Hybrid Logics and Ontology Languages

Ian Horrocks[1] Birte Glimm[2] Ulrike Sattler[3]

School of Computer Science
University of Manchester
Manchester, UK

Abstract

Description Logics (DLs) are a family of logic based knowledge representation formalisms. Although they have a range of applications, they are perhaps best known as the basis for widely used ontology languages such as OIL, DAML+OIL and OWL, the last of which is now a World Wide Web Consortium (W3C) recommendation. $\mathcal{SHOIN}$, the DL underlying OWL DL (the most widely used "species" of OWL), includes familiar features from hybrid logic. In particular, in order to support extensionally defined classes, $\mathcal{SHOIN}$ includes *nominals*: classes whose extension is a singleton set. This is an important feature for a logic that is designed for use in ontology language applications, because extensionally defined classes are very common in ontologies. Binders and state variables are another feature from Hybrid Logic that would clearly be useful in an ontology language, but it is well known that adding this feature to even a relatively weak language would lead to undecidability. However, recent work has shown that this feature could play a very useful role in query answering, where the syntactic structure of queries means that the occurrence of state variables is restricted in a way that allows for decidable reasoning.

Keywords: Description Logic, Hybrid Logic, Tableaux Reasoning

1 Introduction

Description Logics (DLs) are a family of logic based knowledge representation formalisms. Although they have a range of applications (e.g., configuration [23], and information integration [7]), they are perhaps best known as the basis for widely used ontology languages such as OIL, DAML+OIL and OWL [14], the last of which is now a World Wide Web Consortium (W3C) recommendation [4].

The OWL specification describes three language "species", OWL Lite, OWL DL and OWL Full, two of which (OWL Lite and OWL DL) are based on expressive description logics.[4] The decision to base these languages on DLs was motivated by a requirement that key inference problems (such as ontology consistency) be decidable, and hence that it should be possible to provide reasoning services to support ontology design and deployment [14].

[1] Email: horrocks@cs.man.ac.uk

[2] Email: glimm@cs.man.ac.uk

[3] Email: sattler@cs.man.ac.uk

[4] OWL Full uses the same language vocabulary as OWL DL, but does not restrict its use to "well formed formulae".

OWL Lite and OWL DL are based on the DLs $\mathcal{SHIF}$ and $\mathcal{SHOIN}$ respectively—in fact, OWL Lite is just a syntactic subset of OWL DL [14].[5] $\mathcal{SHOIN}$, the DL underlying OWL DL, includes familiar features from hybrid logic. In particular, in order to support extensionally defined classes, $\mathcal{SHOIN}$ includes *nominals*: classes whose extension is a singleton set. This is an important feature for a logic that is designed for use in ontology language applications, because extensionally defined classes are very common in ontologies. For example, OWL provides a `oneOf` class constructor that allows users to define classes by enumerating their members, e.g., when describing a class such as EUCountries by enumerating its members, i.e., $\{\mathsf{Austria}, \ldots, \mathsf{UnitedKingdom}\}$ (such an enumeration is equivalent to a disjunction of nominals). This allows applications to infer, e.g., that persons who only visit EUCountries can visit at most 25 countries. Singleton classes are also widely used in ontologies, e.g., in the well known OWL Wine ontology (see `http://www.w3.org/TR/owl-guide/wine.rdf`), where they are used for colours, grape types, wine producing regions, etc.

Binders (in particular the $\downarrow$ binder) and state variables are another feature from Hybrid Logic that would clearly be useful in an ontology language, but it is well known that adding this feature to even a relatively weak language would lead to undecidability [5]. Recent work has shown, however, that this feature could play a very useful role in query answering, where the syntactic structure of queries means that the occurrence of state variables is restricted in a way that allows for decidable reasoning.

2 Preliminaries

The correspondence between modal and description logics has long been understood [29]. The basic propositionally closed DL $\mathcal{ALC}$ is equivalent to the propositional modal logic $\mathcal{K}_{(\mathbf{m})}$. The logic $\mathcal{S}$ extends $\mathcal{ALC}$ with transitive roles, and so can be thought of as the union of $\mathcal{K}_{(\mathbf{m})}$ and $\mathbf{K4}_{(\mathbf{m})}$. $\mathcal{SHOIQ}$ extends $\mathcal{S}$ with a hierarchy of roles ($\mathcal{H}$), i.e., the ability to assert implications w.r.t. modalities, nominals ($\mathcal{O}$), inverse roles ($\mathcal{I}$), i.e., converse modalities, and qualified cardinality constraints ($\mathcal{Q}$), i.e., graded modalities. The DL $\mathcal{SHOIN}$ is a restricted version of $\mathcal{SHOIQ}$, which is obtained by allowing only unqualified cardinality constraints, i.e., graded modalities can range over the concept $\top$ only.

