interpretation is called a* model *of* $\mathcal{R}$*.*

Definition 2. *Let* N_C *be a set of* concept names *with a subset* $N_I \subseteq N_C$ *of* nominals*. The set of* $\mathcal{SHOIN}$-concepts *(or* concepts *for short) is the smallest set such that*

1. *every concept name* $C \in N_C$ *is a concept,*
2. *if* C *and* D *are concepts and* R *is a role, then* $(C \sqcap D)$*,* $(C \sqcup D)$*,* $(\neg C)$*,* $(\forall R.C)$*, and* $(\exists R.C)$ *are also concepts (the last two are called universal and existential restrictions, resp.), and*
3. *if* R *is a* simple *role*[2] *and* $n \in \mathbb{N}$*, then* $\leqslant nR$ *and* $\geqslant nR$ *are also concepts (called atmost and atleast number restrictions).*

The interpretation function $\cdot^{\mathcal{I}}$ *of an interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *maps, additionally, every concept to a subset of* $\Delta^{\mathcal{I}}$ *such that*

$$
\begin{aligned}
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}}, \qquad (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}, \qquad \neg C^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}, \\
&\sharp o^{\mathcal{I}} = 1 \qquad \text{for all } o \in N_I, \\
(\exists R.C)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \text{There is a } y \in \Delta^{\mathcal{I}} \text{ with } \langle x, y\rangle \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}, \\
(\forall R.C)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \text{For all } y \in \Delta^{\mathcal{I}}, \text{ if } \langle x, y\rangle \in R^{\mathcal{I}}, \text{then } y \in C^{\mathcal{I}}\}, \\
\leqslant nR^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \sharp\{y \mid \langle x, y\rangle \in R^{\mathcal{I}}\} \leqslant n\}, \\
\geqslant nR^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \sharp\{y \mid \langle x, y\rangle \in R^{\mathcal{I}}\} \geqslant n\},
\end{aligned}
$$

where, for a set M*, we denote the cardinality of* M *by* $\sharp M$*.*

For C *and* D *(possibly complex) concepts,* $C \dot{\sqsubseteq} D$ *is called a* general concept inclusion *(GCI), and a finite set of GCIs is called a* TBox*.*

An interpretation $\mathcal{I}$ satisfies *a GCI* $C \dot{\sqsubseteq} D$ *if* $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$*, and* $\mathcal{I}$ *satisfies a TBox* $\mathcal{T}$ *if* $\mathcal{I}$ *satisfies each GCI in* $\mathcal{T}$*; such an interpretation is called a* model of $\mathcal{T}$*.*

A concept C *is called* satisfiable with respect to a role hierarchy $\mathcal{R}$ *and a TBox* $\mathcal{T}$ *if there is a model* $\mathcal{I}$ *of* $\mathcal{R}$ *and* $\mathcal{T}$ *with* $C^{\mathcal{I}} \neq \emptyset$*. Such an interpretation is called a* model of C *w.r.t.* $\mathcal{R}$ *and* $\mathcal{T}$*. A concept* D subsumes *a concept* C *w.r.t.* $\mathcal{R}$ *and* $\mathcal{T}$ *(written* $C \sqsubseteq_{\mathcal{R},\mathcal{T}} D$*) if* $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ *holds in every model* $\mathcal{I}$ *of* $\mathcal{R}$ *and* $\mathcal{T}$*. Two concepts* C, D *are* equivalent *w.r.t.* $\mathcal{R}$ *and* $\mathcal{T}$ *(written* $C \equiv_{\mathcal{R},\mathcal{T}} D$*) iff they are mutually subsuming w.r.t.* $\mathcal{R}$ *and* $\mathcal{T}$*. (When* $\mathcal{R}$ *and* $\mathcal{T}$ *are obvious from the context, we will often write* $C \sqsubseteq D$ *and* $C \equiv D$*.) For an interpretation* $\mathcal{I}$*, an individual* $x \in \Delta^{\mathcal{I}}$ *is called an* instance *of a concept* C *iff* $x \in C^{\mathcal{I}}$*.*

Note that, as usual, subsumption and satisfiability can be reduced to each other, and reasoning w.r.t. *general TBoxes* and role hierarchies can be reduced to reasoning w.r.t. role hierarchies only [22, 27].

