Data Complexity of Query Answering in Description Logics (Extended Abstract)*

D. Calvanese[1], G. De Giacomo[2], D. Lembo[2], M. Lenzerini[2], R. Rosati[2]

[1] Faculty of Computer Science
Free University of Bozen-Bolzano
calvanese@inf.unibz.it

[2] Dip. di Ing. Informatica, Automatica e Gestionale
Sapienza Università di Roma
lastname@dis.uniroma1.it

Abstract

We study the data complexity of answering conjunctive queries over Description Logic knowledge bases constituted by a TBox and an ABox. In particular, we are interested in characterizing the FO-rewritability and the polynomial tractability boundaries of conjunctive query answering, depending on the expressive power of the DL used to express the knowledge base. What emerges from our complexity analysis is that the Description Logics of the *DL-Lite* family are essentially the maximal logics allowing for conjunctive query answering through standard database technology.

1 Introduction

The idea of using ontologies as a conceptual view over data repositories is becoming more and more popular. For example, in Enterprise Application Integration [Lee *et al.*, 2003], Data Integration [Lenzerini, 2002], and the Semantic Web [Heflin and Hendler, 2001], the intensional level of the application domain can be profitably represented by an ontology, so that clients can rely on a shared conceptualization when accessing the services provided by the system. In these contexts, the set of instances of the concepts in the ontology is to be managed in the data layer of the system architecture (e.g., in the lowest of the three tiers of the Enterprise Software Architecture), and, since instances correspond to the data items of the underlying information system, such a layer constitutes a repository that is very large (much larger than the intensional level of the ontology), to be stored in secondary storage (see, e.g., [Borgida *et al.*, 1989]).

When clients access the application ontology, it is very likely that one of the main services they need is the one of answering *complex* queries over the extensional level of the ontology, which means computing the query answers that are logically implied by the whole ontology. Here, by 'complex' we mean that it does not suffice to ask for the instances of concepts, but we need at least to express conjunctive conditions on the extensional level [Calvanese *et al.*, 1998; Horrocks and Tessaris, 2000; Fikes *et al.*, 2005; Calvanese *et al.*, 2007b; 2008; Lutz, 2008; Glimm *et al.*, 2008]. Given the size of the instance repository, when measuring the computational complexity of query answering (and reasoning in general) the most important parameter is the size of the data, i.e., we are interested in the so-called *data complexity* of query answering [Vardi, 1982].

We consider here conjunctive queries (CQs) specified over ontologies expressed in Description Logics (DLs), and study the data complexity of query answering. Since an ontology in DLs is essentially a knowledge base (KB) constituted by a TBox and an ABox, the problem we address is the one of computing the answers to a CQ that are logical consequences of the TBox and the ABox, where complexity is measured with respect to the size of the ABox only. Note that we borrow the notion of data complexity from the database literature [Vardi, 1982], on the premise that an ABox can be naturally viewed as a relational database. Recently, data complexity has attracted the interest of the DL community, first for reasoning over TBox and ABox (i.e., instance checking, which is the simplest form of query answering) [Donini *et al.*, 1994; Hustadt *et al.*, 2005], and then also for answering full conjunctive queries [Ortiz *et al.*, 2008; Eiter *et al.*, 2009]. This gave rise to the study of DLs for which query answering can be done efficiently in data complexity [Calvanese *et al.*, 2005; 2007a; Krisnadhi and Lutz, 2007; Artale *et al.*, 2009], which is a key aspect of the present paper.