We will now briefly introduce the syntax, semantics, and inference problems of the DL $\mathcal{SHOIQ}$; full details can be found in [16].

Definition 2.1 Let $\mathbf{R}$ be a countable 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 *$\mathcal{SHOIQ}$-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 $\Delta^\mathcal{I}$, 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

[5] OWL also includes *datatypes*, a simple form of *concrete domain* [2]. These can, however, be treated exactly as in $\mathcal{SHOQ}(\mathbf{D})/\mathcal{SHOQ}(\mathbf{D}_n)$ [15,11,27], so we will not complicate matters by considering them here.

$P \in \mathbf{R}$ and $R \in \mathbf{R}_+$,

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

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

Definition 2.2 Let N_C be a countable set of *concept names* with a subset $N_I \subseteq N_C$ of *nominals*. The set of $\mathcal{SHOIQ}$-*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 C is a concept, R is a simple role[6] and $n \in \mathbb{N}$, then $(\leqslant nR.C)$ and $(\geqslant nR.C)$ 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

$$\begin{aligned}
(C \sqcap D)^{\mathcal{I}} &= C^{\mathcal{I}} \cap D^{\mathcal{I}}, \quad (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}, \\
\neg C^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}, \qquad\qquad \sharp o^{\mathcal{I}} = 1 \text{ for all } o \in N_I, \\
(\exists R.C)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid R^{\mathcal{I}}(x, C) \neq \emptyset\}, \\
(\forall R.C)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid R^{\mathcal{I}}(x, \neg C) = \emptyset\}, \\
(\leqslant nR.C)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \sharp R^{\mathcal{I}}(x, C) \leqslant n\}, \text{ and} \\
(\geqslant nR.C)^{\mathcal{I}} &= \{x \in \Delta^{\mathcal{I}} \mid \sharp R^{\mathcal{I}}(x, C) \geqslant n\},
\end{aligned}$$

where $\sharp M$ is the cardinality of a set M and $R^{\mathcal{I}}(x, C)$ is defined as $\{y \mid \langle x, y\rangle \in R^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}}\}$. We sometimes use $(=nR.C)$ as an abbreviation for $(\leqslant nR.C) \sqcap (\geqslant nR.C)$.

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*. Let N_A be a countable set of individual names. For $a, b \in N_A$, C a (possibly complex) concept, and R a role, an individual assertion is of the form $C(a)$ or $R(a, b)$, and a finite set of individual assertions is called an *ABox*. For $\mathcal{T}$ a TBox and $\mathcal{A}$ and ABox, a pair $\langle \mathcal{T}, \mathcal{A}\rangle$ is called a Knowledge Base (KB). Individual names can also be seen as a weak form of nominals, and it is well known that, in the presence of nominals, an ABox can be expressed in terms of TBox axioms [28]. For a DL with nominals, we can therefore assume that each individual assertion $C(a)$ is an abbreviation for $a \dot{\sqsubseteq} C$ with $a \in N_I$ and each assertion $R(a, b)$ is an abbreviation for $a \dot{\sqsubseteq} \exists R.b$ with $a, b \in N_I$. Given a nominal o with $o^{\mathcal{I}} = \{x\}$ for some individual $x \in \Delta^{\mathcal{I}}$, we will

[6] A simple role is a role that is neither transitive nor has a transitive sub-role; restricting number restrictions to simple roles is required in order to yield a decidable logic [17].

sometimes abuse our notation by treating o as an *individual name* such that $o^{\mathcal{I}} = x$. Please note that we do not make a *unique name assumption*: two nominals (resp. individual names) might have the same interpretation.

An interpretation $\mathcal{I}$ *satisfies* a GCI $C \dot{\sqsubseteq} D$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, it satisfies an individual assertion $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$, and it satisfies an individual assertion $R(a,b)$ if $\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$. An interpretation $\mathcal{I}$ satisfies a TBox $\mathcal{T}$ (resp. ABox $\mathcal{A}$) if $\mathcal{I}$ satisfies each GCI in $\mathcal{T}$ (resp. each individual assertion in $\mathcal{A}$), and $\mathcal{I}$ satisfies a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ if $\mathcal{I}$ satisfies both $\mathcal{T}$ and $\mathcal{A}$; such an interpretation is called a *model of* $\mathcal{T}$ (resp. $\mathcal{A}$, $\mathcal{K}$).