[2] A role is simple if it is neither transitive nor has any transitive subroles. Restricting number restrictions to simple roles is required in order to yield a decidable logic [22].

3.2 Practical Reasoning Services

Most modern DL systems use *tableaux* algorithms to test concept satisfiability. These algorithms work by trying to construct (a tree representation of) a model of the concept, starting from an individual instance. Tableaux expansion rules decompose concept expressions, add new individuals (e.g., as required by $\exists R.C$ terms),[3] and merge existing individuals (e.g., as required by $\leqslant nR.C$ terms). Nondeterminism (e.g., resulting from the expansion of disjunctions) is dealt with by searching the various possible models. For an unsatisfiable concept, all possible expansions will lead to the discovery of an obvious contradiction known as a *clash* (e.g., an individual that must be an instance of both A and $\neg A$ for some concept A); for a satisfiable concept, a complete and clash-free model will be constructed [34].

Tableaux algorithms have many advantages. It is relatively easy to design provably sound, complete and terminating algorithms, and the basic technique can be extended to deal with a wide range of class and role constructors. Moreover, although many algorithms have a higher worst case complexity than that of the underlying problem, they are usually quite efficient at solving the relatively easy problems that are typical of realistic applications.

Even in realistic applications, however, problems can occur that are much too hard to be solved by naive implementations of theoretical algorithms. Modern DL systems, therefore, include a wide range of optimisation techniques, the use of which has been shown to improve typical case performance by several orders of magnitude [29, 35, 36, 32, 37, 38]. Key techniques include lazy unfolding, absorption and dependency directed backtracking.

Lazy Unfolding In an ontology, or DL Tbox, large and complex concepts are seldom described monolithically, but are built up from a hierarchy of named concepts whose descriptions are less complex. The tableaux algorithm can take advantage of this structure by trying to find contradictions between concept names before adding expressions derived from Tbox axioms. This strategy is known as *lazy unfolding* [29, 36].

The benefits of lazy unfolding can be maximised by lexically *normalising* and *naming* all concept expressions and, recursively, their sub-expressions. An expression C is normalised by rewriting it in a standard form (e.g., disjunctions are rewritten as negated conjunctions); it is named by substituting it with a new concept name A, and adding an axiom $A \equiv C$ to the Tbox. The normalisation step allows lexically equivalent expressions to be recognised and identically named, and can even detect syntactically "obvious" satisfiability and unsatisfiability.

Absorption Not all axioms are amenable to lazy unfolding. In particular, so called *general concept inclusions* (GCIs), axioms of the form $C \sqsubseteq D$ where C is non-atomic, must be dealt with by explicitly making every individual in the

[3] Cycle detection techniques known as *blocking* may be required in order to guarantee termination.

model an instance of $D \sqcup \neg C$. Large numbers of GCIs result in a very high degree of non-determinism and catastrophic performance degradation [36].

Absorption is another rewriting technique that tries to reduce the number of GCIs in the Tbox by absorbing them into axioms of the form $A \sqsubseteq C$, where A is a concept name. The basic idea is that an axiom of the form $A \sqcap D \sqsubseteq D'$ can be rewritten as $A \sqsubseteq D' \sqcup \neg D$ and absorbed into an existing $A \sqsubseteq C$ axiom to give $A \sqsubseteq C \sqcap (D' \sqcup \neg D)$ [39]. Although the disjunction is still present, lazy unfolding ensures that it is only applied to individuals that are already known to be instances of A.