Specifically, we are interested in characterizing the FO-rewritability and the polynomial tractability boundaries of conjunctive query answering, depending on the expressive power of the DL used to specify the KB. We say that query answering is *FO-rewritable* in a DL $\mathcal{L}$, if for every conjunctive query q over an $\mathcal{L}$ TBox $\mathcal{T}$, one can effectively compute a first-order (FO) query q_r with the following property: for all ABoxes $\mathcal{A}$, the answers to q with respect to the KB $\langle \mathcal{T}, \mathcal{A} \rangle$ are the same as the answers to q_r over the database corresponding to the ABox $\mathcal{A}$. Since first-order queries can be expressed in SQL, the importance of FO-rewritability is that, when query answering enjoys this property, we can take advantage of Relational Data Base Management System (RDBMS) techniques for both representing data, i.e., ABox assertions, and answering queries via reformulation into SQL. Notably, in this case, the data complexity of conjunctive query answering over ontologies is the one of evaluating FO queries over relational databases, i.e., AC^0 [Abiteboul *et al.*, 1995], a

*This paper is an extended abstract of the Artificial Intelligence Journal publication [Calvanese *et al.*, 2013].

complexity class strictly contained in LOGSPACE [Reingold, 2008]. In the class of DLs for which query answering is FO-rewritable, we essentially find the DLs of the *DL-Lite* family [Calvanese *et al.*, 2007a]. Notably, the two simplest DLs of this family (namely, *DL-Lite*$_\mathcal{R}$ and *DL-Lite*$_\mathcal{F}$) are rich enough to express basic ontology languages, e.g., extensions of (the DL subset of) RDFS, or fragments of OWL 2; conceptual data models, e.g., Entity-Relationship; and object-oriented formalisms, e.g., basic UML class diagrams. In fact, in the present paper we consider a new DL of the *DL-Lite* family, called *DLR-Lite*$_{\mathcal{A},\sqcap}$, which generalizes both *DL-Lite*$_\mathcal{R}$ and *DL-Lite*$_\mathcal{F}$ by allowing for the use of n-ary relations between (instances of) concepts, the specification of keys on relations, combined together (in a controlled way) with inclusions between (projections on) relations, and the use of conjunctions in the left-hand side of the inclusion assertions constituting the knowledge base TBox.

We are also interested in knowing for which DLs we go beyond FO-rewritability. For this purpose, we single out those DLs for which query answering becomes NLOGSPACE-hard and PTIME-hard, respectively, thus not allowing for FO-rewritability. In spite of the fact that for such languages query answering is polynomially tractable (in NLOGSPACE and PTIME, respectively), these hardness results tell us that for query answering we cannot take advantage of state-of-the-art database query optimization strategies, and this might hamper practical feasibility for very large ABoxes.

Finally, we address the problem of going even beyond PTIME, establishing coNP-hardness of conjunctive query answering with respect to data complexity for surprisingly simple DLs. In particular, we show that we get intractability as soon as the DL is able to express simple forms of union.

What emerges from our complexity analysis is that the DLs of the *DL-Lite* family enjoy FO-rewritability of conjunctive query answering and cannot be extended with any construct typical of DLs [Baader *et al.*, 2003] without losing this property. In this sense, the DLs of the *DL-Lite* family studied here are the maximal logics that allow for answering conjunctive queries through standard database technology.

This paper is an extend abstract of [Calvanese *et al.*, 2013]. For a complete treatment, formal proofs of all results, and a thorough discussion of related work we refer to the full paper.

2 Preliminaries

Description Logics (DLs) [Baader *et al.*, 2003] are logics that represent the domain of interest in terms of *objects*, i.e., individuals, *concepts*, which are abstractions for sets of objects, and *relations* among concepts. Relations are typically binary in DLs (they are called *roles*), but in this paper we also consider n-ary relations, in the spirit of the DL DLR [Calvanese *et al.*, 1998; 2008].

A DL knowledge base (KB) $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is a pair formed by a set $\mathcal{T}$, called *TBox*, of intensional assertions, i.e., axioms specifying general properties of concepts, and a set $\mathcal{A}$, called *ABox*, of extensional assertions, i.e., axioms about individual objects. Informally, in an ABox a concept assertion specifies that an object is an instance of an atomic concept. Analogously, the other types of ABox assertions specify instances of atomic roles and relations.

