View-based query answering in Description Logics: Semantics and complexity

Diego Calvanese [a],*, Giuseppe De Giacomo [b], Maurizio Lenzerini [b], Riccardo Rosati [b]

[a] KRDB Research Centre for Knowledge and Data, Free University of Bozen-Bolzano, Piazza Domenicani 3, I-39100 Bolzano, Italy
[b] Dipartimento di Informatica e Sistemistica, Sapienza Università di Roma, Via Ariosto 25, I-00185 Roma, Italy

A B S T R A C T

View-based query answering is the problem of answering a query based only on the precomputed answers to a set of views. While this problem has been widely investigated in databases, it is largely unexplored in the context of Description Logic ontologies. Differently from traditional databases, Description Logics may express several forms of incomplete information, and this poses challenging problems in characterizing the semantics of views. In this paper, we first present a general framework for view-based query answering, where we address the above semantical problems by providing two notions of view-based query answering over ontologies, all based on the idea that the precomputed answers to views are the certain answers to the corresponding queries. We also relate such notions to privacy-aware access to ontologies. Then, we provide decidability results, algorithms, and data complexity characterizations for view-based query answering in several Description Logics, ranging from those with limited modeling capability to highly expressive ones.

© 2011 Elsevier Inc. All rights reserved.

1. Introduction

View-based query processing is the problem of processing a query under the assumption that the only accessible extensional information consists of the precomputed answers to a set of queries, called views. Several articles in the literature point out that this problem [1–3] is relevant in many aspects of information management, including query optimization, data warehousing, data integration, and query answering with incomplete information. In all these contexts, the problem arises of answering a query posed to a database only on the basis of the information about a set of views, which are again queries over the same database. In query optimization, the problem is relevant because using the views may speed up query processing. In data integration, the views represent the only information sources accessible to answer a query. A data warehouse can be seen as a set of materialized views, and, therefore, query processing reduces to view-based query answering. Finally, since the views provide partial knowledge on the database, view-based query processing can be seen as a special case of query answering with incomplete information.

In this article, the context that we are most interested in is the one of privacy-aware access to information. In the logical approach to privacy-aware data access, each user (or, class of users) is associated with a set of views, called authorization

views, which specify the information that the user is allowed to access [4–6]. View-based query answering in this setting captures the requirement that only information deriving from such views can be revealed to the user.

There are two approaches to view-based query processing, called *query rewriting* and *query answering*, respectively. In the former approach, originated with [7], we are given a query q and a set of view definitions, and the goal is to reformulate the query into an expression that refers to the views (or only to the views), and provides the answer to q when evaluated over the view extension. Typically, the rewriting is expressed in the same language used for both the query and the views. The latter approach, originated in [2], takes a more direct route: based on the view definitions and the view extensions, it tries to compute the so-called *certain answers*, i.e., the tuples satisfying the query in all databases consistent with the views. The relationship between the two approaches has been discussed, e.g., in [8,9]. In the rest of this article, we only deal with view-based query answering.

A large number of results is reported on view-based query answering in the recent database literature, both for the case of relational databases and for the case of semistructured and XML data. On the other hand, the problem is still largely unexplored in the context of Description Logics. We discuss related work in detail in Section 6.

Description Logics [10] (DLs) were introduced in the early 80s in the attempt to provide a formal ground to Semantic Networks and Frames. Since then, they have evolved into knowledge representation languages that are able to capture virtually all class-based representation formalisms used in Artificial Intelligence, Software Engineering, and Databases. In particular, DLs have proved adequate as logic-based formalisms for expressing ontologies [11]. Here, by ontology we mean a formal representation of a domain of interest, expressed in terms of both intensional (concepts, attributes, and relations) and extensional (instances of concepts, attributes and relations) properties. When ontologies are expressed in DLs, the intensional and the extensional assertions form the so-called TBox, and ABox, respectively. In DLs, the domain of interest is modeled by means of concepts and roles, which denote classes of objects and binary relations between classes of objects, respectively. Concepts and roles can be denoted using expressions of a specified language, and the various DLs differ in the expressive power of such a language. One of the distinguishing features of the work on these logics is the detailed computational complexity analysis both of the associated reasoning algorithms, and of the logical implication problem that the algorithms are supposed to solve. By virtue of this analysis, most of these logics have optimal reasoning algorithms, and practical systems implementing such algorithms are now used in several projects.

In this article, we present a first study on view-based query answering in DL ontologies. The idea at the basis of our work is that, differently from traditional databases, DLs may express several forms of incomplete information, and this should be taken into account in characterizing the semantics of the views. In particular, while a database can be considered as a single interpretation structure, an ontology is characterized by a set of models, and the answers to a query are those that are *certain*, i.e., those that satisfy the query in every model of this set. To address this issue, we follow the novel approach to consider the precomputed answers to views as the certain answers to the corresponding queries. Our contributions can be summarized as follows.

- We present a general formal framework for view-based query answering in DL ontologies. Users pose queries to a system, whose knowledge is represented by an ontology expressed in a given DL. The system associates to each user (or, class of users) a set of views, whose extensions are computed as certain answers to the ontology. The system answers user queries coherently with the ontology, though hiding information not implied by the views. This idea is formalized based on the fundamental notion of *solution*. Roughly speaking, given an ontology $\mathcal{K}$ with TBox $\mathcal{T}$ and ABox $\mathcal{A}$ expressed in a DL $\mathcal{L}$, and a set of views V with extensions E, a solution for $\langle \mathcal{T}, V, E \rangle$ is a set of interpretations for $\mathcal{K}$ which satisfy $\mathcal{T}$ and such that computing the certain answers of the views V over such interpretations yields exactly E.
- We show that different definitions can be provided for the semantics of view-based query answering, each one capturing specific properties for the notion of solution. In particular, we refer to two notions of solutions, yielding two different semantics for view-based query answering, called *model-centered* and *TBox-centered semantics*, respectively. In the model-centered semantics, we simply insist that a solution is a set of models of the TBox $\mathcal{T}$. In the TBox-centered semantics, we additionally require that such a set can be expressed in terms of an ABox paired to $\mathcal{T}$.
- We relate the framework to the problem of privacy-aware access to ontologies, by illustrating how view-based query answering is able to conceal from the user the information that are not logical consequences of the associated authorization views. In particular, a fact $q(\vec{t})$ that logically follows from a DL ontology $\mathcal{K}$ is concealed from the user when there is a solution for $\langle \mathcal{T}, V, E \rangle$ that falsifies $q(\vec{t})$ and cannot be distinguished from the models of $\mathcal{K}$ by using V and $\mathcal{T}$. The two semantics defined in the framework correspond to adopting different notions of solution, and this in turn implies that the different semantics disclose different amounts of information to the user.
- We illustrate several decidability results, algorithms, and data complexity characterizations for view-based query answering, under the two semantics, and for different DLs, ranging from tractable ones (the *DL-Lite* family [12,13] and the $\mathcal{EL}$ family [14–16]), to more expressive ones ($\mathcal{AL}$, and $\mathcal{SHIQ}$ [17–19]).

Generally speaking, our work shows that, for all DLs considered, the complexity of view-based query answering under the model-centered semantics is essentially the same as the complexity of computing the certain answers over an ontology. On the other hand, the complexity of view-based query answering under the TBox-centered semantics is consistently higher than under the model-centered semantics. This is an indication that the model-centered semantics may represent a good trade-off between the requirement of answering queries based on a set of views, and computing such answers efficiently.

The article is organized as follows. In the next section, we present basic notions of DL ontologies, and we provide the definition of the various DLs used in our work. Then, we illustrate in Section 3 our formal framework for view-based query answering, and its relationship with privacy-aware access to ontologies. The next two sections present general results on view-based query answering, and specific results for various DLs, both in the model-centered (Section 4) and in the TBox-centered semantics (Section 5). We discuss related work in Section 6, and we conclude the article in Section 7 with a discussion on both the results presented here, and future directions for continuing our work.

This article is a revised, corrected, and extended version of [20].

2. Preliminaries

In this section, we define some preliminary notions used in the rest of the article. In particular, we first present the notion of ontology that we adopt, and then we provide the definition of all the DLs considered in our work.

2.1. Description Logic ontologies

Let $\mathcal{S}$ be a signature of unary predicates (also called *atomic concepts*), binary predicates (also called *atomic roles*), and constants (also called *individuals*). A DL ontology $\mathcal{K}$ over the signature $\mathcal{S}$ is a pair formed by a set of assertions $\mathcal{T}$, called *TBox*, and a set of assertions $\mathcal{A}$, called *ABox*. Intuitively, $\mathcal{T}$ contains universal assertions about concepts and roles, and $\mathcal{A}$ contains assertions about individuals that are instances of concepts and roles. In the rest of the article, we implicitly refer to a given signature $\mathcal{S}$, and therefore we often omit to refer to the signature explicitly.

In TBox and ABox assertions, (complex) concepts and roles are denoted by means of expressions. An *expression* is formed starting from atomic concepts and roles, and using various constructs. Different DLs allow for different constructs in forming expressions, as we will see in the next subsection, where we will consider specific DLs.

We now formally define the notion of DL ontology.

Definition 1 *(Ontology).* A DL *ontology* is a pair $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where

- The TBox $\mathcal{T}$ is a finite set of *intensional assertions*, i.e., axioms over the signature $\mathcal{S}$.
- The ABox $\mathcal{A}$ is a finite set of *extensional assertions* of the form:

$$A(a) \quad \text{(concept membership assertion)}$$

$$P(a, b) \quad \text{(role membership assertion)}$$

$$a \neq b \quad \text{(inequality assertion)},$$

with A and P respectively an atomic concept and an atomic role occurring in $\mathcal{T}$, and a, b constants.

Note that specific DLs may rule out some of the assertions referred to in the above definition.

The intuitive meaning of TBox and ABox assertions is as follows. Each intensional assertion in $\mathcal{T}$ specifies a general property of concepts and roles. As for extensional assertions in the ABox, a membership assertion specifies that a given object (respectively, pair of objects) is an instance of a concept (respectively, a role). Finally, an inequality assertion simply states that two constants denote different objects. Note that ABox assertions do not include equality assertions. The reason is that we can enforce the equality $a = b$ in $\mathcal{A}$, with a, b constants, by simply substituting every occurrence of b with a in the assertions.

We observe that the distinction between the two components of a DL ontology serves two purposes. On the one hand, it reflects the distinction between intensional knowledge about concepts and roles, and extensional knowledge about individuals. On the other hand, we will see later in the paper that, for various computational tasks involving an ontology $\langle \mathcal{T}, \mathcal{A} \rangle$, the complexity is measured with respect to the ABox $\mathcal{A}$ only.

The semantics of a DL ontology is given in terms of first-order interpretations. Specifically, an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty interpretation domain $\Delta^{\mathcal{I}}$ and an interpretation function $\cdot^{\mathcal{I}}$ for the signature. The function $\cdot^{\mathcal{I}}$ interprets each constant c in terms of an object $c^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, each concept as a subset of $\Delta^{\mathcal{I}}$, and each role as a subset of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$.

More precisely, there are two types of interpretations, depending on whether the *unique name assumption* (UNA) is adopted or not. An interpretation $\mathcal{I}$ following the unique name assumption interprets different constants as different objects, i.e., is such that $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ for every pair of constants a, b. On the contrary, an interpretation that does not follow the unique name assumption may assign the same object to two different constants. In the following, we use the term *UNA-interpretation* to denote an interpretation following the UNA.

We now turn our attention to the notion of satisfaction of assertions. Clearly, at the stage we can only be precise about the notion of satisfaction of ABox assertions: an interpretation $\mathcal{I}$ *satisfies* a membership assertion of the form $A(a)$, if $a^{\mathcal{I}} \in A^{\mathcal{I}}$, a membership assertion of the form $P(a, b)$, if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in P^{\mathcal{I}}$, and an inequality assertion of the form $a \neq b$, if $a^{\mathcal{I}} \neq b^{\mathcal{I}}$. We refer the reader to the next subsection for the notion of satisfaction of specific TBox assertions.

Note that, if $\mathcal{I}$ is a UNA-interpretation, every inequality assertion is trivially satisfied in $\mathcal{I}$. Also, observe that the UNA for an ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ can be enforced by adding inequality assertions for every pair constants appearing in $\mathcal{A}$. However, this technique cannot be used in the case where the DL in which $\mathcal{K}$ is expressed does not allow inequality assertions.

An interpretation is a *model* of an ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ if it satisfies all TBox assertions in $\mathcal{T}$ and all ABox assertions in $\mathcal{A}$. We denote by $Mod(\mathcal{K})$ the set of models of $\mathcal{K}$, and we denote by $Mod^{UNA}(\mathcal{K})$ the set of all UNA-interpretations in $Mod(\mathcal{K})$. An ontology $\mathcal{K}$ is *satisfiable without UNA* if $Mod(\mathcal{K}) \neq \emptyset$, and *with UNA* if $Mod^{UNA}(\mathcal{K}) \neq \emptyset$

To extract information from an ontology, we consider queries over ontologies. In particular, in this article we concentrate on conjunctive queries, which form the most important class of queries studied in database theory.