Dependency Directed Backtracking Inherent unsatisfiability concealed in sub-expressions can lead to large amounts of unproductive backtracking search known as thrashing. For example, expanding the expression $(C_1 \sqcup D_1) \sqcap \ldots \sqcap (C_n \sqcup D_n) \sqcap \exists R.(A \sqcap B) \sqcap \forall R.\neg A$ could lead to the fruitless exploration of 2^n possible expansions of $(C_1 \sqcup D_1) \sqcap \ldots \sqcap (C_n \sqcup D_n)$ before the inherent unsatisfiability of $\exists R.(A \sqcap B) \sqcap \forall R.\neg A$ is discovered. This problem is addressed by adapting a form of dependency directed backtracking called *backjumping*, which has been used in solving constraint satisfiability problems [40].

Backjumping works by labelling concepts with a dependency set indicating the non-deterministic expansion choices on which they depend. When a clash is discovered, the dependency sets of the clashing concepts can be used to identify the most recent non-deterministic expansion where an alternative choice might alleviate the cause of the clash. The algorithm can then jump back over intervening non-deterministic expansions *without* exploring any alternative choices. Similar techniques have been used in first order theorem provers, e.g., the "proof condensation" technique employed in the HARP theorem prover [41].

4 Research Challenges for Ontology Reasoning

The development of the OWL language, and the successful use of reasoning systems in tools such as the Protégé editor [42], has demonstrated the utility of logic and automated reasoning in the ontology domain. The increasing use of DL based ontologies in areas such as e-Science and the Semantic Web is, however, already stretching the capabilities of existing DL systems, and brings with it a range of challenges for future research.

Scalability Practical ontologies may be very large—tens or even hundreds of thousands of classes. Dealing with large-scale ontologies already presents a challenge to the current generation of DL reasoners, in spite of the fact that many existing large-scale ontologies are relatively simple. In the 40,000 concept Gene Ontology (GO), for example, much of the semantics is currently encoded in class names such as "heparin-metabolism"; enriching GO with more complex definitions, e.g., by explicitly modelling the fact that heparin-metabolism is a kind of "metabolism" that "acts-on" the carbohydrate "heparin", would make the semantics more accessible, and would greatly increase the value of GO by enabling

new kinds of query such as "what biological processes act on glycosaminoglycan" (heparin is a kind of glycosaminoglycan) [43]. However, adding more complex class definitions can cause the performance of existing reasoners to degrade to the point where it is no longer acceptable to users. Similar problems have been encountered with large medical terminology ontologies, such as the GALEN ontology [44].

Moreover, as well as using a conceptual model of the domain, many applications will also need to deal with very large volumes of instance data—the Gene Ontology, for example, is used to annotate millions of individuals, and practitioners want to answer queries that refer both to the ontology and to the relationships between these individuals, e.g., "what DNA binding products interact with insulin receptors". Answering this query requires a reasoner not only to identify individuals that are (perhaps only implicitly) instances of DNA binding products and of insulin receptors, but also to identify which pairs of individuals are (perhaps only implicitly) instances of the interactsWith role. For existing ontology languages it is possible to use DL reasoning to answer such queries, but dealing with the large volume of GO annotated gene product data is far beyond the capabilities of existing DL systems [45].

Several different approaches to this problem are already under investigation. One of these involves the use of a hybrid DL-DB architecture in which instance data is stored in a database, and query answering exploits the relatively simple relational structure encountered in typical data sets in order minimise the use of DL reasoning and maximise the use of database operations [46]. Another technique that is under investigation is to use reasoning techniques based on the encoding of $\mathcal{SHIQ}$ ontologies in Datalog [47]. On the one hand, theoretical investigations of this technique have revealed that data complexity (i.e., the complexity of answering queries against a fixed ontology and set of instance data) is significantly lower than the complexity of class consistency reasoning (i.e., NP-complete for $\mathcal{SHIQ}$, and even polynomial-time for a slight restriction of $\mathcal{SHIQ}$) [48]; on the other hand, the technique would allow relatively efficient Datalog engines to be used to store and reason with large volumes of instance data.