Assertion Type	DL Syntax
concept inclusion	$Cl \sqsubseteq Cr$
role inclusion	$Ql \sqsubseteq Qr$
relation inclusion	$Vl \sqsubseteq Vr$
role functionality assertion	$(\mathsf{funct}\ Q)$
relation key assertion	$(\mathsf{key}\ j_1, \ldots, j_\ell\text{:}\, V)$
concept ABox assertion	$A(a)$
role ABox assertion	$P(a_1, a_2)$
relation ABox assertion	$R(a_1, \ldots, a_n)$

Table 1: TBox and ABox assertions

Different DLs allow for both different concept and role expressions, and different TBox intensional assertions. In other words, defining a specific DL means providing a specification of both the language for building complex expressions, and the language for specifying intensional assertions.

In this paper we consider the following constructs:

$$\begin{array}{lcl} C & \longrightarrow & A \mid \neg C \mid C \sqcap \cdots \sqcap C \mid C \sqcup \cdots \sqcup C \mid \\ & & \exists Q \mid \exists Q.C \mid \forall Q.C \mid \exists i\text{:}R \\ Q & \longrightarrow & P \mid P^- \mid \neg P \mid \neg P^- \\ V & \longrightarrow & R \mid R[i_1, \ldots, i_h] \mid \neg R \mid \neg R[i_1, \ldots, i_h], \end{array}$$

where A denotes an atomic concept, P an atomic role, and P^- its inverse, $\neg$, $\sqcap$, $\sqcup$ are the usual Boolean constructs, and R is an atomic n-ary relation. $\exists Q.C$ is the *qualified existential role quantification* denoting the objects that have a Q-role successor that belongs to C. $\exists Q$ is its *unqualified* variant denoting simply the objects that have a Q-role successor. $\forall Q.C$ is the *universal quantification*, denoting the objects whose Q role successors are all in C. $\exists i\text{:}R$ denotes the objects that participate as i-th component of the n-ary relation R. Finally $R[i_1, \ldots, i_h]$ is the relation corresponding to the projection of R on its $i_1, \ldots, i_h$ components.

Table 1 illustrates the various TBox and ABox assertions used in this paper. We distinguish between the constructs that we allow in the left-hand side (Cl, Ql, Vl) and in the right-hand side (Cr, Qr, Vr) of inclusion assertions. As TBox assertions we have concept, role, and relation inclusions, expressing containment between sets, pairs, and tuples of objects, respectively. We then have role functionality assertions expressing that a (direct or inverse) role is a functional binary relation, and key assertions expressing that there are no two tuples of objects in a relation V sharing the same key components $j_1, \ldots, j_\ell$. As ABox assertions, we have ground facts involving atomic concepts, roles, and relations.

The semantics of a DL KB is given in terms of first-order interpretations. We say that one such interpretation $\mathcal{I}$ is a *model* of a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ if $\mathcal{I}$ satisfies all assertions in $\mathcal{T}$ and in $\mathcal{A}$. $\mathcal{K}$ is *satisfiable* if it has at least a model. Given a sentence ϕ, $\mathcal{K} \models \phi$ denotes the implication of ϕ by $\mathcal{K}$, i.e., the fact that ϕ holds in all models of $\mathcal{K}$.

Example 1. Let us assume that our signature includes the atomic concepts $Supplier$, $Customer$, and $Product$, the ternary relation $supply$, and the binary relation $clientOf$. We define the following TBox $\mathcal{T}_e$:

$$\exists 1{:}supply \sqsubseteq Supplier \quad (1)$$
$$\exists 2{:}supply \sqsubseteq Customer \quad (2)$$
$$\exists 3{:}supply \sqsubseteq Product \quad (3)$$
$$Product \sqsubseteq \neg Supplier \quad (4)$$
$$Product \sqsubseteq \neg Customer \quad (5)$$
$$(\mathsf{key}\ 2, 3{:}\ supply) \quad (6)$$
$$Supplier \sqcap Customer \sqsubseteq \exists 1{:}supply \quad (7)$$
$$Supplier \sqcap Customer \sqsubseteq \exists 2{:}supply \quad (8)$$
$$supply[1, 2] \sqsubseteq clientOf[2, 1] \quad (9)$$