Definition 2 *(Conjunctive query).* Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a DL ontology over a signature $\mathcal{S}$. A *conjunctive query* (CQ) q over $\mathcal{K}$ is an expression of the form

$$q(\vec{x}) \leftarrow conj(\vec{x}, \vec{y}),$$

where q is a symbol that is not part of the signature $\mathcal{S}$, $\vec{x}$ are the so-called *distinguished variables*, $\vec{y}$ are the *non-distinguished* variables, which are implicitly existentially quantified, and $conj(\vec{x}, \vec{y})$ is a conjunction of atoms, each of the form $A(z)$ or $P(z, z')$, where A is an atomic concept, P is an atomic role, and z, z' are constants appearing in $\mathcal{A}$ or variables in $\vec{x}$ or $\vec{y}$.

The *arity* of $\vec{x}$ is the arity of the query q. When $\vec{x}$ is the tuple $\langle \rangle$ of arity 0, then q is called a *Boolean query*. In this article we use the so-called Datalog notation for CQs [21], i.e., we write conjunctions simply as sequences.

Given an interpretation $\mathcal{I}$, $q^{\mathcal{I}}$ is the set of tuples of domain elements in $\Delta^{\mathcal{I}}$ that, when assigned to the distinguished variables $\vec{x}$ of q, make the formula $\exists \vec{y}.conj(\vec{x}, \vec{y})$ true [21]. We denote $\vec{t} \in q^{\mathcal{I}}$ also as $\mathcal{I} \models q(\vec{t})$.

We now introduce the notion of evaluating a query q with respect to an ontology $\mathcal{K}$. Query evaluation is one of the central notions in database theory. In this context, a (relational) database over the signature $\mathcal{S}$ can be seen as an interpretation $\mathcal{I}$ over $\mathcal{S}$, and evaluating a query q simply means computing $q^{\mathcal{I}}$. On the other hand, an ontology is characterized by a *set of models*, and therefore we have to resort to the notion of evaluating a query over a set of interpretations. To deal with this problem, we sanction that the certain answers to a query q over a set $\mathcal{W}$ of interpretations is the set of tuples that are the answers to q in *all* the interpretations in $\mathcal{W}$.

Formally, given a set $\mathcal{W}$ of interpretations and a CQ q, the set of *certain answers* $cert(q, \mathcal{W})$ *to* q *over* $\mathcal{W}$ is defined as follows:

$$cert(q, \mathcal{W}) = \left\{ \vec{t} \mid \vec{t} \text{ is a tuple of constants in } \mathcal{S} \text{ s.t. } \mathcal{I} \models q(\vec{t}) \text{ for every } \mathcal{I} \in \mathcal{W} \right\}.$$

Based on this notion, and considering that the semantics of an ontology $\mathcal{K}$ is captured by $Mod(\mathcal{K})$ without UNA, and by $Mod^{UNA}(\mathcal{K})$ with UNA, it is natural to define the answers to q to an ontology $\mathcal{K}$ as follows.

Definition 3 *(Certain answer).* Given an ontology $\mathcal{K}$, we define the set of *certain answers to q over $\mathcal{K}$ without UNA* as the set $cert(q, Mod(\mathcal{K}))$. Analogously, we define the set of *certain answers to q over $\mathcal{K}$ with UNA* as the set $cert(q, Mod^{UNA}(\mathcal{K}))$.

In what follows, when we talk about query answering (with and without UNA, respectively) in a DL ontology $\mathcal{K}$, we mean computing the certain answers to a conjunctive query over $\mathcal{K}$ (with and without UNA, respectively), and when we talk about the computational complexity of query answering for a particular DL $\mathcal{L}$, we refer to the complexity of checking whether $\vec{t} \in cert(q, Mod(\mathcal{K}))$ (without UNA) or $\vec{t} \in cert(q, Mod^{UNA}(\mathcal{K}))$ (with UNA), for an input ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ expressed in $\mathcal{L}$, a tuple $\vec{t}$, and a query q. In particular, we are interested in the so-called *data complexity*, i.e., the complexity of query answering with respect to the size of $\mathcal{A}$ only (and, therefore, with $\mathcal{T}$ and q fixed).

2.2. Description Logics

As we said before, different DLs allow for different constructs in concept and role expressions and for different types of intensional and extensional assertions. Some of the results presented in this paper hold for specific DLs. In this subsection, we describe such DLs, which include:

- very expressive DLs, in particular $\mathcal{SHIQ}$;
- moderately expressive DLs, such as $\mathcal{AL}$;
- lightweight DLs from the $\mathcal{EL}$ family, namely, $\mathcal{EL}$ and $\mathcal{ELH}$;
- lightweight DLs from the *DL-Lite* family, namely, *DL-Lite$_F$* and *DL-Lite$_R$*.

Actually, all DLs mentioned in this article are sub-logics of $\mathcal{SHIQ}$ [18], which is one of the most expressive DLs proposed in the literature, and is defined next. In what follows, if $\mathcal{L}$ is a DL, then the expressions in $\mathcal{L}$ denoting concepts (respectively, roles) are called $\mathcal{L}$-*concepts* (respectively, $\mathcal{L}$-*roles*).

Definition 4 *($\mathcal{SHIQ}$ ontology).* An *ontology* in $\mathcal{SHIQ}$ is a DL ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where the assertions in $\mathcal{T}$ are of the form:

$C_l \sqsubseteq C_r$ (concept inclusion assertion)

$R_l \sqsubseteq R_r$ (role inclusion assertion)

(funct R)   (role functionality assertion)

(trans R)   (role transitivity assertion),

with C_l, C_r $\mathcal{SHIQ}$-concepts, and R_l, R_r, R $\mathcal{SHIQ}$-roles. Concepts and roles in $\mathcal{SHIQ}$ are formed according to the following syntax:

$$C, C' \longrightarrow A \mid \neg C \mid C \sqcap C' \mid \exists R \mid \exists R.C \mid \forall R.C \mid \, \leqslant nR.C$$

$$R \longrightarrow P \mid P^-,$$

where A denotes an *atomic concept*, P an *atomic role*, C, C' *arbitrary concepts*, R an *arbitrary role*, and n a non-negative integer.

The intuitive meaning of TBox assertions is as follows. A concept inclusion assertion in a TBox specifies that every instance of the concept on the left-hand side is also an instance of the concept on the right-hand side. Analogously, a role inclusion assertion specifies containment between the instances of two roles. Finally, a role functionality (respectively, role transitivity) assertion specifies that a role is functional (respectively, transitive).

Note that in the syntax, we have distinguished the concepts (and roles) appearing in the left-hand side of an inclusion assertion (C_l and R_l) from those appearing in the corresponding right-hand side (C_r and R_r). The reason is that there are DLs allowing for different expressions for the two parts of such inclusions (see later).

We now discuss concept and role expressions in $\mathcal{SHIQ}$. Intuitively, $\neg C$ denotes negation of concepts, $C \sqcap C'$ concept intersection, $\exists R$ unqualified existential quantification, $\exists R.C$ qualified existential quantification, $\forall R.C$ (qualified) universal quantification, $\leqslant nR.C$ qualified maximum cardinality restriction (also called unqualified) number restriction, and P^- the *inverse* of an atomic role.

In the rest of the article, we use abbreviations for union ($\sqcup$) and qualified minimum cardinality restrictions ($\geqslant nR.C$) as follows:

$$C \sqcup C' \quad \text{for } \neg\big(\neg C \sqcap \neg C'\big),$$

$$\geqslant nR.C \quad \text{for } \neg(\leqslant n{-}1R.C).$$

Qualified maximum or minimum cardinality restrictions are also called *qualified number restrictions*. Note that the constructs $\exists R$, $\exists R.C$, and $\forall R.C$ appearing in the syntax of $\mathcal{SHIQ}$ are abbreviations for $R.(\neg(A \sqcap \neg A)$, $\neg(\leqslant 0R.C)$, and $\neg(\exists R.\neg C)$, respectively. However, we have included them explicitly because they appear in some of the sub-logics of $\mathcal{SHIQ}$ that we consider in the following, and in these logics they cannot be expressed by the other constructs.

Also, notice that in a DL with union ($\sqcup$), negation ($\neg$), and qualified number restriction, the TBox assertion (funct R) can be expressed as

$$A \sqcup \neg A \sqsubseteq \, \leqslant 1R.(A \sqcup \neg A).$$

Again, we have explicitly considered such kind of assertions because in some DLs they cannot be expressed by the other constructs.

To specify the semantics of the constructs used in $\mathcal{SHIQ}$, we illustrate in Fig. 1 how an interpretation $\mathcal{I}$ interprets the various concept and role expressions allowed in such a DL (including concept expressions in abbreviated forms). Notice that Fig. 1 specifies how complex concepts and roles should be interpreted starting from the interpretation of atomic concepts and roles.

We now turn our attention to the notion of satisfaction of TBox assertions. An interpretation $\mathcal{I}$ *satisfies* a concept inclusion assertion of the form $C_l \sqsubseteq C_r$, if $C_l^{\mathcal{I}} \subseteq C_r^{\mathcal{I}}$, a role inclusion assertion of the form $R_l \sqsubseteq R_r$, if $R_l^{\mathcal{I}} \subseteq R_r^{\mathcal{I}}$, a role functionality assertion of the form (funct R) if for every $o_1, o_2, o_3 \in \Delta^{\mathcal{I}}$, $(o_1, o_2) \in R^{\mathcal{I}}$ and $(o_1, o_3) \in R^{\mathcal{I}}$ implies $o_2 = o_3$, and a role transitivity assertion of the form (trans R) if $R^{\mathcal{I}}$ is a transitive relation.

Due to its expressive power, reasoning in $\mathcal{SHIQ}$ is computationally expensive. For example, checking whether a TBox assertion is logically implied by a given TBox is an ExpTime-hard problem. For this reason, several sub-logics have been studied with the goal of singling out DLs with more appealing computational properties of TBox reasoning. Essentially, each sub-logic of $\mathcal{SHIQ}$ rules out some of the assertions that are expressible in $\mathcal{SHIQ}$.

One prominent example of such sub-logics is the DL $\mathcal{AL}$ [22]. Based on the specification of $\mathcal{SHIQ}$ that we have provided above, $\mathcal{AL}$ can be defined as follows:

$$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$$

$$P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$$

$$\neg C^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$$

$$(C \sqcap C')^{\mathcal{I}} = C^{\mathcal{I}} \cap C'^{\mathcal{I}}$$

$$(\exists R)^{\mathcal{I}} = \{o \mid \exists o'.(o, o') \in R^{\mathcal{I}}\}$$

$$(\exists R.C)^{\mathcal{I}} = \{o \mid \exists o'.(o, o') \in R^{\mathcal{I}} \wedge o' \in C^{\mathcal{I}}\}$$

$$(\forall R.C)^{\mathcal{I}} = \{o \mid \forall o'.(o, o') \in R^{\mathcal{I}} \supset o' \in C^{\mathcal{I}}\}$$

$$(\leqslant nR.C)^{\mathcal{I}} = \{o \mid |\{o' \in C^{\mathcal{I}} \mid (o, o') \in Q^{\mathcal{I}}\}| \leqslant n\}$$

$$(P^-)^{\mathcal{I}} = \{(o, o') \mid (o', o) \in P^{\mathcal{I}}\}$$

Fig. 1. Semantics of $\mathcal{SHIQ}$ concepts and roles in an interpretation $\mathcal{I}$.

- Role expressions do not use the inverse of roles, whereas concept expressions are built by using only the following constructs: A, $C \sqcap C'$, $\exists R$, $\forall R.C$, and $\neg A$ (i.e., in $\mathcal{AL}$ negation can only be applied to atomic concepts).
- TBox assertions may only be concept inclusion assertions, whereas ABox assertions are those allowed in $\mathcal{SHIQ}$.

Observe that, since $\mathcal{SHIQ}$ and $\mathcal{AL}$ allow for inequality assertions in the ABox, in these DLs we can enforce the UNA on an ontology through suitable ABox assertions.

Although the limited expressiveness of $\mathcal{AL}$ has an impact on TBox reasoning, both $\mathcal{SHIQ}$ and $\mathcal{AL}$ can be regarded as expressive DLs, and their expressive power comes at a price. Indeed, in the DLs ranging from $\mathcal{AL}$ to $\mathcal{SHIQ}$, answering conjunctive queries is computationally expensive (coNP-complete in data complexity, see later), both with and without UNA.

Other DLs have been proposed in the recent years that are less expressive than these logics, but admit computationally tractable query answering. Two families of DLs with these characteristics are the $\mathcal{EL}$ family, and the *DL-Lite* family.