Expressive Power OWL is a relatively rich ontology language, but many applications require even greater expressive power than that which is provided by the existing OWL standard. For example, in ontologies describing complex physically structured domains such as biology [43] and medicine [44], it is often important to describe aggregation relationships between structures and their component parts, and to assert that certain properties of the component parts transfer to the structure as a whole (a femur with a fractured shaft is a fractured femur) [49]. The importance of this kind of knowledge can be gauged from the fact that various "work-arounds" have been described for use with ontology languages that cannot express it directly [50].

It may not be possible to satisfy all expressive requirements while staying within a decidable fragment of first order logic. Recent research has, therefore,

studied the use in ontology reasoning of semi-decision procedures such as resolution based theorem provers for full first order logic [51]. There have also been studies of languages that combine a DL with some other logical formalism, often Datalog style rules, with the connection between the two formalisms being restricted so as to maintain decidability [52, 47, 53]

Extended Reasoning Services Finally, in addition to solving problems of class consistency/subsumption and instance checking, explaining how such inferences are derived may be important, e.g., to help an ontology designer to rectify problems identified by reasoning support, or to explain to a user why an application behaved in an unexpected manner.

Work on developing practical explanation systems is at a relatively early stage, with different approaches still being developed and evaluated. One such technique involves exploiting standard reasoning services to identify a small set of axioms that still support the inference in question, the hope being that presenting a much smaller (than the complete ontology) set of axioms to the user will help them to understand the "cause" of the inference [54]. Another (possibly complementary) technique involves explaining the steps by which the inference was derived, e.g., using a sequence of simple natural deduction style inferences [55, 56].

As well as explanation, so-called "non-standard inferences" could also be important in supporting ontology design; these include matching, approximation, and difference computations. Non-standard inferences are the subject of ongoing research [57–60]; it is still not clear if they can be extended to deal with logics as expressive as those that underpin modern ontology languages, or if they will scale to large applications ontologies.

5 Summary

Description Logics are a family of class based knowledge representation formalisms characterised by the use of various constructors to build complex classes from simpler ones, and by an emphasis on the provision of sound, complete and (empirically) tractable reasoning services. They have been used in a wide range of applications, but perhaps most notably (at least in recent times) in providing a formal basis and reasoning services for (web) ontology languages such as OWL.

The effective use of logic based ontology languages in applications will, however, critically depend on the provision of efficient reasoning services to support both ontology design and deployment. The increasing use of DL based ontologies in areas such as e-Science and the Semantic Web is, however, already stretching the capabilities of existing DL systems, and brings with it a range of challenges for future research. The extended ontology languages needed in some applications may demand the use of more expressive DLs, and even for existing languages, providing efficient reasoning services is extremely challenging.

Some applications may even call for ontology languages based on larger fragments of FOL. The development of such languages, and reasoning services to

support them, extends these challenges to the whole logic based Knowledge Representation community.

Acknowledgements

I would like to acknowledge the contribution of those who provided me with inspiration and guidance, and the many collaborators with whom I have been privileged to work. These include Franz Baader, Sean Bechhofer, Dieter Fensel, Carole Goble, Frank van Harmelen, Carsten Lutz, Alan Rector, Ulrike Sattler, Peter F. Patel-Schneider, Stephan Tobies and Andrei Voronkov.