In the above TBox, inclusions (1)–(3) specify the domain respectively of the first, second, and third component of the relation *supply*, with the intended meaning that suppliers provide customers with products. Assertions (4) and (5) impose that the set of products is disjoint from the set of customers and the set of suppliers, respectively. Assertion (6) imposes that positions 2 and 3 in *supply* constitute a key of *supply*, with the intended meaning that a customer for a certain product has only one supplier. Assertions (7) and (8) specify that those individuals that are both suppliers and customers must participate in both the first and the second component of the relation *supply*. Finally, assertion (9) says that each individual that is a supplier of a customer (for a certain product), has such a customer as a client. As an ABox, consider:

$$\mathcal{A}_e = \{\ Customer(\mathsf{SmithInc}),\ Supplier(\mathsf{SmithInc}),\ clientOf(\mathsf{SmithInc}, \mathsf{SmartCompany})\ \}$$ ∎

We consider *conjunctive queries* (CQs) over satisfiable DL KBs. Given one such query q over a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, answering q over $\mathcal{K}$ amounts to computing its *certain answers over* $\mathcal{K}$, i.e., all the tuples $\vec{t}$ of constants in $\mathcal{K}$ such that $\mathcal{K} \models q(\vec{t})$, where $q(\vec{t})$ denotes the query obtained from q by replacing its free variables with the constants in $\vec{t}$.

We say that (conjunctive) query answering in a DL language $\mathcal{L}$ is *FO-rewritable*, if for every TBox $\mathcal{T}$ expressed in $\mathcal{L}$ and every (conjunctive) query q over $\mathcal{T}$, one can effectively compute a FO query q_r over $\mathcal{T}$ such that the following holds: for every ABox $\mathcal{A}$ for which $\langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable, and every tuple $\vec{t}$ of constants occurring in $\mathcal{A}$, $\langle \mathcal{T}, \mathcal{A} \rangle \models q(\vec{t})$ if and only if $\langle \emptyset, \mathcal{A} \rangle \models q_r(\vec{t})$. The query q_r is called the *FO-rewriting of q w.r.t. $\mathcal{T}$*. In this paper, we study *data complexity* of query answering, i.e., the complexity of the decision problem associated to query answering measured w.r.t. the size of the ABox only, i.e., by considering both the input TBox and the input query as fixed. Observe that every FO query can be evaluated in AC^0 with respect to data complexity (see e.g., [Abiteboul *et al.*, 1995]). It follows that, if query answering in $\mathcal{L}$ is FO-rewritable, then query answering in $\mathcal{L}$ is in AC^0 w.r.t. data complexity. Vice-versa, if query answering (or KB satisfiability) is $\mathcal{C}$-hard w.r.t. data complexity for some complexity class $\mathcal{C}$ that strictly contains AC^0 (e.g., LOGSPACE, NLOGSPACE, PTIME, coNP, etc.), then it is not FO-rewritable.

3 *DLR-Lite*$_{\mathcal{A},\sqcap}$ and FO-rewritability

The logic *DLR-Lite*$_{\mathcal{A},\sqcap}$ is one the most expressive members of the *DL-Lite* family. It extends the basic *DL-Lite* logics *DL-Lite*$_{\mathcal{R}}$ and *DL-Lite*$_{\mathcal{F}}$ [Calvanese *et al.*, 2007a] with n-ary relations, inclusions between projections of n-ary relations, conjunction in the left-hand side of concept inclusions, and key dependencies on (projections of) n-ary relations (in a controlled way). The syntax of *DLR-Lite*$_{\mathcal{A},\sqcap}$ is given in Table 2. The KB given in Example 1 is a *DLR-Lite*$_{\mathcal{A},\sqcap}$ KB.