A concept C is *satisfiable w.r.t. a role hierarchy* $\mathcal{R}$ *and a KB* $\mathcal{K}$ if there is a model $\mathcal{I}$ of $\mathcal{R}$ and $\mathcal{K}$ with $C^{\mathcal{I}} \neq \emptyset$. Such an interpretation is called a *model of* C w.r.t. $\mathcal{R}$ and $\mathcal{K}$. A concept D *subsumes* a concept C w.r.t. $\mathcal{R}$ and $\mathcal{K}$ (written $C \sqsubseteq_{\mathcal{R},\mathcal{K}} D$) if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds in every model $\mathcal{I}$ of $\mathcal{R}$ and $\mathcal{K}$. Two concepts C, D are *equivalent* w.r.t. $\mathcal{R}$ and $\mathcal{K}$ (written $C \equiv_{\mathcal{R},\mathcal{K}} D$) if they are mutually subsuming w.r.t. $\mathcal{R}$ and $\mathcal{K}$.

For an individual name a, we say that a is an instance of a concept C w.r.t. a role hierarchy $\mathcal{R}$ and a KB $\mathcal{K}$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{R}$ and $\mathcal{K}$, and, for $a, b \in \mathbb{N}_A$, the pair (a,b) is an instance of a role R if $\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in R^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{R}$ and $\mathcal{K}$.

3 Reasoning with Nominals

A key motivation for basing OWL DL on Description Logics was in order to exploit both theoretical results and implemented DL reasoning systems. Interestingly, though, at the time of the standardisation of OWL there was no known "practical" algorithm for deciding the satisfiability of a $\mathcal{SHOIQ}$ KB (where by practical, we mean a goal directed procedure that is likely to behave well on typical ontology derived problems [33,15]). As a consequence, no implemented systems were available either. Although, via correspondences with other logics, $\mathcal{SHOIQ}$ was known to be decidable [33,26], it proved difficult to extend existing tableaux decision procedures to deal with $\mathcal{SHOIQ}$. The problem is not caused by nominals alone—decision procedures for $\mathcal{SHOQ}$ (i.e., $\mathcal{SHOIQ}$ minus inverse roles) and $\mathcal{SHOI}$ (i.e., $\mathcal{SHOIQ}$ minus number restrictions) were already known—but by the combination of nominals with inverse roles and number restrictions.

In order to see why this is problematical, it is useful to first consider some of the features of $\mathcal{SHIQ}$ (i.e., $\mathcal{SHOIQ}$ minus nominals), and of the tableaux decision procedure for $\mathcal{SHIQ}$. One reason why DLs (and propositional modal and dynamic logics) enjoy good computational properties, such as being robustly decidable, is that they have some form of tree model property [35,10], i.e., if an ontology is consistent, then it has a model with a tree-like relational structure. This feature is crucial in the design of tableaux algorithms, allowing them to search only for tree-like models.

More precisely, DL tableaux algorithms decide consistency of an ontology by trying to construct an abstraction of a model for it, a so-called "completion graph". In a completion graph, each node x represents one or more individuals, and is labelled with a set of concepts which the individuals represented by x are instances of; each edge $\langle x, y \rangle$ represents one or more pairs of individuals, and is labelled with a set of roles which the pairs of individuals represented by $\langle x, y\rangle$ are instances of. The algorithm works by initialising the graph with one node for each individual name/nominal in the input KB, and using a set of expansion rules to syntactically decompose concepts in node labels; each such rule application can add new concepts to node labels and/or new nodes and edges to the completion graph, thereby explicating the structure of a model. The rules are repeatedly applied until either the graph is fully expanded (no more rules are applicable), in which case the graph can be used to construct a model that is a *witness* to the satisfiability of the input KB, or an obvious contradiction (called a *clash*) is discovered (e.g., both C and $\neg C$ in a node label), proving that the completion graph does not correspond to a model. The input KB is consistent iff the rules (some of which are nondeterministic) can be applied in such a way as to produce a fully expanded and clash free completion graph.