Among the DLs of the $\mathcal{EL}$ family [14,15], in this article we concentrate on two DLs, namely $\mathcal{EL}$ and $\mathcal{ELH}$, whose characteristics are as follows:

- In both $\mathcal{EL}$ and $\mathcal{ELH}$, role expressions do not use the inverse of roles, whereas concept expressions are built by using only the following constructs: A, $C \sqcap C$, $\exists R$, and $\exists R.C$. Note that these DLs allow for no form of negation and no form of universal quantification.
- TBox assertions in $\mathcal{EL}$ may only be concept inclusion assertions.
- TBox assertions in $\mathcal{ELH}$ may only be concept or role inclusion assertions.
- In both DLs, inequality assertions are not allowed in the ABox.

Note that, since $\mathcal{EL}$ and $\mathcal{ELH}$ include neither functionality assertions nor inequalities, the UNA is immaterial for these logics, in the sense that from every model of an ontology $\mathcal{K}$ we can build a UNA-interpretation that satisfies all the assertions of $\mathcal{K}$. It follows that computing the certain answers with UNA is the same as computing the certain answers without UNA.

The *DL-Lite* family [12,23] is a family of tractable DLs particularly suited for dealing with ontologies with very large ABoxes, which can be managed through relational database technology. In this article, we concentrate on two DLs of this family, namely *DL-Lite$_F$* and *DL-Lite$_R$*, which are characterized as follows:

- Concept expressions in both *DL-Lite$_F$* and *DL-Lite$_R$* are built by using only the constructs A, $\exists R$ (which are used in both C_l and C_r), and the constructs $\neg A$, $\neg \exists R$ (which are used only in C_r), whereas role expressions include both atomic roles and the inverse of roles.
- TBox assertions in *DL-Lite$_F$* may only be concept inclusion or role functionality assertions, whereas TBox assertions in *DL-Lite$_R$* may only be concept inclusion or role inclusion assertions.
- In both DLs, inequality assertions are not allowed in the ABox.

Note that, since *DL-Lite$_F$* allows for functionality assertions, there is a difference in this DL between the case with UNA and the case without UNA. On the contrary, as for $\mathcal{EL}$ and $\mathcal{ELH}$, the UNA is immaterial for *DL-Lite$_R$*.

Table 1 summarizes the various characteristics of the DLs discussed in this article, including the complexity of conjunctive query answering both with and without UNA.

We have included in the table the references to the articles where the respective upper and/or lower bounds are shown. When the results follow directly from those of other entries in the table, we have included no reference. In particular, the coNP upper bound for $\mathcal{SHIQ}$ (and therefore for $\mathcal{AL}$ too) with UNA follows from the corresponding result without UNA described in [24,19]. The coNP lower bound derives from the lower bound for $\mathcal{AL}$ [23], which holds both with and without UNA. The PTime upper bound for $\mathcal{ELH}$ (and therefore for $\mathcal{EL}$ too) without UNA derives from the corresponding result with

Table 1
The DLs considered in this article with the data complexity of conjunctive query answering.

DL	C_l	C_r	R	TBox	With UNA	Without UNA
$\mathcal{SHIQ}$	$A \mid C \sqcap C \mid \neg C \mid \exists R \mid \exists R.C \mid$ $\forall R.C \mid \leqslant nR.C$	C_l	$P \mid P^-$	$C_l \sqsubseteq C_r$ $R_l \sqsubseteq R_r$ (funct R) (trans R)	coNP-complete	coNP-complete [24,19]
$\mathcal{AL}$	$A \mid C \sqcap C \mid \neg A \mid \exists R \mid \forall R.C$	C_l	P	$C_l \sqsubseteq C_r$	coNP-complete [23]	coNP-complete
$\mathcal{ELH}$	$A \mid C \sqcap C \mid \exists R \mid \exists R.C$	C_l	P	$C_l \sqsubseteq C_r$ $R_l \sqsubseteq R_r$	PTime-complete [15,16]	PTime-complete
$\mathcal{EL}$	$A \mid C \sqcap C \mid \exists R \mid \exists R.C$	C_l	P	$C_l \sqsubseteq C_r$	PTime-complete [23]	PTime-complete
$DL\text{-}Lite_F$	$A \mid \exists R$	$C_l \mid \neg C_l$	$P \mid P^-$	$C_l \sqsubseteq C_r$ (funct R)	in AC^0 [12]	PTime-complete [25]
$DL\text{-}Lite_R$	$A \mid \exists R$	$C_l \mid \neg C_l$	$P \mid P^-$	$C_l \sqsubseteq C_r$ $R_l \sqsubseteq R_r$	in AC^0 [12]	in AC^0

UNA [15,16]. The PTime lower bound of $\mathcal{EL}$ without UNA follows from the result with UNA reported in [23]. Finally, the upper bound for $DL\text{-}Lite_R$ without UNA directly follows from the corresponding upper bound with UNA.

3. A framework for view-based query answering

In this section we describe a formal framework for view-based query answering over DL ontologies. In the first subsection we present the basic definitions, and in the second subsection we illustrate a relevant example of usage of the framework, related to the notion of privacy-aware access to ontologies.

3.1. Basic definitions

In view-based query answering, the answer to a query posed to an ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is computed solely on the basis of the knowledge on a *finite sequence* $V = \langle v_1, \ldots, v_n \rangle$ *of views* over the ontology. In this article we consider conjunctive views, and therefore the definition v_i of each view is a CQ of the form

$$v_i(\vec{x}_i) \leftarrow conj_i(\vec{x}_i, \vec{y}_i),$$

where each v_i is a symbol that is not part of the signature $\mathcal{S}$, and $v_i \neq v_j$ for $i \neq j$. In the following, if no confusion arises, we use v_i both for the symbol denoting the view, and for the CQ constituting its definition.

Given $V = \langle v_1, \ldots, v_n \rangle$, we call V-*extension* any sequence $E = \langle e_1, \ldots, e_n \rangle$, where each e_i is a finite set of tuples of constants, of the same arity as v_i. Intuitively, the V-extension specifies which are the answers to the queries corresponding to the views. Since an ontology is characterized by a set of models, we are interested in those extensions of the views V that correspond to the certain answers of the queries $\langle v_1, \ldots, v_n \rangle$ with respect to such a set of models.

Formally, if $\mathcal{W}$ is a set of interpretations for the signature $\mathcal{S}$, the *certain extension* of V with respect to $\mathcal{W}$, denoted $cert(V, \mathcal{W})$, is the sequence $\langle e_1, \ldots, e_n \rangle$ where $e_i = cert(v_i, \mathcal{W})$. Obviously, when we consider the problem of computing the view-based answers to queries posed to an ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ without the UNA, the extension of the views V that is of interest is $cert(V, Mod(\mathcal{K}))$, i.e., the certain extension of V with respect to the set of models of the ontology $\mathcal{K}$. Analogously, when we consider the problem of computing the view-based answers to queries posed to an ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ with the UNA, the extension of the views V that is of interest is $cert(V, Mod^{UNA}(\mathcal{K}))$.

In what follows, we call $cert(V, Mod(\mathcal{K}))$ and $cert(V, Mod^{UNA}(\mathcal{K}))$ respectively the V-*extension for $\mathcal{K}$ without UNA* and the V-*extension for $\mathcal{K}$ with UNA*. Informally speaking, we base our framework on the following characterizing elements.

- Users pose CQs to the information system, whose knowledge is represented by an ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ expressed in a given DL $\mathcal{L}$ over a signature $\mathcal{S}$.
- The system associates to each user (or, class of users) some views V, which are CQs over $\mathcal{S}$, whose extension E coincides with the V-extension for $\mathcal{K}$.
- The system answers user queries *faithfully*, i.e., coherently with $\mathcal{K}$, though hiding information not implied by the views V and extensions E. This idea is captured by
  – introducing the notion of *view-based query answering setting*, or simply *vba-setting*, in a DL $\mathcal{L}$ as a triple $\langle \mathcal{T}, V, E \rangle$, where $\mathcal{T}$ is a TBox in $\mathcal{L}$, V is a collection of views, and E is a V-extension, and
  – grounding the semantics of view-based query answering on the notion of solution for $\langle \mathcal{T}, V, E \rangle$.
  Intuitively, a solution for $\langle \mathcal{T}, V, E \rangle$ is a set of interpretations over $\mathcal{S}$ that are models of $\mathcal{T}$, and such that the certain extension of the views V with respect to such a set coincides with E. With the notion of solution in place, we can define the set of answers that the system provides to a user query q as the set of tuples that are certain over every solution for $\langle \mathcal{T}, V, E \rangle$.

We now formally define the notions that we have introduced with the above observations. In particular, in this work we refer to two notions of solution of a vba-setting, called model-centered and TBox-centered solution, respectively.

Definition 5 *(Model-centered solution).* A *model-centered solution without UNA*, or simply *M*-solution, for a vba-setting $\langle \mathcal{T}, V, E \rangle$ is a set of interpretations $\mathcal{W}$ such that $\mathcal{W} \subseteq Mod(\mathcal{T})$ and $cert(V, \mathcal{W}) = E$. An *M-solution with UNA* for $\langle \mathcal{T}, V, E \rangle$ is a set of interpretations $\mathcal{W}$ such that $\mathcal{W} \subseteq Mod^{UNA}(\mathcal{T})$ and $cert(V, \mathcal{W}) = E$.

Definition 6 *(TBox-centered solution).* A *TBox-centered solution without UNA*, or simply *T*-solution, for a vba-setting $\langle \mathcal{T}, V, E \rangle$ is an *M*-solution $\mathcal{W}$ for $\langle \mathcal{T}, V, E \rangle$ such that there exists an ABox $\mathcal{A}$ expressed in $\mathcal{L}$ with $\mathcal{W} = Mod(\langle \mathcal{T}, \mathcal{A} \rangle)$. A *T-solution with UNA* for $\langle \mathcal{T}, V, E \rangle$ is an *M*-solution $\mathcal{W}$ for $\langle \mathcal{T}, V, E \rangle$ such that there exists an ABox $\mathcal{A}$ expressed in $\mathcal{L}$ with $\mathcal{W} = Mod^{UNA}(\langle \mathcal{T}, \mathcal{A} \rangle)$.

The two notions of solution give rise to two definitions of semantics for view-based query answering, called *model-centered* and *TBox-centered* semantics, respectively.

Observe that, as an immediate consequence of the above definitions, every *T*-solution for $\langle \mathcal{T}, V, E \rangle$ is also an *M*-solution for $\langle \mathcal{T}, V, E \rangle$, both without and with UNA. However, not all *M*-solutions are *T*-solutions. Indeed, while an *M*-solution is simply a set of interpretations that are coherent with $\mathcal{T}$, V, and E, the additional condition for an *M*-solution to be a *T*-solution is that it is captured by an ontology of the form $\langle \mathcal{T}, \mathcal{A} \rangle$ in $\mathcal{L}$.

Also, it is easy to see that, for a vba-setting $\langle \mathcal{T}, V, E \rangle$, if E is the result of computing the certain extension of the views V for an ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, then $\langle \mathcal{T}, V, E \rangle$ has at least one solution. Indeed, in this case, $Mod(\mathcal{K})$ is both an *M*-solution and a *T*-solution for $\langle \mathcal{T}, V, E \rangle$. On the contrary, if we start with an arbitrary $\langle \mathcal{T}, V, E \rangle$ such that E has not necessarily been computed as certain extension of the views V for an ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, then it may happen that no solution (neither *M* nor *T*) exists for the given vba-setting.

Example 7. Let $\mathcal{L}$ be *DL-Lite$_R$*, and consider the *DL-Lite$_R$* ontology $\mathcal{K}$ defined over a signature with concepts *House*, *HouseOwner*, and roles *Owns*, *OwnsHouse*, *Located*, where the intuitive meaning of these roles is as follows: *Owns*(a, b) means that a owns b, *OwnsHouse*(a, b) means that a owns the house b, and *Located*(x, y) means that x is located in y. The ontology $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is constituted by the following TBox $\mathcal{T}$:

$$OwnsHouse \sqsubseteq Owns$$

$$HouseOwner \sqsubseteq \exists OwnsHouse$$

$$\exists OwnsHouse \sqsubseteq HouseOwner$$

$$\exists OwnsHouse^- \sqsubseteq House,$$

and the following ABox $\mathcal{A}$:

$$\{OwnsHouse(john, h55), \; Located(h55, london)\}.$$

Now, consider the following views $V = \langle v_1, v_2 \rangle$, where:

$$v_1(x, y) \leftarrow House(x), Located(x, y)$$

$$v_2(x, y) \leftarrow OwnsHouse(x, z), Located(z, y).$$

The resulting V-extension $E = \langle e_1, e_2 \rangle$ for $\mathcal{K}$ is

$$e_1 = \{\langle h55, london \rangle\}$$

$$e_2 = \{\langle john, london \rangle\}.$$

An obvious *M*-solution for $\langle \mathcal{T}, V, E \rangle$ is $Mod(\mathcal{K})$. Another *M*-solution can be obtained as follows: Let S be the set of models obtained by adding to $Mod(\mathcal{K})$ the model m_1 of $\mathcal{T}$ satisfying exactly the following facts:

HouseOwner(*john*)

OwnsHouse(*john*, *h100*)

Owns(*john*, *h100*)

House(*h100*)

Located(*h100*, *london*)

House(*h55*)

Located(*h55*, *london*).