	DLR-Lite$_{\mathcal{A},\sqcap}$
Cl	$A \mid \exists i{:}R \mid Cl_1 \sqcap \cdots \sqcap Cl_n$
Cr	$A \mid \exists i{:}R \mid \neg A \mid \neg\exists i{:}R$
Vl	$R \mid R[i_1, \ldots, i_h]$
Vr	$Vl \mid \neg Vl$
TBox assertions	$Cl \sqsubseteq Cr \qquad Vl \sqsubseteq Vr$ $(\mathsf{key}\ j_1, \ldots, j_\ell{:}\ Vl)$ $^{(*)}$
ABox assertions	$A(a) \qquad R(a_1, \ldots, a_n)$

$(*)$: relations occurring in key assertions cannot occur positively in the right-hand side of inclusion assertions between relations.

Table 2: Syntax of *DLR-Lite*$_{\mathcal{A},\sqcap}$

In [Calvanese *et al.*, 2013], we show that query answering in *DLR-Lite*$_{\mathcal{A},\sqcap}$ is FO-rewritable, and therefore that it is in AC^0 in data complexity. To this aim we provide an algorithm, called PerfectRef, that, taken a CQ q and a *DLR-Lite*$_{\mathcal{A},\sqcap}$ TBox $\mathcal{T}$ as input, returns a FO query q_r, in fact a union of conjunctive queries, that we prove to be a FO-rewriting of q w.r.t. $\mathcal{T}$. In a nutshell, the algorithm compiles in q_r both the query q and the assertions of $\mathcal{T}$ that are relevant to compute the answers to q. Notably, to compute q_r, PerfectRef needs to consider only the positive inclusions explicitly asserted in $\mathcal{T}$, i.e., inclusions that do not use negation. Such inclusions are used as rewriting rules, applied from right to left, for the query q. Besides rewriting steps, PerfectRef performs also some unifications between query atoms, which may make further rewriting steps applicable. Both such steps are executed iteratively until a fixpoint is reached. We refer to [Calvanese *et al.*, 2013] for further details. We notice that the algorithm we described is structurally similar to an analogous algorithm presented in [Calvanese *et al.*, 2007a] for computing the FO-rewriting of a CQ over either a *DL-Lite*$_{\mathcal{R}}$ or a *DL-Lite*$_{\mathcal{F}}$ TBox. The PerfectRef algorithm we discuss here is however complicated by the fact that while rewriting the query, it has to properly manage the presence of n-ary relations, both in the query and in the inclusions, and the presence of conjunctions in concept inclusions. In particular, such conjunctions may cause some CQs produced by the algorithm as part of the rewriting to have more atoms than the original input query (notice that for *DL-Lite* logics without conjunctions in the left-hand side of inclusions, all rewritten CQs have at most as many atoms as the input query).

Example 2. Consider the query $q(x) \leftarrow supply(x, y, z), Product(z)$, posed over the TBox $\mathcal{T}_e$ of Example 1. The algorithm applies inclusion (3) and generates the query $q(x) \leftarrow supply(x, y, z), supply(w_1, w_2, z)$. Such a query cannot be further rewritten through positive inclusions. Indeed, the only inclusions having *supply* in their right-hand side, i.e., inclusions (7) and (8), do not propagate all join and free variables occurring in the query atoms. By unifying the two

supply atoms in the query, with unifier $\{w_1/x, w_2/y\}$, the algorithm then produces the query $q(x) \leftarrow supply(x, y, z)$. Such a query can be rewritten using inclusion (7), and the query $q(x) \leftarrow Supplier(x), Customer(x)$ is thus added to the rewriting. We notice that the unification step is necessary to generate this query. The evaluation of the last query over the ABox $\mathcal{A}_e$ produces the set $\{\mathsf{SmithInc}\}$. Such a set constitutes in fact the set of certain answers of the input query over the KB $\langle \mathcal{T}_e, \mathcal{A}_e \rangle$. ∎