For logics with the tree model property, we can restrict our search/construction to tree-shaped completion graphs. For expressive DLs, this restriction is crucial, since tableaux algorithms for them employ a cycle detection technique called *blocking* in order to guarantee termination. This is of special interest for $\mathcal{SHIQ}$, where the interaction between inverse roles and number restrictions results in the loss of the *finite model property*, i.e., there are consistent ontologies that only admit infinite models. On such an input, the $\mathcal{SHIQ}$ tableaux algorithm generates a finite, tree-shaped completion graph that can be *unravelled* into an infinite tree model, and where a node in the completion graph may stand for infinitely many elements of the model. Even when the language includes nominals, but *excludes* one of number restrictions or inverse roles [15,13], or if only individual names instead of nominals are allowed [18], we can work on forest-shaped completion graphs, with each nominal (individual) being the root of a tree-like section; this causes no inherent difficulty as the size of the non-tree part of the graph is restricted by the number of individual names/nominals in the input.

The difficulty in extending the $\mathcal{SHOQ}$ or $\mathcal{SHIQ}$ algorithms to $\mathcal{SHOIQ}$ is due to the interaction between nominals, number restrictions, and inverse roles, which leads to the *almost* complete loss of the tree model property, and causes the complexity of the ontology consistency problem to jump from ExpTime to NExpTime [32]. To see this interaction, consider an ontology containing the following two axioms involving a nominal o and a non-negative integer n:

$$F \mathrel{\dot{\sqsubseteq}} \exists U^{-}.o$$
$$o \mathrel{\dot{\sqsubseteq}} ({\leqslant} n\, U.F)$$

The first statement requires that, in a model of this ontology, every instance of F has an incoming U-edge from o, while the second statement restricts the number of U-edges going from o to instances of F to at most n. The nominal o thus acts as a so-called "spy point" for instances of the concept F (o can "see" every instance of F), and the number restriction on the inverse of U imposes an upper bound of n on the number of instances of F. If we add further axioms, we might need to consider arbitrarily complex relational structures amongst instances of F. For example, if we add the following axiom, then there must be exactly n instances of F, and each instance of F is necessarily R-related to every instance of F (including itself):

$$F \mathrel{\dot\sqsubseteq} ({=}nR.F).$$

Similarly, the following axiom would enforce S-cycles over instances of F:

$$F \mathrel{\dot\sqsubseteq} ({=}1S.F) \sqcap ({=}1S^{-}.F).$$

Hence a tableaux algorithm for $\mathcal{SHOIQ}$ needs to be able to handle arbitrarily complex relational structures, and thus we cannot restrict our attention to completion trees or forests.

To complicate matters even more, recall that there are even $\mathcal{SHIQ}$ axioms that enforce the existence of an infinite number of instances of a concept. For example, the concept $\neg N \sqcap \exists T.N$ is satisfiable w.r.t. the following axiom, but only in models with infinitely many instances of N:

$$N \mathrel{\dot\sqsubseteq} ({\leqslant}1T^{-}.(A \sqcup \neg A)) \sqcap \exists T.N.$$

Now consider an ontology that contains, amongst others, all the above mentioned axioms. The consistency of this ontology then crucially depends on the relations enforced between instances of F and N. For example, the additional axioms

$$N \mathrel{\dot\sqsubseteq} \exists V.F \quad \text{and}$$
$$F \mathrel{\dot\sqsubseteq} ({\leqslant}kV^{-}.N)$$

yield an inconsistent ontology since our at most n instances of F cannot play the rôle of V-fillers for infinitely many instances of F when each of them can be the V-filler of at most k instances of N.

Summing up, a tableaux algorithm for $\mathcal{SHOIQ}$ needs to be able to handle both arbitrarily complex relational structures and finite tree structures representing infinite trees, and to make sure that all constraints are satisfied—especially number restrictions on relations between these two parts—while still guaranteeing termination.

Two key intuitions have allowed us to devise a tableaux algorithm that meets all of these requirements. The first intuition is that, when extending a $\mathcal{SHOIQ}$ completion graph, we can distinguish those nodes that may be arbitrarily interconnected (so-called *nominal nodes*) from those nodes that still form a tree structure (so-called *blockable nodes*). Moreover, restrictions on the ways in which nominal nodes can be created allow us to fix a (double exponential) upper bound on the number of nominal nodes that can occur in a completion graph. This allows us to restrict blocking, and hence unravelling, to blockable nodes, i.e., blockable nodes may represent an infinite number of individuals, whereas a nominal node represents exactly one individual in the model. This allows us to fix an upper bound on the size of a completion graph, but it is still not enough to guarantee termination, as we may repeatedly create and merge nodes.