Note that *OwnsHouse*(*john*, *h55*) is false in m_1. It is not hard to see that S too is an *M*-solution of $\langle \mathcal{T}, V, E \rangle$. On the other hand, we now show that S is not a *T*-solution of $\langle \mathcal{T}, V, E \rangle$. Indeed, since $\langle john, london \rangle$ is a certain answer to v_2, one can

verify that, due to the characteristics of *DL-Lite$_R$*, if S' is a T-solution of $\langle \mathcal{T}, V, E \rangle$, then there must exist an individual a such that every model in S' satisfies both *OwnsHouse*($john, a$), and *Located*($a, london$). Since $\langle h55, london \rangle$ is the only certain answer to v_1, it follows that $h55$ is the only individual x such that *House*(x), and *Located*($x, london$) are satisfied by every model in S'. Since m_1 is a model in S where *OwnsHouse*($john, h55$) is false, it follows that there is no ABox $\mathcal{A}'$ such that $Mod(\langle \mathcal{T}, \mathcal{A}' \rangle) = S$, and therefore S is not a T-solution of $\langle \mathcal{T}, V, E \rangle$.

We are now ready to formally introduce the notion of view-based query answering in both semantics. Intuitively, while the certain answers with respect to an ontology are the tuples satisfying the query in every model of the ontology, in view-based query answering they are those tuples satisfying the query in every solution. In the following, σ stands for either M (for model-centered), or T (for TBox-centered), thus referring to one of the two semantics defined above.

Definition 8 *(View-based answer).* The set of *view-based answers without UNA* (or simply the view-based answers) to a query q with respect to $\langle \mathcal{T}, V, E \rangle$ under the σ-centered semantics, denoted by $vba_\sigma(q, \mathcal{T}, V, E)$, is the set of tuples $\vec{t}$ such that $\vec{t} \in cert(q, \mathcal{W})$ for every σ-solution $\mathcal{W}$ without UNA for $\langle \mathcal{T}, V, E \rangle$. The set of *view-based answers with UNA* to a query q with respect to $\langle \mathcal{T}, V, E \rangle$ under the σ-centered semantics, denoted by $vba_\sigma^{UNA}(q, \mathcal{T}, V, E)$, is the set of tuples $\vec{t}$ such that $\vec{t} \in cert(q, \mathcal{W})$ for every σ-solution $\mathcal{W}$ with UNA for $\langle \mathcal{T}, V, E \rangle$.

In this article, we study the decision problem associated to computing view-based answers in a DL $\mathcal{L}$, defined as follows.

Definition 9 *(View-based query answering).* *View-based query answering without UNA (respectively, with UNA)* in a DL $\mathcal{L}$ is the following decision problem: given a TBox $\mathcal{T}$ in $\mathcal{L}$, views V, a query q, a tuple $\vec{t}$ of constants, and a V-extension E, check whether $\vec{t} \in vba_\sigma(q, \mathcal{T}, V, E)$ (respectively, $\vec{t} \in vba_\sigma^{UNA}(q, \mathcal{T}, V, E)$).

Actually, for the sake of simplicity, we will study view-based query answering under the assumption that all constants appearing in q also appear in E. However, all the results presented here hold without such an assumption as well.

Finally, since we are interested in data complexity, we will measure the complexity of the view-based query answering decision problem only with respect to the size of the V-extension E.

We end this subsection by noticing that the problem of checking the existence of a solution for $\langle \mathcal{T}, V, E \rangle$ can be reduced to view-based query answering: it is sufficient to consider a concept A' not appearing in $\mathcal{T}$, and computing the view-based answers to the boolean query $q() \leftarrow A'(x)$. It is easy to see that a σ-solution (with or without UNA) for $\langle \mathcal{T}, V, E \rangle$ exists if and only if such an answer under the σ-centered semantics (with or without UNA) is false. For this reason, we only focus on view-based query answering in the rest of this article.

3.2. Privacy-aware ontology access

We now relate the framework presented in the previous subsection to the privacy scenario illustrated in the introduction. For simplicity, we will only refer to the case without UNA, but all observations hold also for the case with UNA. In such a scenario:

- The ontology $K = \langle \mathcal{T}, \mathcal{A} \rangle$ over a signature S expressed in a given DL $\mathcal{L}$ represents the knowledge that the system has on the domain of interest.
- The views V associated to a user (or, class of users) are its *authorization views*, i.e., the V-extensions E for $\mathcal{K}$ represent the knowledge that the system is authorized to disclose to the user.
- The user posing queries to the system is aware of the signature $\mathcal{S}$, but is in principle unaware of all other aspects managed by the system (i.e., $\mathcal{L}$, $\mathcal{K}$, V, and E).
- The answers provided by the system to a user query q are those logically implied by its authorization views, i.e., the view-based answers to q with respect to $\langle \mathcal{T}, V, E \rangle$.

Observe that $\mathcal{K}$, or, more precisely, $Mod(\mathcal{K})$, is obviously a solution for $\langle \mathcal{T}, V, E \rangle$, but many other ontologies, or, more precisely, sets of models, are solutions as well. Observe also that if $\mathcal{W}_i$ and $\mathcal{W}_j$ are both solutions for $\langle \mathcal{T}, V, E \rangle$, then they are indistinguishable by means of V, because $cert(V, \mathcal{W}_i) = cert(V, \mathcal{W}_j) = E$. In other words, if the tuple $\vec{t}$ is such that $q(\vec{t})$ is logically implied by $\mathcal{K}$, view-based query answering hides this tuple from the user if there is a solution $\mathcal{W}$ for $\langle \mathcal{T}, V, E \rangle$ where $\vec{t}$ is not a certain answer to q, i.e., $\vec{t} \notin cert(V, \mathcal{W})$. In such a case, $\mathcal{W}$ is an evidence for concealing $q(\vec{t})$ from the user.

The difference in the two semantics described above, namely the model-centered, and the TBox-centered semantics, is that they allow for increasing levels of information disclosure to the user. In particular, let q be a user query, and let $q(\vec{t})$ be logically implied by $\mathcal{K}$:

1. In the model-centered semantics, $q(\vec{t})$ is returned as an answer if there is no set of models $\mathcal{W} \subseteq Mod(\mathcal{K})$ that is indistinguishable from $Mod(\mathcal{K})$ itself with respect to the authorization views V, and such that $\vec{t} \notin cert(q, \mathcal{W})$. In other words, the evidence for concealing $q(\vec{t})$ from the user can be simply any subset of the models of the TBox $\mathcal{T}$ of $\mathcal{K}$.

2. In the TBox-centered semantics, the set of models $\mathcal{W}$ forming the evidence for concealing an answer from the user must be expressible by some ABox paired to $\mathcal{T}$. Intuitively, this semantics captures the situation where the user is aware not only of the signature $\mathcal{S}$, but also of the TBox $\mathcal{T}$ of $\mathcal{K}$.[1]

We end this section with an example of view-based query answering in the context of the privacy scenario we are discussing. We remind the reader that we are considering the case without UNA.

Example 10. Let $\mathcal{L}$ be *DL-Lite$_R$*, and consider again the vba-setting $\langle \mathcal{T}, V, E \rangle$ as described in Example 7. In particular, suppose that the system can only disclose to a user u the information about where houses are located, and in which locations the various persons own their houses. This requirement can be captured by specifying $V = \langle v_1, v_2 \rangle$ as the authorization views associated to u. It is immediate to verify that *Owns*(*john*, *h55*) is concealed from user u under the model-centered semantics. Indeed, it is sufficient to consider the *M*-solution S of $\langle \mathcal{T}, V, E \rangle$ described in Example 7, and notice that it contains a model of $\mathcal{T}$, namely m_1, where *Owns*(*john*, *h55*) is false. This implies that $\langle john, h55 \rangle$ is not a certain answer for S, and therefore is not a view-based answer to the query $q(x, y) \leftarrow Owns(x, y)$ with respect to $\langle \mathcal{T}, V, E \rangle$ under the model-centered semantics. On the other hand, *Owns*(*john*, *h55*) is not concealed under the TBox-centered semantics. Indeed, we have shown in Example 7 that, based on the characteristics of *DL-Lite$_R$*, if S' is *T*-solution of $\langle \mathcal{T}, V, E \rangle$, then *OwnsHouse*(*john*, *h55*) is satisfied by every model in S'. This means that, contrary to the model-centered semantics, the TBox-centered semantics allow the user u to obtain information about who is the owner of a given house.

4. Model-centered semantics

We address now view-based conjunctive query answering under the model-centered semantics in the various DLs that we have presented in Section 2.

4.1. General results

We start by stating some results on *M*-solutions that are independent of the specific DL considered. This requires the following preliminary definition. Given views $V = \langle v_1, \ldots, v_n \rangle$, where each v_i is defined as $v_i(\vec{x}_i) \leftarrow conj_i(\vec{x}_i, \vec{y}_i)$, and V-extension $E = \langle e_1, \ldots, e_n \rangle$, where each e_i is of the form $\{\vec{t}_i^1, \ldots, \vec{t}_i^{m_i}\}$, we denote by $\alpha(V, E)$ the first-order sentence

$$\alpha(V, E) = \exists_{1 \leqslant i \leqslant n, 1 \leqslant j \leqslant m_i} \big(conj_1(\vec{t}_1^1, \vec{y}_1^1) \wedge \cdots \wedge conj_1(\vec{t}_1^{m_1}, \vec{y}_1^{m_1}) \wedge \cdots$$
$$\wedge conj_n(\vec{t}_n^1, \vec{y}_n^1) \wedge \cdots \wedge conj_n(\vec{t}_n^{m_n} \vec{y}_n^{m_n}) \big).$$

We observe that $\alpha(V, E)$ is essentially a boolean query over the signature $\mathcal{S}$.

In the following results (Lemmas 11, 12, 13, and Theorem 14) we assume that the views V and the V-extension E are defined as specified above.

Lemma 11. *Every M-solution, both with and without UNA, for a vba-setting $\langle \mathcal{T}, V, E \rangle$ logically implies $\alpha(V, E)$.*

Proof. We give the proof only for the case without UNA. The case with UNA is analogous. Towards a contradiction, assume that there exists an *M*-solution $\mathcal{W}$ for $\langle \mathcal{T}, V, E \rangle$ and an interpretation $\mathcal{I} \in \mathcal{W}$ such that $\mathcal{I} \not\models \alpha(V, E)$. Then, there exists a conjunct $conj_i$ and a tuple $\vec{t}_i^j$ such that $\mathcal{I} \not\models \exists \vec{y}_i^j . conj_i(\vec{t}_i^j, \vec{y}_i^j)$. But then, considering the definition of v_i, i.e., $v_i(\vec{x}_i) \leftarrow conj_i(\vec{x}_i, \vec{y}_i)$, we have that $\vec{t}_i^j \notin e_i$. Thus, since $\mathcal{I} \in \mathcal{W}$, we have that $cert(V, \mathcal{W}) \neq E$ and hence $\mathcal{W}$ is not an *M*-solution, which is a contradiction. $\square$

In the following, we write $Mod(\mathcal{T} \cup \alpha(V, E))$ to denote $Mod(\mathcal{T}) \cap \{\mathcal{I} \mid \mathcal{I} \models \alpha(V, E)\}$, i.e., the set of interpretations that are models of $\mathcal{T}$ without UNA and that satisfy the first-order sentence $\alpha(V, E)$. Analogously, we use the notation $Mod^{UNA}(\mathcal{T} \cup \alpha(V, E))$ for the case with UNA.

Lemma 12. *$Mod(\mathcal{T} \cup \alpha(V, E))$ is an M-solution for a vba-setting $\langle \mathcal{T}, V, E \rangle$, if one such solution exists. Analogously, $Mod^{UNA}(\mathcal{T} \cup \alpha(V, E))$ is an M-solution with UNA for $\langle \mathcal{T}, V, E \rangle$, if one such solution exists.*

Proof. We give the proof only for the case without UNA. The case with UNA is analogous. Assume that there exists an *M*-solution $\mathcal{W}$ for $\langle \mathcal{T}, V, E \rangle$, and that $\mathcal{W}_\alpha = Mod(\mathcal{T} \cup \alpha(V, E))$ is not an *M*-solution for $\langle \mathcal{T}, V, E \rangle$. Then, since $\mathcal{W}_\alpha \subseteq Mod(\mathcal{T})$, we must have $cert(V, \mathcal{W}_\alpha) \neq E$. Moreover, by Lemma 11, $\mathcal{W} \subseteq \mathcal{W}_\alpha$, and $cert(V, \mathcal{W}_\alpha) \subseteq cert(V, \mathcal{W}) = E$. Hence, there exists a tuple $\vec{t}_i^j \in e_i$ such that $\vec{t}_i^j \notin cert(v_i, \mathcal{W}_\alpha)$. Hence, there exists an $\mathcal{I} \in \mathcal{W}_\alpha$ such that $\mathcal{I} \not\models \exists \vec{y}_i^j . conj_i(\vec{t}_i^j, \vec{y}_i^j)$, where the definition of v_i is $v_i(\vec{x}_i) \leftarrow conj_i(\vec{x}_i, \vec{y}_i)$. But then $\mathcal{I} \not\models \alpha(V, E)$, hence $\mathcal{I} \notin \mathcal{W}_\alpha$, which is a contradiction. $\square$

---

[1] In [20], a further semantics for view-based query answering was considered, where the user, besides $\mathcal{S}$ and $\mathcal{T}$, is aware of the DL $\mathcal{L}$ used to express the ontology.