4 Going beyond FO-rewritability

Next, we show that, as soon as we consider further, minimal extensions of *DLR-Lite*$_{\mathcal{A},\sqcap}$, we cross the boundary of AC^0 data complexity. Going beyond AC^0 data complexity means actually that we lose the property of FO-rewritability and therefore query answering requires more powerful engines than those available in standard relational database technology. An immediate consequence of this fact is that we cannot take advantage anymore of data management tools and query optimization techniques of current DBMSs.

We point out that the extensions of *DLR-Lite*$_{\mathcal{A},\sqcap}$ that we consider make query answering harder even also if we restrict relations to be binary, i.e., if we have only roles. Therefore, in the following, we consider DLs with roles rather than n-ary relations. Moreover, such extensions have a negative impact on complexity even if we consider them alone or added to very basic DLs. Specifically, we study the complexity of query answering for DLs in which the ABox of a KB is as described in Table 1, while the TBox consists of *(i)* concept inclusion assertions of the form $Cl \sqsubseteq Cr$, where the syntax of Cl and Cr is defined case by case (cf. Table 3), and *(ii)* possibly, key (actually, functionality) assertions on roles.

In our investigation we identify various DL languages for which conjunctive query answering is NLOGSPACE-hard, PTIME-hard, or coNP-hard.

As for NLOGSPACE-hard DLs, we notice that to reach such complexity it suffices to allow for the use of existential role quantification qualified on atomic concepts in the left-hand side of positive concept inclusions (cf. row 2 in Table 3). Indeed, this kind of TBox assertion requires per se the use of recursion to answer a CQ. To get an intuition, consider for example the assertion $\exists$*fatherOf*.*Human* $\sqsubseteq$ *Human*, which states that each individual that is father of a human is also a human. Then, to answer a query asking for all individuals that are humans, we need to retrieve those that are asserted to be human, those that are fathers of asserted humans, those that are fathers of fathers of asserted humans, and so on. This computation requires linear recursion. In [Calvanese *et al.*, 2013] we indeed prove NLOGSPACE-hardness for this case by a LOGSPACE reduction from reachability in directed graphs. For the logic in row 3 in Table 3, NLOGSPACE-hardness is proved by showing that every TBox in this logic is logically equivalent to a TBox expressed in the logic of row 2. As for the logic in row 4, we exhibit a different reduction from reachability in directed graphs. Notice that for all such DLs we need (at least) linear Datalog to answer CQs.

To prove PTIME-hardness results, one can use LOGSPACE reductions from path system accessibility, see [Calvanese *et al.*, 2013]. Note that PTIME-hardness implies that we need at least the power of full Datalog to answer CQs in these cases.

	Cl	Cr	$\mathcal{F}$	$\mathcal{R}$	Complexity
1	*DLR-Lite*$_{\mathcal{A},\sqcap}$		$\surd$(*)	$\surd$(*)	in AC^0
2	$A \mid \exists P.A$	A	–	–	NL-c
3	A	$A \mid \forall P.A$	–	–	NL-c
4	A	$A \mid \exists P.A$	$\surd$	–	NL-hard
5	$A \mid \exists P.A \mid A_1 \sqcap A_2$	A	–	–	PTIME-c
6	$A \mid A_1 \sqcap A_2$	$A \mid \forall P.A$	–	–	PTIME-c
7	$A \mid A_1 \sqcap A_2$	$A \mid \exists P.A$	$\surd$	–	PTIME-c
8	$A \mid \exists P.A \mid \exists P^-.A$	$A \mid \exists P$	–	–	PTIME-c
9	A	$A \mid A_1 \sqcup A_2$	–	–	coNP-c
10	$A \mid \neg A$	A	–	–	coNP-c
11	$A \mid \forall P.A$	A	–	–	coNP-c

Legenda: Cl/Cr = left/right-hand side of concept inclusions, $\mathcal{F}$ = functionalities/keys allowed, $\mathcal{R}$ = role/relation inclusions allowed, A, A_1, A_2 = atomic concepts, P = atomic role. NLOGSPACE (abbreviated to NL) and PTIME hardness results hold already for instance checking. (*) Relations in key assertions are not specialized.