For example, given the axiom:

$$C \sqsubseteq (\exists R.C) \sqcap (\forall R^{-}.o),$$

where o is a nominal, constructing a completion graph for C could lead to a sequence of three nodes, say x_0, x_1 and x_2, with $\{r\}$ labelled edges connecting x_0 with x_1 and x_1 with x_2, and with the nodes labelled $\{C, (\exists R.C), (\forall R^{-}.o), o\}$, $\{C, (\exists R.C), (\forall R^{-}.o), o\}$ and $\{C, (\exists R.C), (\forall R^{-}.o)\}$ respectively. The tableaux rule dealing with nominals might then be applied in order to merge x_1 into x_0 (reflecting the semantics of nominals), but applying the rule dealing with existential restrictions to $(\exists R.C)$ in the label of x_2 would lead to the creation of a new $\{R\}$ labelled edge connecting x_2 to a new node, say x_3, which, after some additional expansion, would have the same label as x_2. Applying the rule dealing with value restrictions to $(\forall R^{-}.o)$ in the label of x_3 would cause o to be added to the label of x_2, and allow the above process to be repeated w.r.t. x_0, x_2 and x_3. This (particular version of the) problem was identified by Baader, and called, for obvious reasons, a "yo-yo" [2,3].

This problem is normally addressed by discarding sub-trees below a node whenever it is merged with another node. In the above case, for example, this would result in x_2 (and the edge from x_1 to x_2) being removed from the graph when x_1 is merged with x_0; the resulting graph is fully expanded, and so the algorithm would terminate. This only works, however, in tree-shaped parts of the completion graph, as the notion of a sub-tree is otherwise not well defined. This can lead to problems if nominal nodes are created and merged.

The second and crucial intuition is that this problem can be overcome by "guessing" the *exact* number of new nominal nodes created as the result of interactions between existing nominal nodes, inverse roles, and number restrictions. This guessing is implemented by a new expansion rule, the *NN*-rule. When applied to a relevant $(\leqslant nR.C)$ concept in the label of a nominal node x, the *NN*-rule chooses (non-deterministically) an integer m, such that $1 \leqslant m \leqslant n$, adds $(\leqslant mR.C)$ to the label of x, and generates m new nominal nodes, all of which are pairwise disjoint, and m new $\{R\}$ labelled edges leading from x to the new nominal nodes. As they are pairwise disjoint, none of these nodes can be merged, and adding $(\leqslant mR.C)$ means that no more nominal nodes can be added as a result of concepts of the form $(\leqslant nR.C)$ in the label of x. Termination is now guaranteed by the upper bound on the number of nominal nodes and the use of standard blocking techniques for the blockable nodes. The non-determinism introduced by this rule will clearly be problematical for large values of n, but large values in number restrictions are already known to be problematical for $\mathcal{SHIQ}$. Moreover, the rule has excellent "pay as you go" characteristics: in case number restrictions are functional (i.e., where n is 1), [7] the new rule becomes deterministic; in case there are no interactions between number restrictions, inverse roles, and nominals, the rule will never be applied; in case there are no nominals, the new algorithm behaves like the algorithm for $\mathcal{SHIQ}$; and in case there are no inverse roles the new algorithm behaves like the algorithm

[7] The DL obtained from $\mathcal{SHIQ}$ by allowing only functional number restrictions is called $\mathcal{SHIF}$. There are many realistic $\mathcal{SHIF}$ ontologies; see, e.g., the DAML ontology library at http://www.daml.org/ontologies/

for $\mathcal{SHOQ}$.

Indeed, recent implementations of this algorithm in Fact++ and Pellet [34,30] show promising behaviour: for example, the Wine ontology[8] can now be classified, and both reasoners only take a few seconds for its classification, despite the fact that it is in $\mathcal{SHOIF}$ and contains 206 nominals. Recent work has also shown how a similar technique can be used in order to extend resolution based reasoning procedures to deal with $\mathcal{SHOIQ}$ [21].