Lemma 13. *There is an M-solution without UNA* (*respectively, with UNA*) *for a vba-setting* $\langle \mathcal{T}, V, E \rangle$ *iff* $cert(V, Mod(\mathcal{T} \cup \alpha(V, E))) = E$ (*respectively,* $cert(V, Mod^{UNA}(\mathcal{T} \cup \alpha(V, E))) = E$).

Proof. We give the proof only for the case without UNA. The case with UNA is analogous.

"$\Rightarrow$" If there is an M-solution for $\langle \mathcal{T}, V, E \rangle$, by Lemma 12, $Mod(\mathcal{T} \cup \alpha(V, E))$ is one such solution, and hence by definition $cert(V, Mod(\mathcal{T} \cup \alpha(V, E))) = E$.

"$\Leftarrow$" If $cert(V, Mod(\mathcal{T} \cup \alpha(V, E))) = E$, since $Mod(\mathcal{T} \cup \alpha(V, E))$ satisfies also the condition $Mod(\mathcal{T} \cup \alpha(V, E)) \subseteq Mod(\mathcal{T})$, it follows that $Mod(\mathcal{T} \cup \alpha(V, E))$ is an M-solution for $\langle \mathcal{T}, V, E \rangle$. $\square$

Theorem 14. *If there exists an M-solution without UNA* (*respectively, with UNA*) *for a vba-setting* $\langle \mathcal{T}, V, E \rangle$, *then* $vba_M(q, \mathcal{T}, V, E) = cert(q, Mod(\mathcal{T} \cup \alpha(V, E)))$ (*respectively,* $vba_M^{UNA}(q, \mathcal{T}, V, E) = cert(q, Mod^{UNA}(\mathcal{T} \cup \alpha(V, E)))$).

Proof. We give the proof only for the case without UNA. The case with UNA is analogous.

"$\Rightarrow$" If $\vec{t} \in vba_M(q, \mathcal{T}, V, E)$, then $\vec{t} \in cert(q, \mathcal{W})$ for every M-solution $\mathcal{W}$, including $Mod(\mathcal{T} \cup \alpha(V, E))$ by Lemma 12.

"$\Leftarrow$" It $\vec{t} \in cert(q, Mod(\mathcal{T} \cup \alpha(V, E)))$, then $\vec{t} \in cert(q, \mathcal{W})$, for every M-solution $\mathcal{W}$, since by Lemma 11, $\mathcal{W} \subseteq Mod(\mathcal{T} \cup \alpha(V, E))$. Hence $\vec{t} \in vba_M(q, \mathcal{T}, V, E)$. $\square$

Observe that in the privacy setting described in the previous section, where the V-extensions are computed from an ontology $\mathcal{K}$, an M-solution always exists, namely $Mod(\mathcal{K})$. Hence by the propositions above we get that we can compute $vba_M(q, \mathcal{T}, V, E)$ by computing $cert(q, Mod(\mathcal{T} \cup \alpha(V, E)))$, and analogously for the case with UNA.

We show now how to reduce conjunctive query answering to view-based query answering under the model-centered semantics in different cases. We first address the case of conjunctive query answering without UNA.

Theorem 15. *Let* $\mathcal{L}$ *be DL that allows for expressing functionality of atomic roles. Then, conjunctive query answering without UNA in* $\mathcal{L}$ *can be reduced in* LogSpace *in data complexity to view-based query answering both with and without UNA under the model-centered semantics in* $\mathcal{L}$.

Proof. We show the reduction to view-based query answering with UNA under the model-centered semantics. The same proof applies also for the reduction to view-based query answering without UNA. Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be an $\mathcal{L}$ ontology and $q(\vec{x}) \leftarrow conj(\vec{x}, \vec{y}, \vec{c})$ a conjunctive query over $\mathcal{K}$, in which the existential variables $\vec{y}$ and the constants $\vec{c}$ occur. We construct a vba-setting $\langle \mathcal{T}', V, E \rangle$ and a query q' as follows.

- $\mathcal{T}' = \mathcal{T} \cup \{(\text{funct } copy)\}$, where $copy$ is a fresh atomic role.
- There is one view v_A in V for each atomic concept A, with definition

$$v_A(x) \leftarrow copy(x, x'), A(x')$$

  and extension $e_A = \{a \mid A(a) \in \mathcal{A}\}$.
- There is one view v_P in V for each atomic role P, with definition

$$v_P(x_1, x_2) \leftarrow copy(x_1, x_1'), copy(x_2, x_2'), P(x_1', x_2')$$

  and extension $e_P = \{(a, b) \mid P(a, b) \in \mathcal{A}\}$.
- The query q' is defined as

$$q'(\vec{x}) \leftarrow copy(\vec{x}, \vec{x}'), copy(\vec{c}, \vec{y}_c), conj(\vec{x}', \vec{y}, \vec{y}_c),$$

  where $copy(\vec{z}, \vec{z}')$, with $\vec{z} = (z_1, \ldots, z_n)$ and $\vec{z}' = (z_1', \ldots, z_n')$, stands for $copy(z_1, z_1'), \ldots, copy(z_n, z_n')$, and $conj(\vec{x}', \vec{y}, \vec{y}_c)$ stands for the body of q in which we have substituted each distinguished variable x with its primed copy x' and each constant c with a new existential variable y_c.

We show that for each tuple $\vec{t}$ of constants we have that $\vec{t} \in cert(q, Mod(\mathcal{K}))$ iff $\vec{t} \in vba_M^{UNA}(q', \mathcal{T}', V, E)$.

"$\Rightarrow$" Let $\vec{t}$ be a tuple of constants such that $\vec{t} \in cert(q, Mod(\mathcal{K}))$ and $\vec{t} \notin vba_M^{UNA}(q', \mathcal{T}', V, E)$. Then there exists an M-solution $\mathcal{W}$ with UNA of $\langle \mathcal{T}', V, E \rangle$ and an interpretation $\mathcal{I} \in \mathcal{W}$ such that $\mathcal{I} \not\models q'(\vec{t})$. Consider the interpretation $\mathcal{J} = (\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}})$ of $\mathcal{K}$ built as follows:

- $\Delta^{\mathcal{J}} = \Delta^{\mathcal{I}}$,
- $A^{\mathcal{J}} = A^{\mathcal{I}}$, for each atomic concept A,
- $P^{\mathcal{J}} = P^{\mathcal{I}}$, for each atomic role P except $copy$, and
- $a^{\mathcal{J}} = o'$, where $(a^{\mathcal{I}}, o') \in copy^{\mathcal{I}}$.

Notice that $a^{\mathcal{J}}$ is well defined, for each constant a appearing in the ABox. Indeed, when $A(a) \in \mathcal{A}$, for some atomic concept A, then $a \in e_A = cert(v_A, \mathcal{W})$, and $\mathcal{I} \models v_A(a)$. Hence, by the definition of v_A, there is an object o' such that

$(a^{\mathcal{I}}, o') \in copy^{\mathcal{I}}$. Moreover, since (funct $copy$) $\in \mathcal{T}'$, the object o' is unique. Similarly for the case when either $P(a, b) \in \mathcal{A}$ or $P(b, a) \in \mathcal{A}$, for some constant b.

Then $\mathcal{J}$ satisfies $\mathcal{T}$ since $\mathcal{I}$ satisfies $\mathcal{T}'$ and $\mathcal{T} \subseteq \mathcal{T}'$. Moreover, for each $A(a) \in \mathcal{A}$, we have that $a^{\mathcal{J}} \in A^{\mathcal{J}}$ by construction of the definition of v_A and its extension e_A. Similarly for each $P(a, b) \in \mathcal{A}$. Then, we get that $\mathcal{J} \models q(\vec{t})$ since $\vec{t} \in cert(q, Mod(\mathcal{K}))$. But then, from the variable assignment that makes $\mathcal{J} \models q(\vec{t})$ hold, we can construct a variable assignment that makes $\mathcal{I} \models q'(\vec{t})$ hold, which is a contradiction.

"$\Leftarrow$" Let $\vec{t}$ be a tuple of constants such that $\vec{t} \in vba_M^{UNA}(q', \mathcal{T}', V, E)$ and $\vec{t} \notin cert(q, Mod(\mathcal{K}))$. Then there exists a model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \not\models q(\vec{t})$. Consider the interpretation $\mathcal{J}$ of $\mathcal{T}'$ built as follows:

- $\Delta^{\mathcal{J}} = \Delta^{\mathcal{I}} \cup \{o_a \mid a \text{ is a constant appearing in } \mathcal{A}\}$,
- $A^{\mathcal{J}} = A^{\mathcal{I}}$, for each atomic concept A,
- $P^{\mathcal{J}} = P^{\mathcal{I}}$, for each atomic role P except $copy$,
- $copy^{\mathcal{J}} = \{(o_a, a^{\mathcal{I}}) \mid a \text{ is a constant appearing in } \mathcal{A}\}$, and
- $a^{\mathcal{J}} = o_a$, for each constant a appearing in $\mathcal{A}$.

By construction $\mathcal{J}$ satisfies the UNA,[2] and since $\mathcal{I}$ is a model of $\mathcal{T}$ and $copy^{\mathcal{J}}$ is functional by construction, we have that $\mathcal{J}$ is a model of $\mathcal{T}'$. Moreover, $\mathcal{J} \models v_A(a)$, for each atomic concept A and constant a such that $A(a) \in \mathcal{A}$, and similarly for roles. Hence $\mathcal{W} = \{\mathcal{J}\}$ is an M-solution with UNA for $\langle \mathcal{T}', V, E \rangle$. Since $\mathcal{I} \not\models q(\vec{t})$, by construction $\mathcal{J} \not\models q'(\vec{t})$, which is a contradiction to $\vec{t} \in vba_M^{UNA}(q', \mathcal{T}', V, E)$. □

The next result is analogous to the one above, but for the case where the DL allows for expressing concept and role inclusions.

Theorem 16. *Let $\mathcal{L}$ be a DL that allows for expressing concept and role inclusions. Then conjunctive query answering without UNA in $\mathcal{L}$ can be reduced in* LogSpace *in data complexity to view-based query answering, both with and without UNA, under the model-centered semantics in $\mathcal{L}$.*

Proof. We give the proof for the case of view-based query answering without UNA. However, the same proof applies also to the case with UNA, provided that we consider interpretations that satisfy the UNA. Given $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, we build a vba-setting $\langle \mathcal{T}', V, E \rangle$ as follows. For each atomic concept A (respectively, atomic role P) occurring in $\mathcal{A}$:

- we introduce a new atomic concept A' (respectively, atomic role P');
- we introduce a view v_A (respectively, v_P) with view definition $v_A(x) \leftarrow A'(x)$ (respectively, $v_P(x_1, x_2) \leftarrow P'(x_1, x_2)$) and view extension $e_A = \{a \mid A(a) \in \mathcal{A}\}$ (respectively, $e_P = \{(a, b) \mid P(a, b) \in \mathcal{A}\}$).

Then $\mathcal{T}'$ consists of all inclusion assertions in $\mathcal{T}$ and a new inclusion assertion $A' \sqsubseteq A$ (respectively, $P' \sqsubseteq P$) for each atomic concept A' (respectively, atomic role P') introduced above.