Table 3: Data Complexity of Query Answering in DLs

The intuition behind the coNP-hardness results given in Table 3 is that in all three cases we consider it is possible to require a reasoning by case analysis, caused by set covering assertions. Indeed, whereas in the first coNP-hard logic (cf. row 9) we can explicitly assert set covering through the use of disjunction in Cr, for the two other cases covering can be asserted on the entire domain. More precisely, in the second coNP-hard logic (cf. row 10), the domain can be covered by A_1 and A_2 through an assertion of the form $\neg A_1 \sqsubseteq A_2$, whereas in the last case (cf. row 11), the domain can be implicitly covered by existentials in the CQ and an assertion of the form $\forall P.A_1 \sqsubseteq A_2$, see [Calvanese *et al.*, 2013].

Finally, note that all our NLOGSPACE-hardness and PTIME-hardness results hold already for *instance checking*, a simpler form of query answering where the query is a ground atom. All our complexity bounds are tight ('-c'(omplete) in Table 3) [Calvanese *et al.*, 2013], except for the DL in row 4, for which NLOGSPACE-membership is open.

5 Conclusions

In this paper, we have presented *DLR-Lite*$_{\mathcal{A},\sqcap}$, an interesting n-ary DL belonging to the *DL-Lite* family, which can be considered as the relation oriented version of the well known logic *DL-Lite*$_{\mathcal{A}}$, and which enjoys nice computational properties, in particular FO-rewritability and thus AC^0 data complexity of query answering. We have then identified rather simple DLs for which query answering is no longer reducible to evaluation of a first-order logic formula (and hence an SQL query) over the data. The results provided in this paper are summarized in Table 3. We observe that data complexity of query answering has become a central theme in the current DL research, and specific attention is now given to FO-rewritability in designing languages for ontology applications that need to deal with large quantities of data [Artale *et al.*, 2009; Ortiz *et al.*, 2011; Lutz and Wolter, 2012; Stefanoni *et al.*, 2014; Bienvenu *et al.*, 2014].

References

[Abiteboul *et al.*, 1995] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison Wesley Publ. Co., 1995.

[Artale *et al.*, 2009] Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. The *DL-Lite* family and relations. *J. Artif. Intell. Res.*, 36:1–69, 2009.

[Baader *et al.*, 2003] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.

[Bienvenu *et al.*, 2014] Meghyn Bienvenu, Diego Calvanese, Magdalena Ortiz, and Mantas Simkus. Nested regular path queries in description logics. In *Proc. of KR*. AAAI Press, 2014.

[Borgida *et al.*, 1989] Alexander Borgida, Ronald J. Brachman, Deborah L. McGuinness, and Lori Alperin Resnick. CLASSIC: A structural data model for objects. In *Proc. of ACM SIGMOD*, pages 59–67, 1989.

[Calvanese *et al.*, 1998] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. On the decidability of query containment under constraints. In *Proc. of PODS*, pages 149–158, 1998.

[Calvanese *et al.*, 2005] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. *DL-Lite*: Tractable description logics for ontologies. In *Proc. of AAAI*, pages 602–607, 2005.

[Calvanese *et al.*, 2007a] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. of Automated Reasoning*, 39(3):385–429, 2007.

[Calvanese *et al.*, 2007b] Diego Calvanese, Thomas Eiter, and Magdalena Ortiz. Answering regular path queries in expressive description logics: An automata-theoretic approach. In *Proc. of AAAI*, pages 391–396, 2007.