4 Answering Conjunctive Queries

Existing DL reasoners (for example FaCT++ [34], KAON2 [20], Pellet [30] and Racer Pro [12]) provide automated reasoning support for checking concepts for satisfiability and subsumption, and also for answering queries that retrieve instances of concepts and roles. In many applications, however, a more powerful query language is required. An obvious approach would be to extend Datalog style conjunctive queries to DLs, and this has led to studies of the problem of conjunctive query answering for DLs [22,19]. As usual, we will restrict our attention to the problem of answering Boolean conjunctive queries, i.e., queries where the answer is either true or false; it is well known how to reduce non-Boolean queries (i.e., queries where the answer is a set of tuples of individuals) to Boolean ones.

Definition 4.1 A *Boolean conjunctive query* q has the form $\langle\rangle \leftarrow conj_1(\vec{y};\vec{c}) \wedge \ldots \wedge conj_n(\vec{y};\vec{c})$, where $\vec{y}$ is a vector of *variables* and $\vec{c}$ is a vector of individual names. We call $\mathbf{T}(q) = \vec{y} \cup \vec{c}$ the set of *terms* in q,[9] and we call each $conj_i(\vec{y};\vec{c})$ for $1 \leq i \leq n$ an atom. *Atoms* are either concept or role atoms: a *concept atom* has the form $C(t)$, and a *role atom* the form $R(t,t')$, for $\{t,t'\} \subseteq \mathbf{T}(q)$, C a possibly complex concept, and R a role.

The semantics of Boolean conjunctive queries is defined in terms of *assignments*. Let $\mathcal{K}$ be an knowledge base, $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ a model for $\mathcal{K}$, and q a Boolean conjunctive query. An *assignment in* $\mathcal{I}$ is a mapping $\cdot^A : \mathbf{T}(q) \rightarrow \Delta^{\mathcal{I}}$. We say that q *is true in* $\mathcal{I}$ and write $\mathcal{I} \models q$ if there exists an assignment $\cdot^A$ in $\mathcal{I}$ s.t. $t^A \in \{t^{\mathcal{I}}\}$ for every individual name $t \in \vec{c}$, $t^A \in C^{\mathcal{I}}$ for every concept atom $C(t)$ in q, and $\langle t^A, t'^A \rangle \in R^{\mathcal{I}}$ for every role atom $R(t,t')$ in q. We call such an assignment *a* q*-mapping w.r.t.* $\mathcal{I}$. If $\mathcal{I} \models \mathcal{K}$ implies $\mathcal{I} \models q$ for all models $\mathcal{I}$ of $\mathcal{K}$, then we say that q *is true in* $\mathcal{K}$, and write $\mathcal{K} \models q$; otherwise we say that q *is false in* $\mathcal{K}$, and write $\mathcal{K} \not\models q$.

The development of a decision procedure for conjunctive query answering in expressive DLs is still a partially open problem. *Grounded* conjunctive queries for $\mathcal{SHIQ}$ and/or $\mathcal{SHOIQ}$ are supported by KAON2, Pellet, and Racer's query language nRQL [24,12]. However, the semantics of grounded queries is different from the usually assumed open-world semantics in DLs since existentially quantified variables are always bound to named individuals.

Various decision procedures [9,25,19] are known for restricted fragments of conjunctive queries. The most common restriction used in order to obtain a decision

[8] Available at `http://www.w3.org/2001/sw/WebOnt/guide-src/wine.rdf`.

[9] For readability, we sometimes abuse our notation and refer to $\vec{y}$ as a set. When referring to a vector $\vec{y}$ as a set, we mean the set $\{y_i \mid y_i \text{ occurs in } \vec{y}\}$.

procedure is that each role that occurs in the query must be a simple role, i.e., a role that is neither transitive nor has a transitive sub-role.

A commonly used technique is to reduce the problem of answering a Boolean conjunctive query to the problem of deciding concept satisfiability w.r.t. a KB. The idea is to view the query as a graph and, starting from leaf nodes, to "roll up" the graph into a single atom of the form $C(t)$ such that the query is true in a KB $\mathcal{K}$ iff $\neg C$ is unsatisfiable w.r.t. $\mathcal{K}$ [9,6,19,31]. For example, the query

$$q = \langle\rangle \leftarrow R(t, t') \wedge S(t', t'') \wedge C(t'')$$

could be rolled up into

$$\langle\rangle \leftarrow (\exists R.(\exists S.C))(t)$$

such that $K \models q$ iff $\neg(\exists R.(\exists S.C))$ is unsatisfiable w.r.t. $\mathcal{K}$.