Then, for every conjunctive query q over $\mathcal{K}$ we have that $cert(q, Mod(\mathcal{K})) = vba_M(q, \mathcal{T}', V, E)$. Indeed let us assume that there exists a tuple $\vec{t}$ of constants such that $\vec{t} \in cert(q, Mod(\mathcal{K}))$ and $\vec{t} \notin vba_M(q, \mathcal{T}', V, E)$. Then there exists an M-solution $\mathcal{W}$ for $\langle \mathcal{T}', V, E \rangle$ and an interpretation $\mathcal{I} \in \mathcal{W}$ such that $\mathcal{I} \not\models q(\vec{t})$. But $\mathcal{I} \models \mathcal{T}$, since $\mathcal{T} \subseteq \mathcal{T}'$, and $\mathcal{I} \models \mathcal{A}$, considering the construction of the view extensions and the new inclusion assertions in $\mathcal{T}'$. Hence, $\vec{t} \notin cert(q, Mod(\mathcal{K}))$, which is a contradiction. Conversely, let us assume that there exists a tuple $\vec{t} \in vba_M(q, \mathcal{T}', V, E)$ such that $\vec{t} \notin cert(q, Mod(\mathcal{K}))$. Then there exists a model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \not\models q(\vec{t})$. Consider the interpretation $\mathcal{I}'$ obtained from $\mathcal{I}$ by interpreting each new symbol A' (respectively, P') as $A'^{\mathcal{I}'} = \{a^{\mathcal{I}} \mid A(a) \in \mathcal{A}\}$ (respectively, $P'^{\mathcal{I}'} = \{(a^{\mathcal{I}}, b^{\mathcal{I}}) \mid P(a, b) \in \mathcal{A}\}$). Then $\mathcal{W} = \{\mathcal{I}'\}$ is an M-solution for $\langle \mathcal{T}', V, E \rangle$ but $\vec{t} \notin cert(q, \mathcal{W})$, which contradicts $\vec{t} \in vba_M(q, \mathcal{T}', V, E)$. □

Note that the above result holds also for those DLs that do not allow for expressing role inclusions, provided that the DL is such that we cannot derive new facts about roles except for the ones explicitly stated in the ABox. Indeed, we can modify the construction of $\langle \mathcal{T}', V, E \rangle$ given in the proof above, by *not* introducing in $\mathcal{T}'$ new atomic roles (although we still introduce new atomic concepts), and defining for each atomic role P in $\mathcal{T}$ the view v_P as $v_P(x_1, x_2) \leftarrow P(x_1, x_2)$ (the view extensions are defined as above). We exploit now the fact that we are considering a DL where we cannot derive new facts about roles besides those in the ABox, and conclude that, if there exists a model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \not\models q(\vec{t})$, then there exists also a model $\mathcal{I}_0$ of $\mathcal{K}$ such that $\mathcal{I}_0 \not\models q(\vec{t})$ and for which $P^{\mathcal{I}_0} = \{(a^{\mathcal{I}_0}, b^{\mathcal{I}_0}) \mid P(a, b) \in \mathcal{A}\}$. Consider the interpretation $\mathcal{I}_0'$ obtained from $\mathcal{I}_0$ by interpreting each new symbol A' as $A'^{\mathcal{I}'} = \{a^{\mathcal{I}} \mid A(a) \in \mathcal{A}\}$. As above, $\mathcal{W} = \{\mathcal{I}_0'\}$ is an M-solution for $\langle \mathcal{T}', V, E \rangle$ but $\vec{t} \notin cert(q, \mathcal{W})$, which gives us a contradiction.

As an immediate consequence of the above theorem, we get that, in DLs that allow for expressing concept and role inclusions, if conjunctive query answering in $\mathcal{L}$ is undecidable, then view-based query answering under the model-centered semantics in $\mathcal{L}$ is undecidable.

---

[2] Note that, if $a^{\mathcal{I}} = b^{\mathcal{I}} = o$, then $copy^{\mathcal{J}}$ maps both o_a and o_b to the same object o.

The next theorem complements the ones above, by giving a general result on how to transfer algorithms and data complexity lower bounds from conjunctive query answering without UNA to view-based query answering without UNA under the model-centered semantics.

Theorem 17. *Let $\mathcal{L}$ be a DL for which the data complexity of conjunctive query answering without UNA is in a complexity class $\mathcal{C}$ that is either* PTIME *or above. Then, the data complexity of view-based query answering without UNA under the model-centered semantics in $\mathcal{L}$ is in $\mathcal{C}$.*

Proof. Let $\langle \mathcal{T}, V, E \rangle$ be a vba-setting, and suppose we have to check whether $\vec{t} \in vba_M(q, \mathcal{T}, V, E)$. By Lemma 13 and Theorem 14, $\vec{t} \in vba_M(q, \mathcal{T}, V, E)$ if and only if either $cert(V, Mod(\mathcal{T} \cup \alpha(V, E))) \neq E$, or $\vec{t} \in cert(q, Mod(\mathcal{T} \cup \alpha(V, E)))$.

We first observe that, for each view v_i with extension e_i, checking whether $cert(v_i, Mod(\mathcal{T} \cup \alpha(V, E))) \neq e_i$ can be done by considering each tuple $\vec{t}\,'$ of the arity of v_i constructed from constants appearing in E, and verifying whether $\vec{t}\,' \in cert(v_i, Mod(\mathcal{T} \cup \alpha(V, E)))$ iff $\vec{t}\,' \notin e_i$. It remains to show that, for a conjunctive query q' and a tuple $\vec{t}\,'$, checking whether $\vec{t}\,' \in cert(q', Mod(\mathcal{T} \cup \alpha(V, E)))$ can be done with the same complexity as query answering in $\mathcal{L}$. For this purpose, we observe that we can construct in PTIME an ABox $\mathcal{A}_{\alpha(V,E)}$ by constructing $\alpha(V, E)$, substituting each variable y in $\alpha(V, E)$ with a fresh constant a_y, and introducing one ABox assertion for each atom in $\alpha(V, E)$. The claim follows from showing that $\vec{t}\,' \in cert(q', Mod(\mathcal{T} \cup \alpha(V, E)))$ iff $\vec{t}\,' \in cert(q', Mod(\langle \mathcal{T}, \mathcal{A}_{\alpha(V,E)} \rangle))$.

"$\Rightarrow$" By contradiction, assume that there exists a model $\mathcal{I}$ of $\langle \mathcal{T}, \mathcal{A}_{\alpha(V,E)} \rangle$ such that $\mathcal{I} \not\models q'(\vec{t}\,')$. Then, $\mathcal{I} \models \alpha(V, E)$ by assigning to each existential variable y the object $a_y^{\mathcal{I}}$. Hence, $\vec{t}\,' \notin cert(q', Mod(\mathcal{T} \cup \alpha(V, E)))$, which is a contradiction.

"$\Leftarrow$" By contradiction, assume that there exists an interpretation $\mathcal{I} \in Mod(\mathcal{T} \cup \alpha(V, E))$ such that $\mathcal{I} \not\models q'(\vec{t}\,')$. Then, we can extend $\mathcal{I}$ to an interpretation $\mathcal{I}'$ by interpreting each new constant a_y as the value assigned to the variable y that makes $\mathcal{I} \models \alpha(V, E)$ hold. We have that $\mathcal{I}' \models \mathcal{A}_{\alpha(V,E)}$ and hence it is a model of $\langle \mathcal{T}, \mathcal{A}_{\alpha(V,E)} \rangle$ such that $\mathcal{I}' \not\models q'(\vec{t}\,')$, which is a contradiction to $\vec{t}\,' \in cert(q', Mod(\langle \mathcal{T}, \mathcal{A}_{\alpha(V,E)} \rangle))$.   □

4.2. Results for specific Description Logics

We now consider view-based query answering under the model-centered semantics for the specific DLs introduced in Section 2.

We start with the DLs of the *DL-Lite* family, and consider first *DL-Lite$_R$*. Since in *DL-Lite$_R$* there is no possibility of forcing two constants to be interpreted as the same object, the presence or absence of the UNA is immaterial.

Theorem 18. *View-based query answering, both with and without UNA, under the model-centered semantics in DL-Lite$_R$ is in* AC^0 *with respect to data complexity.*

Proof. Since in *DL-Lite$_R$* the UNA is immaterial, we prove the claim for the case without UNA. To do so, we reduce view-based query answering without UNA to first-order query evaluation, which is in AC^0 with respect to data complexity [21]. Let $\langle \mathcal{T}, V, E \rangle$ be a vba-setting with $E = \langle e_1, \ldots, e_n \rangle$, and suppose we have to check whether $\vec{t} \in vba_M(q, \mathcal{T}, V, E)$. By Lemma 13 and Theorem 14, $\vec{t} \in vba_M(q, \mathcal{T}, V, E)$ if and only if either $cert(V, Mod(\mathcal{T} \cup \alpha(V, E))) \neq E$, or $\vec{t} \in cert(q, Mod(\mathcal{T} \cup \alpha(V, E)))$. To check this condition, we construct a first-order query φ and evaluate it over the relational structure $\bar{E}$ with predicates $\bar{e}_1, \ldots, \bar{e}_n$, whose extensions are $e_1, \ldots, e_n$, respectively. To define φ, we use the following two notions: the perfect reformulation of a conjunctive query q' with respect to a *DL-Lite$_R$* TBox $\mathcal{T}$ [12], denoted $\mathsf{PerfectRef}(q', \mathcal{T})$, and the rewriting of a union of conjunctive queries q'' with respect to a set V of conjunctive views [7], denoted $\mathsf{Rew}_V(q'')$. It is known that $\mathsf{PerfectRef}(q', \mathcal{T})$ is a union of conjunctive queries [12], and that $\mathsf{Rew}_V(q'')$ is also a union of conjunctive queries [7]. Now let φ be the boolean query

$$\left(\bigvee_{1 \leqslant i \leqslant n} \exists \vec{x} \big(\neg \bar{e}_i(\vec{x}) \wedge \mathsf{Rew}_V\big(\mathsf{PerfectRef}(v_i, \mathcal{T})\big)(\vec{x}) \big) \right) \vee \mathsf{Rew}_V\big(\mathsf{PerfectRef}(q, \mathcal{T})\big)(\vec{t}),$$

where $q'(\vec{x})$ denotes a query q' whose free variables are $\vec{x}$, and $q'(\vec{t})$ denotes the boolean query obtained from $q'(\vec{x})$ by substituting the free variables $\vec{x}$ with the tuple of constants $\vec{t}$. It can be verified that the first disjunct of φ takes care of checking whether $cert(V, Mod(\mathcal{T} \cup \alpha(V, E))) \neq E$, while the second disjunct checks whether $\vec{t} \in cert(q, Mod(\mathcal{T} \cup \alpha(V, E)))$. Therefore, $\vec{t} \in vba_M(q, \mathcal{T}, V, E)$ if and only if evaluating φ over $\bar{E}$ returns true. Since such an evaluation can be done in AC^0 with respect to data complexity, the claim follows.   □

Next we turn to *DL-Lite$_F$*, for which UNA matters, since through functionality assertions one can force two objects to coincide, possibly generating an inconsistency under UNA, if the two objects are denoted by two different constants.

Theorem 19. *View-based query answering, both with and without UNA, under the model-centered semantics in DL-Lite$_F$ is* PTIME-*complete with respect to data complexity.*

Proof. Conjunctive query answering without UNA in *DL-Lite$_F$* is PTime-complete [25]. Hence, by applying Theorem 15, we get the PTime lower bound for the two cases without and with UNA.

Membership in PTime of view-based query answering without UNA is a direct consequence of Theorem 17. For the case with UNA, we can use the following algorithm to decide whether $\vec{t} \in vba_M^{UNA}(q, \mathcal{T}, V, E)$.

1. We construct in PTime the set of atoms constituted by the conjuncts of $\alpha(V, E)$ (note that they contain both constants of E and variables). Then, we chase such atoms according to the functionality assertions in $\mathcal{T}$ [26], thus propagating equality. Let $\beta(V, E)$ be the resulting set of atoms.
2. If in the process of building $\beta(V, E)$ two constants are equated, this means that $Mod^{UNA}(\mathcal{T} \cup \alpha(V, E))$ is empty, and therefore, we return true because, by definition, $\vec{t} \in vba_M^{UNA}(q, \mathcal{T}, V, E)$.
3. Otherwise, by Lemma 13, we check whether $cert(V, Mod^{UNA}(\mathcal{T} \cup \alpha(V, E))) = E$, by constructing, similarly to what done in the proof of Theorem 17, the ABox $\mathcal{A}_{\beta(V,E)}$, computing $cert(V, Mod(\langle \mathcal{T}, \mathcal{A}_{\beta(V,E)} \rangle))$ [25], and checking whether the result projected on the constants of E equals E. It is easy to see that all three steps can be done in PTime. If this is not the case we return true.
4. Otherwise, we return the result of checking whether $\vec{t} \in cert(q, Mod^{UNA}(\mathcal{T} \cup \beta(V, E)))$, which again can be done in PTime [25] by resorting to $\mathcal{A}_{\beta(V,E)}$, analogously to the proof of Theorem 17. □

We now consider the DLs $\mathcal{EL}$ and $\mathcal{ELH}$.

Theorem 20. *View-based query answering, both with and without UNA, under the model-centered semantics in $\mathcal{EL}$ and $\mathcal{ELH}$ is PTime-complete with respect to data complexity.*

Proof. First, we recall that UNA is immaterial for $\mathcal{EL}$ and $\mathcal{ELH}$. Hence, by considering the results in Table 1 on PTime-completeness of conjunctive query answering in $\mathcal{EL}$ and $\mathcal{ELH}$, by applying Theorem 17, we obtain the PTime upper bound, and by applying Theorem 15, we obtain the PTime lower bound. □

Finally, we examine view-based query answering under the model-centered semantics in the DLs ranging from $\mathcal{AL}$ to $\mathcal{SHIQ}$.