References

1. Calvanese, D., De Giacomo, G., Lenzerini, M., Nardi, D., Rosati, R.: Description logic framework for information integration. In: Proc. of the 6th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'98). (1998) 2–13
2. Calvanese, D., De Giacomo, G., Lenzerini, M.: On the decidability of query containment under constraints. In: Proc. of the 17th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS'98). (1998) 149–158
3. Horrocks, I., Tessaris, S., Sattler, U., Tobies, S.: How to decide query containment under constraints using a description logic. In: Proc. of the 7th Int. Workshop on Knowledge Representation meets Databases (KRDB 2000), CEUR (http://ceur-ws.org/) (2000)
4. Horrocks, I., Patel-Schneider, P.F., van Harmelen, F.: From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. J. of Web Semantics **1** (2003) 7–26
5. Knublauch, H., Fergerson, R., Noy, N., Musen, M.: The protégé OWL plugin: An open development environment for semantic web applications. In McIlraith, S.A., Plexousakis, D., van Harmelen, F., eds.: Proc. of the 2004 International Semantic Web Conference (ISWC 2004). Number 3298 in Lecture Notes in Computer Science, Springer (2004) 229–243
6. Liebig, T., Noppens, O.: Ontotrack: Combining browsing and editing with reasoning and explaining for OWL Lite ontologies. In McIlraith, S.A., Plexousakis, D., van Harmelen, F., eds.: Proc. of the 2004 International Semantic Web Conference (ISWC 2004). Number 3298 in Lecture Notes in Computer Science, Springer (2004) 244–258
7. Rector, A.L., Nowlan, W.A., Glowinski, A.: Goals for concept representation in the GALEN project. In: Proc. of the 17th Annual Symposium on Computer Applications in Medical Care (SCAMC'93), Washington DC, USA (1993) 414–418
8. Visser, U., Stuckenschmidt, H., Schuster, G., Vögele, T.: Ontologies for geographic information processing. Computers in Geosciences (to appear)
9. Oberle, D., Sabou, M., Richards, D.: An ontology for semantic middleware: extending daml-s beyond web-services. In: Proceedings of ODBASE 2003. (2003)
10. Wroe, C., Goble, C.A., Roberts, A., Greenwood, M.: A suite of DAML+OIL ontologies to describe bioinformatics web services and data. Int. J. of Cooperative Information Systems (2003) Special Issue on Bioinformatics.
11. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic Web. Scientific American **284** (2001) 34–43

12. The DAML Services Coalition: DAML-S: Web service description for the semantic web. In: Proc. of the 2003 International Semantic Web Conference (ISWC 2003). Number 2870 in Lecture Notes in Computer Science, Springer (2003)
13. Uschold, M., King, M., Moralee, S., Zorgios, Y.: The enterprise ontology. Knowledge Engineering Review **13** (1998)
14. Stevens, R., Goble, C., Horrocks, I., Bechhofer, S.: Building a bioinformatics ontology using OIL. IEEE Transactions on Information Technology in Biomedicine **6** (2002) 135–141
15. Rector, A., Horrocks, I.: Experience building a large, re-usable medical ontology using a description logic with transitivity and concept inclusions. In: Proceedings of the Workshop on Ontological Engineering, AAAI Spring Symposium (AAAI'97), AAAI Press, Menlo Park, California (1997)
16. Spackman, K.: Managing clinical terminology hierarchies using algorithmic calculation of subsumption: Experience with SNOMED-RT. J. of the Amer. Med. Informatics Ass. (2000) Fall Symposium Special Issue.
17. Emmen, A.: The grid needs ontologies—onto-what? (2002) http://www.hoise.com/primeur/03/articles/monthly/AE-PR-02-03-7.html.
18. Tuecke, S., Czajkowski, K., Foster, I., Frey, J., Graham, S., Kesselman, C., Vanderbilt, P.: Grid service specification (draft). GWD-I draft , GGF Open Grid Services Infrastructure Working Group (2002) http://www.globalgridforum.org/.
19. Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The physiology of the grid: An open grid services architecture for distributed systems integration (2002) http://www.globus.org/research/papers/ogsa.pdf.
20. Wolstencroft, K., McEntire, R., Stevens, R., Tabernero, L., Brass, A.: Constructing Ontology-Driven Protein Family Databases. Bioinformatics **21** (2005) 1685–1692
21. Horrocks, I., Tessaris, S.: Querying the semantic web: a formal approach. In Horrocks, I., Hendler, J., eds.: Proc. of the 2002 International Semantic Web Conference (ISWC 2002). Number 2342 in Lecture Notes in Computer Science, Springer-Verlag (2002) 177–191
22. Horrocks, I., Sattler, U., Tobies, S.: Practical reasoning for expressive description logics. In Ganzinger, H., McAllester, D., Voronkov, A., eds.: Proc. of the 6th Int. Conf. on Logic for Programming and Automated Reasoning (LPAR'99). Number 1705 in Lecture Notes in Artificial Intelligence, Springer (1999) 161–180
23. Horrocks, I., Sattler, U.: A tableaux decision procedure for $\mathcal{SHOIQ}$. In: Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005). (2005) To appear.
24. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F., eds.: The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press (2003)
25. Baader, F., Hanschke, P.: A schema for integrating concrete domains into concept languages. In: Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91). (1991) 452–457
26. Blackburn, P., Seligman, J.: Hybrid languages. J. of Logic, Language and Information **4** (1995) 251–272
27. Horrocks, I., Sattler, U.: Ontology reasoning in the $\mathcal{SHOQ}$(D) description logic. In: Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001). (2001) 199–204
28. Donini, F.M., Lenzerini, M., Nardi, D., Nutt, W.: The complexity of concept languages. Information and Computation **134** (1997) 1–58
29. Baader, F., Franconi, E., Hollunder, B., Nebel, B., Profitlich, H.J.: An empirical analysis of optimization techniques for terminological representation systems or:

Making KRIS get a move on. Applied Artificial Intelligence. Special Issue on Knowledge Base Management **4** (1994) 109–132
30. Horrocks, I.: The FaCT system. In de Swart, H., ed.: Proc. of the 2nd Int. Conf. on Analytic Tableaux and Related Methods (TABLEAUX'98). Volume 1397 of Lecture Notes in Artificial Intelligence., Springer (1998) 307–312
31. Patel-Schneider, P.F.: DLP system description. In: Proc. of the 1998 Description Logic Workshop (DL'98), CEUR Electronic Workshop Proceedings, http://ceur-ws.org/Vol-11/ (1998) 87–89
32. Haarslev, V., Möller, R.: RACER system description. In: Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2001). Volume 2083 of Lecture Notes in Artificial Intelligence., Springer (2001) 701–705
33. Pan, J.Z.: Description Logics: Reasoning Support for the Semantic Web. PhD thesis, University of Manchester (2004)
34. Horrocks, I., Sattler, U., Tobies, S.: Practical reasoning for very expressive description logics. J. of the Interest Group in Pure and Applied Logic **8** (2000) 239–264
35. Bresciani, P., Franconi, E., Tessaris, S.: Implementing and testing expressive description logics: Preliminary report. In: Proc. of the 1995 Description Logic Workshop (DL'95). (1995) 131–139
36. Horrocks, I.: Using an expressive description logic: FaCT or fiction? In: Proc. of the 6th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'98). (1998) 636–647
37. Patel-Schneider, P.F.: DLP. In: Proc. of the 1999 Description Logic Workshop (DL'99), CEUR Electronic Workshop Proceedings, http://ceur-ws.org/Vol-22/ (1999) 9–13
38. Horrocks, I., Patel-Schneider, P.F.: Optimizing description logic subsumption. J. of Logic and Computation **9** (1999) 267–293
39. Horrocks, I., Tobies, S.: Reasoning with axioms: Theory and practice. In: Proc. of the 7th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2000). (2000) 285–296
40. Baker, A.B.: Intelligent Backtracking on Constraint Satisfaction Problems: Experimental and Theoretical Results. PhD thesis, University of Oregon (1995)
41. Oppacher, F., Suen, E.: HARP: A tableau-based theorem prover. J. of Automated Reasoning **4** (1988) 69–100
42. Protégé: http://protege.stanford.edu/ (2003)
43. Wroe, C., Stevens, R., Goble, C.A., Ashburner, M.: A methodology to migrate the Gene Ontology to a description logic environment using DAML+OIL. In: Proc. of the 8th Pacific Symposium on Biocomputing (PSB). (2003)
44. Rogers, J.E., Roberts, A., Solomon, W.D., van der Haring, E., Wroe, C.J., Zanstra, P.E., Rector, A.L.: GALEN ten years on: Tasks and supporting tools. In: Proc. of MEDINFO2001. (2001) 256–260
45. Horrocks, I., Li, L., Turi, D., Bechhofer, S.: The instance store: DL reasoning with large numbers of individuals. In: Proc. of the 2004 Description Logic Workshop (DL 2004). (2004) 31–40
46. Bechhofer, S., Horrocks, I., Turi, D.: The OWL instance store: System description. In: Proc. of the 20th Int. Conf. on Automated Deduction (CADE-20). Lecture Notes in Artificial Intelligence, Springer (2005) To appear.
47. Hustadt, U., Motik, B., Sattler, U.: Reducing SHIQ-description logic to disjunctive datalog programs. In: Proc. of the 9th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2004). (2004) 152–162