Unfortunately, the above described technique is only able to deal with tree shaped queries, i.e., those where the query graph does not contain a cycle, because the above mentioned tree model property means that DL concepts cannot capture cyclic relationships. Even nominals cannot express the arbitrary cyclic structures that can occur in a query [31].

Recent work has, however, shown how an extension of DLs with a restricted form of the $\downarrow$ binder operator and state variables can enable an extension of the rolling-up technique to deal also with cyclic queries [9]. It is easy to see how these features can be used to roll up an arbitrary query into a suitable concept. For example, the query

$$q = \langle\rangle \leftarrow R(t, t') \wedge S(t', t)$$

could be rolled up into the concept

$$\langle\rangle \leftarrow (\downarrow t.(\exists R.(\exists S.t)))(t)$$

Although the $\downarrow$ binder already makes the DL $\mathcal{ALC}$ undecidable, when used in query answering the $\downarrow$ binder occurs only in a very restricted form, and the resulting extension of $\mathcal{SHOIQ}$ is still decidable [8] if only simple roles are used in the query. Moreover, it is relatively easy to extend a tableaux algorithm for $\mathcal{SHOIQ}$ to one with this restricted form of the $\downarrow$ binder. However, although it is now known that conjunctive query answering for $\mathcal{SHIQ}$ (without any restrictions) is decidable [1], it is not clear how to extend the $\downarrow$ binder technique to queries in which non-simple roles occur in a cycle. In this case, the binder interacts with blocking in a way that makes termination problematical.

5 Discussion

As we have seen, description logics underlying state of the art ontology languages include nominals, a well known feature of hybrid logic. Nominals are an important feature of ontology languages, as extensionally defined and singleton classes are common in ontologies. Recent advances in DL reasoning techniques have shown how the tableaux algorithm for $\mathcal{SHIQ}$, which is widely used in DL reasoners, can be extended to deal with nominals, and implementations of the shown algorithm have already exhibited promising results. Moreover, it has also been shown that a similar technique can be applied to resolution based reasoning procedures.

The $\downarrow$ binder, another familiar feature of hybrid logics, has also been investigated in the context of DL reasoning, in this case algorithms for conjunctive query answering. It has been shown that an extension of $\mathcal{SHOIQ}$ to include a restricted form of this feature is not only decidable, but is extremely useful in the context of conjunctive query answering. Currently, the decision procedure is restricted to conjunctive queries with only simple roles and extending the technique to unrestricted conjunctive queries is still an open problem.

References

[1] Anonymous. Conjunctive query answering for $\mathcal{SHIQ}$. Technical report, 2006. http://ijcai.tripod.com/SHIQ-CQ.pdf.

[2] F. Baader and P. Hanschke. A schema for integrating concrete domains into concept languages. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)*, pages 452–457, 1991.

[3] F. Baader and U. Sattler. An overview of tableau algorithms for description logics. *Studia Logica*, 69(1):5–40, Oct. 2001.

[4] S. Bechhofer, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, and L. A. Stein. OWL web ontology language reference. W3C Recommendation, 10 February 2004. Available at http://www.w3.org/TR/owl-ref/.

[5] P. Blackburn and J. Seligman. Hybrid languages. *Journal of Logic, Language and Information*, 4(3):251–272, 1995. Special issue on decompositions of first-order logic.

[6] D. Calvanese, G. De Giacomo, and M. Lenzerini. 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)*, pages 149–158, 1998.

[7] D. Calvanese, G. De Giacomo, M. Lenzerini, D. Nardi, and R. Rosati. Description logic framework for information integration. In *Proc. of the 6th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 2–13, 1998.

[8] B. Glimm, I. Horrocks, and U. Sattler. Conjunctive query answering for description logics with transitive roles. In *Proc. of the 2006 Description Logic Workshop (DL 2006)*, pages 3–14. CEUR Electronic Workshop Proceedings, http://ceur-ws.org/Vol-189/, 2006.

[9] B. Glimm, I. Horrocks, and U. Sattler. Conjunctive query answering for the description logic $\mathcal{SHOIQ}$. Technical report, The University of Manchester, 2006. http://www.cs.man.ac.uk/~horrocks/Publications/download/2006/GlHS06b.pdf.

[10] E. Grädel. The decidability of guarded fixed point logic. In J. Gerbrandy, M. Marx, M. de Rijke, and Y. Venema, editors, *Essays Dedicated to Johan van Benthem on the Occasion of his 50th Birthday*. Amsterdam University Press, 1999.