Theorem 21. *View-based query answering, both with and without UNA, under the model-centered semantics in $\mathcal{AL}$ is coNP-hard with respect to data complexity.*

Proof. First, we recall that UNA is immaterial for $\mathcal{AL}$. Hence, by considering the results in Table 1 on coNP-hardness of conjunctive query answering in $\mathcal{AL}$, by applying Theorem 15, we obtain the coNP lower bound. □

As for the upper bound, we show next membership in coNP for $\mathcal{SHIQ}$, which is the most expressive DL considered in this paper.

Theorem 22. *View-based query answering, both with and without UNA, under the model-centered semantics in $\mathcal{SHIQ}$ is in coNP with respect to data complexity.*

Proof. In $\mathcal{SHIQ}$ we can polynomially encode UNA by expressing inequalities between constants in the ABox. Hence, it suffices to show the result for view-based query answering without UNA in $\mathcal{SHIQ}$. For this case, the claim follows directly from the coNP upper bound of conjunctive query answering without UNA in $\mathcal{SHIQ}$ in Table 1, by applying Theorem 17. □

5. TBox-centered semantics

In this section we analyze view-based query answering under the TBox-centered semantics. We first provide some general results, then we study the computational properties of the problem in all the specific DLs that we have considered so far.

5.1. General results

First, we show a sufficient condition under which view-based query answering under the model-centered semantics and view-based query answering under the TBox-centered semantics coincide.

Theorem 23. *If $Mod(\mathcal{T} \cup \alpha(V, E))$ is a T-solution for a vba-setting $\langle \mathcal{T}, V, E \rangle$, then $vba_T(q, \mathcal{T}, V, E) = vba_M(q, \mathcal{T}, V, E)$. Analogously, if $Mod^{UNA}(\mathcal{T} \cup \alpha(V, E))$ is a T-solution with UNA for $\langle \mathcal{T}, V, E \rangle$, then $vba_T^{UNA}(q, \mathcal{T}, V, E) = vba_M^{UNA}(q, \mathcal{T}, V, E)$.*

Proof. We give the proof only for the case without UNA. The case with UNA is analogous. Since by Lemma 11 every M-solution for $\langle \mathcal{T}, V, E \rangle$ logically implies $\alpha(V, E)$, it follows that every M-solution (and thus every T-solution) is contained in $Mod(\mathcal{T} \cup \alpha(V, E))$. Moreover, since every T-solution is also an M-solution, $Mod(\mathcal{T} \cup \alpha(V, E))$ is an M-solution, which immediately implies that both $vba_T(q, \mathcal{T}, V, E)$ and $vba_M(q, \mathcal{T}, V, E)$ are equal to the certain answers of q computed over the set of interpretations $Mod(\mathcal{T} \cup \alpha(V, E))$. □

We now provide a general decidability result for view-based query answering under the TBox-centered semantics.

Theorem 24. *For a DL $\mathcal{L}$, if conjunctive query answering without UNA in $\mathcal{L}$ is decidable, then view-based query answering without UNA under the TBox-centered semantics in $\mathcal{L}$ is decidable. The same holds for the case with UNA.*

Proof. We give the proof only for the case without UNA. The case with UNA is analogous. Let $\mathcal{T}$ be a TBox in $\mathcal{L}$, $V = \langle v_1, \ldots, v_n \rangle$ a set of views, and $E = \langle e_1, \ldots, e_n \rangle$ a V-extension.

First, assume that $\vec{t} \notin vba_T(q, \mathcal{T}, V, E)$, and let $\mathcal{A}$ be an ABox such that $Mod(\langle \mathcal{T}, \mathcal{A} \rangle)$ is a T-solution for the vba-setting $\langle \mathcal{T}, V, E \rangle$ and $\vec{t} \notin cert(q, Mod(\langle \mathcal{T}, \mathcal{A} \rangle))$. We define an ABox $\mathcal{A}'$ as follows. For every $e_i \in E$, and for every $\vec{t}_j \in e_i$, let $\mathcal{A}_{\vec{t}_j}$ be an (arbitrarily chosen) minimal subset of $\mathcal{A}$ such that $\langle \mathcal{T}, \mathcal{A}_{\vec{t}_j} \rangle \models v_i(\vec{t}_j)$. Then, we define $\mathcal{A}' = \bigcup_{1 \leqslant i \leqslant n} \bigcup_{\vec{t}_j \in e_i} A_{\vec{t}_j}$. Since $\mathcal{A}' \subseteq \mathcal{A}$ and since by hypothesis $\vec{t} \notin cert(q, Mod(\langle \mathcal{T}, \mathcal{A} \rangle))$, it follows that $\vec{t} \notin cert(q, Mod(\langle \mathcal{T}, \mathcal{A}' \rangle))$. Moreover, the above definition of $\mathcal{A}'$ immediately implies that $cert(V, Mod(\langle \mathcal{T}, \mathcal{A}' \rangle)) = E$, i.e., $Mod(\langle \mathcal{T}, \mathcal{A}' \rangle)$ is also a T-solution for $\langle \mathcal{T}, V, E \rangle$. Finally, it is immediate to verify that, if $\vec{t}_j \in e_i$, then $\mathcal{A}_{\vec{t}_j}$ contains a number of instance assertions that is at most equal to the number of atoms in v_i, and a number of constant symbols that is at most equal to the number of existential variable symbols occurring in v_i. Now, let k be the maximum number of existential variable symbols occurring in a view in V. Then, let $a_1, \ldots, a_p$ be the set of constant symbols that occur in $\mathcal{A}'$ and that do not occur in E: we have that p is less than or equal to $m = |E| \cdot k$. Now let $b_1, \ldots, b_m$ be symbols not occurring in E and let $\mathcal{A}''$ be the ABox obtained from $\mathcal{A}'$ by replacing each constant symbol a_i with b_i, for every i such that $1 \leqslant i \leqslant p$. It is easy to check that $Mod(\langle \mathcal{T}, \mathcal{A}'' \rangle)$ is also a T-solution for $\langle \mathcal{T}, V, E \rangle$. Moreover, since $\vec{t} \notin cert(q, Mod(\langle \mathcal{T}, \mathcal{A}' \rangle))$, we immediately obtain $\vec{t} \notin cert(q, Mod(\langle \mathcal{T}, \mathcal{A}'' \rangle))$ (remember that q can mention only constant symbols occurring in E). We have thus shown that, if $\vec{t} \notin vba_T(q, \mathcal{T}, V, E)$, then there exists an ABox $\mathcal{A}'' \in \Phi$ such that $Mod(\langle \mathcal{T}, \mathcal{A}'' \rangle)$ is a T-solution for $\langle \mathcal{T}, V, E \rangle$ and $\vec{t} \notin cert(q, Mod(\langle \mathcal{T}, \mathcal{A}'' \rangle))$, where Φ is the set of ABoxes that can be built over the set of concept and role names occurring in $\mathcal{T}$ and E and on the set of constant symbols occurring in E plus the symbols $b_1, \ldots, b_m$.

Then, assume $\vec{t} \in vba_T(q, \mathcal{T}, V, E)$. In this case, it trivially follows that, for every ABox $\mathcal{A} \in \Phi$ such that $Mod(\langle \mathcal{T}, \mathcal{A} \rangle)$ is a T-solution for $\langle \mathcal{T}, V, E \rangle$, $\vec{t} \in cert(q, Mod(\langle \mathcal{T}, \mathcal{A} \rangle))$.

Therefore, we have that $\vec{t} \notin vba_T(q, \mathcal{T}, V, E)$ iff there exists an ABox $\mathcal{A}'' \in \Phi$ such that $Mod(\langle \mathcal{T}, \mathcal{A}'' \rangle)$ is a T-solution for $\langle \mathcal{T}, V, E \rangle$ and $\vec{t} \notin cert(q, Mod(\langle \mathcal{T}, \mathcal{A}'' \rangle))$. Since the number of ABoxes in Φ is finite, and the size of every such ABox is finite (in particular, polynomial with respect to the size of E) and since by hypothesis query answering in $\mathcal{L}$ is decidable, the claim follows. □

Exploiting the proof of the above theorem, we obtain the following general upper bounds for view-based query answering under the TBox-centered semantics.

Theorem 25. *For a DL $\mathcal{L}$, if conjunctive query answering without UNA in $\mathcal{L}$ is in* PTIME *with respect to data complexity, then view-based query answering without UNA under the TBox-centered semantics in $\mathcal{L}$ is in* coNP *with respect to data complexity. The same holds for the case with UNA.*

Proof. We give the proof only for the case without UNA. The case with UNA is analogous. Let $\mathcal{T}$ be a TBox in $\mathcal{L}$, let $V = \langle v_1, \ldots, v_n \rangle$ be a set of views and $E = \langle e_1, \ldots, e_n \rangle$ a V-extension, let q be a CQ, and let $\vec{t}$ be a tuple of constants. We prove that verifying whether $\vec{t} \notin vba_T(q, \mathcal{T}, V, E)$ is in NP. Indeed, from the proof of Theorem 24, it immediately follows that $\vec{t} \notin vba_T(q, \mathcal{T}, V, E)$ if there exists an ABox $\mathcal{A}$ whose size is polynomial with respect to the size of E, and such that $Mod(\langle \mathcal{T}, \mathcal{A} \rangle)$ is a T-solution for $\langle \mathcal{T}, V, E \rangle$ and $\vec{t} \notin cert(q, Mod(\langle \mathcal{T}, \mathcal{A} \rangle))$. Since by hypothesis conjunctive query answering in $\mathcal{L}$ is in PTIME with respect to data complexity, verifying whether $Mod(\langle \mathcal{T}, \mathcal{A} \rangle)$ is a T-solution for $\langle \mathcal{T}, V, E \rangle$ can be done in polynomial time, thus the claim follows. □

Theorem 26. *For a DL $\mathcal{L}$, if conjunctive query answering without UNA in $\mathcal{L}$ is in* coNP *with respect to data complexity, then view-based query answering without UNA under the TBox-centered semantics in $\mathcal{L}$ is in Π_2^p with respect to data complexity. The same holds for the case with UNA.*

Proof. We give the proof only for the case without UNA. The case with UNA is analogous. The proof is analogous to the proof of Theorem 25. As shown in that proof, $\vec{t} \notin vba_T(q, \mathcal{T}, V, E)$ if there exists an ABox $\mathcal{A}$ whose size is polynomial with respect to the size of E, and such that $Mod(\langle \mathcal{T}, \mathcal{A} \rangle)$ is a T-solution for $\langle \mathcal{T}, V, E \rangle$ and $\vec{t} \notin cert(q, Mod(\langle \mathcal{T}, \mathcal{A} \rangle))$. In the present case, since by hypothesis conjunctive query answering in $\mathcal{L}$ is in coNP with respect to data complexity, verifying whether

$Mod(\langle \mathcal{T}, \mathcal{A} \rangle)$ is a T-solution for $\langle \mathcal{T}, V, E \rangle$ can be done in polynomial time using an NP-oracle, which immediately implies the claim. □

We now show the following, very general, hardness result for view-based query answering under the TBox-centered semantics.

Theorem 27. *For every DL $\mathcal{L}$, view-based query answering, both with and without UNA, under the TBox-centered semantics in $\mathcal{L}$ is* coNP-*hard with respect to data complexity.*

Proof. We prove the claim for the case without UNA by reducing graph 3-colorability to view-based query answering in $\mathcal{L}$.

Let $G = (V_G, E_G)$ be a graph. The view definitions V are the following:

$$v_1(x) \leftarrow vertexCol(x, y), col(y)$$
$$v_2(x, y) \leftarrow edge(x, y)$$
$$v_3(x) \leftarrow col(x).$$

Intuitively, v_1 is used to assign a color to x, v_2 denotes the presence of an edge between x and y; and v_3 denotes colors. Consider the following V-extension E, populating v_1 with the nodes of G, v_2 with the edges of G, and v_3 with 3 colors:

$$E = \langle V_G, E_G, \{r, g, b\} \rangle.$$

Then consider the following Boolean query q, denoting two adjacent vertices (notice that the query is symmetric with respect to edges) colored with the same color:

$$q \leftarrow edge(x, y), vertexCol(x, z), vertexCol(y, z).$$

It is immediate to verify that G is 3-colorable iff $\langle \rangle \notin vba_T(q, \emptyset, V, E)$ (observe that the TBox is empty in this reduction). The key point is that, on the one hand, every ABox $\mathcal{A}$ such that $Mod(\langle \mathcal{T}, \mathcal{A} \rangle)$ is a T-solution for $(\emptyset, V, E)$ *must* be such that, for every vertex a of the graph, there exists a constant c such that the assertions $vertexCol(a, c)$, $col(c)$ belong to $\mathcal{A}$. On the other hand, due to the extension of the view v_3, such a c must be in the set $\{r, g, b\}$. This immediately implies the claim.