48. Motik, B., Sattler, U., Studer, R.: Query answering for OWL-DL with rules. In: Proc. of the 2004 International Semantic Web Conference (ISWC 2004). (2004) 549–563
49. Rector, A.: Analysis of propagation along transitive roles: Formalisation of the galen experience with medical ontologies. In: Proc. of DL 2002, CEUR (http://ceur-ws.org/) (2002)
50. Schulz, S., Hahn, U.: Parts, locations, and holes - formal reasoning about anatomical structures. In: Proc. of AIME 2001. Volume 2101 of Lecture Notes in Artificial Intelligence., Springer (2001)
51. Tsarkov, D., Riazanov, A., Bechhofer, S., Horrocks, I.: Using Vampire to reason with OWL. In McIlraith, S.A., Plexousakis, D., van Harmelen, F., eds.: Proc. of the 2004 International Semantic Web Conference (ISWC 2004). Number 3298 in Lecture Notes in Computer Science, Springer (2004) 471–485
52. Eiter, T., Lukasiewicz, T., Schindlauer, R., Tompits, H.: Combining answer set programming with description logics for the semantic web. In: Proc. of the 9th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2004), Morgan Kaufmann, Los Altos (2004) 141–151
53. Rosati, R.: On the decidability and complexity of integrating ontologies and rules. J. of Web Semantics **3** (2005) 61–73
54. Schlobach, S., Cornet, R.: Explanation of terminological reason-ing: A preliminary report. In: Proc. of the 2003 Description Logic Workshop (DL 2003). (2003)
55. McGuinness, D.L.: Explaining Reasoning in Description Logics. PhD thesis, Rutgers, The State University of New Jersey (1996)
56. Borgida, A., Franconi, E., Horrocks, I.: Explaining $\mathcal{ALC}$ subsumption. In: Proc. of the 14th Eur. Conf. on Artificial Intelligence (ECAI 2000). (2000)
57. Baader, F., Küsters, R., Borgida, A., McGuinness, D.L.: Matching in description logics. J. of Logic and Computation **9** (1999) 411–447
58. Brandt, S., Turhan, A.Y.: Using non-standard inferences in description logics — what does it buy me? In: Proc. of KI-2001 Workshop on Applications of Description Logics (KIDLWS'01). Volume 44 of CEUR (http://ceur-ws.org/). (2001)
59. Küsters, R.: Non-Standard Inferences in Description Logics. Volume 2100 of Lecture Notes in Artificial Intelligence. Springer Verlag (2001)
60. Brandt, S., Küsters, R., Turhan, A.Y.: Approximation and difference in description logics. In: Proc. of the 8th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2002). (2002) 203–214