[11] V. Haarslev and R. Möller. Description of the RACER system and its applications. In *Proc. of the 2001 Description Logic Workshop (DL 2001)*, pages 132–141. CEUR Electronic Workshop Proceedings, http://ceur-ws.org/Vol-49/, 2001.

[12] 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.

[13] J. Hladik and J. Model. Tableau systems for SHIO and SHIQ. In *Proc. of the 2004 Description Logic Workshop (DL 2004)*. CEUR, 2004. Available from ceur-ws.org.

[14] 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.

[15] I. Horrocks and U. Sattler. Ontology reasoning in the $\mathcal{SHOQ}$(D) description logic. In *Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001)*, pages 199–204. Morgan Kaufmann, Los Altos, 2001.

[16] 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)*, pages 448–453, 2005.

[17] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for expressive description logics. In H. Ganzinger, D. McAllester, and A. Voronkov, editors, *Proc. of the 6th Int. Conf. on Logic for Programming and Automated Reasoning (LPAR'99)*, number 1705 in Lecture Notes in Artificial Intelligence, pages 161–180. Springer, 1999.

[18] I. Horrocks, U. Sattler, and S. Tobies. Reasoning with individuals for the description logic $\mathcal{SHIQ}$. In D. 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.

[19] I. Horrocks and S. Tessaris. A conjunctive query language for description logic ABoxes. In *Proc. of the 17th Nat. Conf. on Artificial Intelligence (AAAI 2000)*, pages 399–404, 2000.

[20] U. Hustadt, B. Motik, and U. Sattler. Reducing SHIQ-description logic to disjunctive datalog programs. In *Proc. of the 9th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2004)*, pages 152–162, 2004.

[21] Y. Kazakov and B. Motik. A resolution-based decision procedure for $\mathcal{SHOIQ}$. In *IJCAR-06*, 2006.

[22] M. Lenzerini and A. Schaerf. Concept languages as query languages. In *Proc. of the 9th Nat. Conf. on Artificial Intelligence (AAAI'91)*, pages 471–476, 1991.

[23] D. L. McGuinness and J. R. Wright. An industrial strength description logic-based configuration platform. *IEEE Intelligent Systems*, pages 69–77, 1998.

[24] B. Motik, U. Sattler, and R. Studer. Query answering for OWL-DL with rules. In *Proc. of the 3rd Int. Semantic Web Conf. (ISWC'04)*, 2004.

[25] M. M. Ortiz, D. Calvanese, and T. Eiter. Characterizing data complexity for conjunctive query answering in expressive description logics. In *Proc. of the 21st Nat. Conf. on Artificial Intelligence (AAAI'06)*, 2006. to appear.

[26] L. Pacholski, W. Szwast, and L. Tendera. Complexity of two-variable logic with counting. In *Proc. of the 12th IEEE Symp. on Logic in Computer Science (LICS'97)*, pages 318–327. IEEE Computer Society Press, 1997.

[27] J. Pan and I. Horrocks. Web ontology reasoning with datatype groups. In D. Fensel, K. Sycara, and J. Mylopoulos, editors, *Proc. of the 2003 International Semantic Web Conference (ISWC 2003)*, number 2870 in Lecture Notes in Computer Science, pages 47–63. Springer, 2003.

[28] A. Schaerf. Reasoning with individuals in concept languages. *Data Knowledge Engineering*, 13(2):141–176, 1994.

[29] K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)*, pages 466–471, 1991.

[30] E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical owl-dl reasoner. Submitted for publication to Journal of Web Semantics, 2005.

[31] S. Tessaris. *Questions and Answers: Reasoning and Querying in Description Logic.* PhD thesis, University of Manchester, Department of Computer Science, Apr. 2001.

[32] S. Tobies. The complexity of reasoning with cardinality restrictions and nominals in expressive description logics. *J. of Artificial Intelligence Research*, 12:199–217, 2000.

[33] S. Tobies. *Complexity Results and Practical Algorithms for Logics in Knowledge Representation.* PhD thesis, LuFG Theoretical Computer Science, RWTH-Aachen, Germany, 2001.

[34] D. Tsarkov and I. Horrocks. FaCT++ description logic reasoner: System description. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2006)*, 2006. To appear.

[35] M. Y. Vardi. Why is modal logic so robustly decidable. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 31, pages 149–184. American Mathematical Society, 1997.