[Calvanese *et al.*, 2008] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Conjunctive query containment and answering under description logics constraints. *ACM Trans. on Computational Logic*, 9(3):22.1–22.31, 2008.

[Calvanese *et al.*, 2013] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Data complexity of query answering in description logics. *Artificial Intelligence*, 195:335–360, 2013.

[Donini *et al.*, 1994] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Andrea Schaerf. Deduction in concept languages: From subsumption to instance checking. *J. of Logic and Computation*, 4(4):423–452, 1994.

[Eiter *et al.*, 2009] Thomas Eiter, Carsten Lutz, Magdalena Ortiz, and Mantas Šimkus. Query answering in description logics with transitive roles. In *Proc. of IJCAI*, pages 759–764, 2009.

[Fikes *et al.*, 2005] Richard Fikes, Patrick Hayes, and Ian Horrocks. OWL-QL: A language for deductive query answering on the semantic web. *J. of Web Semantics*, 2(1):19–29, 2005.

[Glimm *et al.*, 2008] Birte Glimm, Carsten Lutz, Ian Horrocks, and Ulrike Sattler. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. *J. Artif. Intell. Res.*, 31:151–198, 2008.

[Heflin and Hendler, 2001] Jeff Heflin and James Hendler. A portrait of the Semantic Web in action. *IEEE Intelligent Systems*, 16(2):54–59, 2001.

[Horrocks and Tessaris, 2000] Ian Horrocks and Sergio Tessaris. A conjunctive query language for description logic ABoxes. In *Proc. of AAAI*, pages 399–404, 2000.

[Hustadt *et al.*, 2005] Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Data complexity of reasoning in very expressive description logics. In *Proc. of IJCAI*, pages 466–471, 2005.

[Krisnadhi and Lutz, 2007] Adila Krisnadhi and Carsten Lutz. Data complexity in the $\mathcal{EL}$ family of description logics. In *Proc. of LPAR*, pages 333–347, 2007.

[Lee *et al.*, 2003] Jinyoul Lee, Keng Siau, and Soongoo Hong. Enterprise integration with ERP and EAI. *Communications of the ACM*, 46(2):54–60, 2003.

[Lenzerini, 2002] Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proc. of PODS*, pages 233–246, 2002.

[Lutz and Wolter, 2012] Carsten Lutz and Frank Wolter. Non-uniform data complexity of query answering in description logics. In *Proc. of KR*, pages 297–307, 2012.

[Lutz, 2008] Carsten Lutz. The complexity of conjunctive query answering in expressive description logics. In *Proc. of IJCAR*, volume 5195 of *LNAI*, pages 179–193. Springer, 2008.

[Ortiz *et al.*, 2008] Magdalena Ortiz, Diego Calvanese, and Thomas Eiter. Data complexity of query answering in expressive description logics via tableaux. *J. of Automated Reasoning*, 41(1):61–98, 2008.

[Ortiz *et al.*, 2011] Magdalena Ortiz, Sebastian Rudolph, and Mantas Simkus. Query answering in the Horn fragments of the description logics $\mathcal{SHOIQ}$ and $\mathcal{SROIQ}$. In *Proc. of IJCAI*, pages 1039–1044. IJCAI/AAAI, 2011.

[Reingold, 2008] Omer Reingold. Undirected connectivity in Log-Space. *J. of the ACM*, 55(4):17:1–17:24, 2008.

[Stefanoni *et al.*, 2014] Giorgio Stefanoni, Boris Motik, Markus Krötzsch, and Sebastian Rudolph. The complexity of answering conjunctive and navigational queries over OWL 2 EL knowledge bases. *J. Artif. Intell. Res.*, 51:645–705, 2014.

[Vardi, 1982] Moshe Y. Vardi. The complexity of relational query languages. In *Proc. of STOC*, pages 137–146, 1982.