For the case with UNA, it is immediate to verify that adopting the UNA is immaterial, i.e., $\langle \rangle \notin vba_T(q, \emptyset, V, E)$ iff $\langle \rangle \notin vba_T^{UNA}(q, \emptyset, V, E)$. □

Notice that, in the above proof, we only exploit the fact that a DL ontology is interpreted under the open world assumption, and therefore, even an ontology constituted by an ABox only has incomplete information. This implies that the above hardness result holds also for incomplete relational databases.

5.2. Results for specific Description Logics

We now provide the computational characterization of view-based query answering under the TBox-centered semantics for all the specific DLs introduced in Section 2. We start by considering $DL\text{-}Lite_R$ and $DL\text{-}Lite_F$.

Theorem 28. *View-based query answering, both with and without UNA, under the TBox-centered semantics in $DL\text{-}Lite_R$ is* coNP-*complete with respect to data complexity.*

Proof. Membership in coNP follows from the fact that conjunctive query answering both with UNA and without UNA in $DL\text{-}Lite_R$ is in AC^0 with respect to data complexity (see Table 1) and from Theorem 25. coNP-hardness follows from Theorem 27. □

Theorem 29. *View-based query answering, both with and without UNA, under the TBox-centered semantics in $DL\text{-}Lite_F$ is* coNP-*complete with respect to data complexity.*

Proof. Membership in coNP follows from the fact that the data complexity of answering CQs in $DL\text{-}Lite_F$ is in AC^0 with UNA and in PTime without UNA (see Table 1), and from Theorem 25. In both cases, coNP-hardness follows from Theorem 27. □

Then, we turn our attention to the DLs $\mathcal{EL}$ and $\mathcal{ELH}$.

Theorem 30. *View-based query answering, both with and without UNA, under the TBox-centered semantics in $\mathcal{EL}$ and $\mathcal{ELH}$ is* coNP-*complete with respect to data complexity.*

Proof. Membership in coNP follows from Theorem 25 and from the fact that conjunctive query answering both with UNA and without UNA in $\mathcal{ELH}$ is in PTime(see Table 1). Hardness follows from Theorem 27. $\square$

Finally, we examine view-based query answering under the TBox-centered semantics in the DLs ranging from $\mathcal{AL}$ to $\mathcal{SHIQ}$.

Theorem 31. *View-based query answering, both with and without UNA, under the TBox-centered semantics in $\mathcal{AL}$ is Π_2^p-hard with respect to data complexity.*

Proof. Since the UNA is immaterial for $\mathcal{AL}$, it suffices to give the proof for the case without UNA. We prove the claim by reducing 2-QBF validity to view-based query answering under the TBox-centered semantics in $\mathcal{AL}$.

Let $\phi = \forall \vec{x}.\exists \vec{y}.f(\vec{x}, \vec{y})$ be a 2-QBF formula, where $\vec{x} = x_1, \ldots, x_n$, $\vec{y} = y_1, \ldots, y_m$ and f is a 3-CNF formula over the propositional variables $\vec{x}$ and $\vec{y}$, i.e., a formula of the form $f = c_1 \wedge \cdots \wedge c_k$, where each c_i is of the form $c_i = l_1^i \vee l_2^i \vee l_3^i$ and each l_j^i is a literal over the variables $\vec{x}$ and $\vec{y}$. We define the following vba-setting $\langle \mathcal{T}, V, E \rangle$:

- $\mathcal{T}$ is the following $\mathcal{AL}$-TBox:

$$
\begin{array}{ll}
& Aux_2 \sqsubseteq \forall Val.\neg C_f \\
\neg C_t \sqsubseteq C_f & \forall Val.\neg C_f \sqsubseteq Aux_2 \\
C_t \sqsubseteq \neg C_f & \neg Aux_1 \sqsubseteq \forall Val.C_t \\
D_t \sqsubseteq C_t & \neg Aux_2 \sqsubseteq \forall Val.C_f \\
D_f \sqsubseteq C_f & \neg Aux_1 \sqsubseteq \forall Comp.\forall Val.C_f \\
Aux_1 \sqsubseteq \forall Val.\neg C_t & \neg Aux_2 \sqsubseteq \forall Comp.\forall Val.C_t \\
\forall Val.\neg C_t \sqsubseteq Aux_1 & C_{ex} \sqsubseteq \exists Val.
\end{array}
$$

- V is constituted by the following view definitions:

$$
\begin{array}{ll}
v_{ex}(x) \leftarrow C_{ex}(x) & v_{GX}(x) \leftarrow C_{univ}(x), Val(x, y), TV(y, y) \\
v_{comp}(x, y) \leftarrow Comp(x, y) & v_{EX}(y) \leftarrow C_{univ}(x), Val(x, y), TV(y, y) \\
v_{Lit_1}(x, y) \leftarrow Lit_1(x, y) & v_{GY}(x, y) \leftarrow C_{ex}(x), Val(x, y) \\
v_{Lit_2}(x, y) \leftarrow Lit_2(x, y) & v_{TV}(x) \leftarrow TV(x, x) \\
v_{Lit_3}(x, y) \leftarrow Lit_3(x, y) & v_{D_t}(x) \leftarrow D_t(x) \\
v_{univ}(x) \leftarrow C_{univ}(x) & v_{D_f}(x) \leftarrow D_f(x) \\
\multicolumn{2}{c}{v_\phi \leftarrow Lit_1(x, y_1), Lit_2(x, y_2), Lit_3(x, y_3),} \\
\multicolumn{2}{c}{\qquad Val(y_1, z_1), Val(y_2, z_2), Val(y_3, z_3),} \\
\multicolumn{2}{c}{\qquad C_f(z_1), C_f(z_2), C_f(z_3).}
\end{array}
$$

- E is the following V-extension, where we denote with $E(v)$ the extension associated to view v:

$$
\begin{array}{ll}
E(v_{ex}) = \{y_1, \ldots, y_m, \vec{y}_1, \ldots, \vec{y}_n\} & \\
E(v_{comp}) = \big\{ \langle x_1, \vec{x}_1 \rangle, \ldots, \langle x_n, \vec{x}_n \rangle, & E(v_{EX}) = \{0, 1\} \\
\qquad \langle y_1, \vec{y}_1 \rangle, \ldots, \langle y_m, \vec{y}_m \rangle \big\} & E(v_{GY}) = \emptyset \\
E(v_{Lit_1}) = \big\{ \langle c_i, l_i^1 \rangle \mid 1 \leqslant i \leqslant n \big\} & E(v_{TV}) = \{0, 1\} \\
E(v_{Lit_2}) = \big\{ \langle c_i, l_i^2 \rangle \mid 1 \leqslant i \leqslant n \big\} & E(v_{D_t}) = \{1\} \\
E(v_{Lit_3}) = \big\{ \langle c_i, l_i^3 \rangle \mid 1 \leqslant i \leqslant n \big\} & E(v_{D_f}) = \{0\} \\
E(v_{univ}) = \{x_1, \ldots, x_m, \vec{x}_1, \ldots, \vec{x}_n\} & E(v_\phi) = \{\langle \rangle\}. \\
E(v_{GX}) = \{x_1, \ldots, x_m, \vec{x}_1, \ldots, \vec{x}_n\} &
\end{array}
$$

We now show that the formula ϕ is valid iff $\langle \mathcal{T}, V, E \rangle$ has no T-solution.

First, suppose that ϕ is not valid. Then, there exists an assignment μ_x for $\vec{x}$ such that for every assignment μ_y for $\vec{y}$, the evaluation of f in $\mu_x \cup \mu_y$ is false. We now define an ABox $\mathcal{A}$ as follows:

$$
\begin{aligned}
& \big\{ C_{ex}(y) \mid y \in E(v_{ex}) \big\} \\
& \cup \big\{ Comp(x, y) \mid \langle x, y \rangle \in E(v_{comp}) \big\} \\
& \cup \big\{ Lit_1(x, y) \mid \langle x, y \rangle \in E(v_{Lit_1}) \big\} \\
& \cup \big\{ Lit_2(x, y) \mid \langle x, y \rangle \in E(v_{Lit_2}) \big\}
\end{aligned}
$$

$$\cup \left\{ Lit_3(x, y) \mid \langle x, y \rangle \in E(v_{Lit_3}) \right\}$$

$$\cup \left\{ C_{univ}(x) \mid x \in E(v_{univ}) \right\}$$

$$\cup \left\{ TV(x, x) \mid x \in E(v_{TV}) \right\}$$

$$\cup \left\{ D_t(1), D_f(0) \right\}$$

$$\cup \left\{ Val(x, y) \mid x \in E(v_{univ}) \text{ and } y = \mu_x(x) \right\}.$$

It is easy to verify that $Mod(\langle \mathcal{T}, \mathcal{A} \rangle)$ is a T-solution for $\langle \mathcal{T}, V, E \rangle$. The key points are the following: (i) in the KB, the interpretation of every propositional variable $p \in \{x_1, \dots, x_n, y_1, \dots, y_m\}$ is obtained by looking at the object connected to p through the role Val (such an object must belong either to the concept C_t or to the concept C_f); (ii) the interpretation of the variables $x_1, \dots, x_n$ is fixed in the ABox, and is the same as μ_x; (iii) since by hypothesis for every assignment μ_y for $\vec{y}$, the evaluation of f in $\mu_x \cup \mu_y$ is false (i.e., there exists a disjunct of f that is evaluated to false in $\mu_x \cup \mu_y$) it follows that, in every model if $\langle \mathcal{T}, \mathcal{A} \rangle$, the view v_ϕ is true, thus $E(v_\phi)$ corresponds to the certain answers of v_ϕ over $\langle \mathcal{T}, \mathcal{A} \rangle$.

On the other hand, suppose ϕ is not valid. Then, for every assignment μ_x for $\vec{x}$ there exists an assignment μ_y for $\vec{y}$ such that the evaluation of f in $\mu_x \cup \mu_y$ is true. Now, suppose $\mathcal{A}$ is an ABox which is a T-solution for $\langle \mathcal{T}, V, E \rangle$. Due to the form of V, it follows that: (i) to obtain $E(v_{GX}) = cert(v_{GX}, Mod(\langle T, \mathcal{A} \rangle))$ the ABox $\mathcal{A}$ *must* decide the truth interpretation of every $x_i \in \{x_1, \dots, x_n\}$, i.e., it must contain an assertion of the form $Val(x_i, n)$; (ii) to obtain $E(v_{GY}) = cert(v_{GY}, Mod(\langle T, \mathcal{A} \rangle))$ the ABox $\mathcal{A}$ *must not* decide the truth interpretation of any $y_i \in \{y_1, \dots, y_m\}$, i.e., it cannot contain any assertion of the form $Val(y_i, n)$. Now let μ' be the assignment for $\vec{x}$ that corresponds to the guess represented in $\mathcal{A}$. By hypothesis, there exists an assignment μ'' for $\vec{y}$ such that the evaluation of f in $\mu' \cup \mu''$ is true. But then, there exists a model $\mathcal{I}$ of $\langle \mathcal{T}, \mathcal{A} \rangle$ that interprets (through the role Val) the variables in $\vec{y}$ as in μ'', consequently the query v_ϕ is false in $\mathcal{I}$, and hence $\langle \rangle \notin cert(v_\phi, Mod(\langle \mathcal{T}, \mathcal{A} \rangle))$, thus contradicting the hypothesis that $Mod(\langle \mathcal{T}, \mathcal{A} \rangle)$ is a T-solution for $\langle \mathcal{T}, V, E \rangle$.

Then, we define a query $q \leftarrow C(a)$ where C is a concept that does not occur in $\mathcal{T}$ and V. It is immediate to see that $\langle \rangle \in vba_T(q, \mathcal{T}, V, E)$ iff $\langle \mathcal{T}, V, E \rangle$ has no T-solution. Consequently, the claim holds. $\square$

We are now ready to characterize the data complexity of view-based query answering for the DLs lying between $\mathcal{AL}$ and $\mathcal{SHIQ}$, i.e., those DLs that are both sub-logics of $\mathcal{SHIQ}$ and that have $\mathcal{AL}$ as a sub-logic. Note that many interesting DLs studied in the literature, including all the $\mathcal{AL}$-family [10] and a large part of the $\mathcal{S}$-family [19], belong to this class.

Theorem 32. *View-based query answering, both with and without UNA, under the TBox-centered semantics in all DLs lying between $\mathcal{AL}$ and $\mathcal{SHIQ}$ is Π_2^p-complete with respect to data complexity.*

Proof. The lower bound for $\mathcal{AL}$ follows from the proof of Theorem 31 and from the fact that query answering both with UNA and without UNA in $\mathcal{AL}$ is coNP-hard (see Table 1), while the upper bound for $\mathcal{SHIQ}$ follows from Theorem 26 and from the fact that conjunctive query answering both with UNA and without UNA in $\mathcal{SHIQ}$ is in coNP (see Table 1). $\square$

6. Related work

The interest in view-based query processing originated in the context of relational databases. As we said in the introduction, there are two approaches to this problem, namely, query rewriting and query answering.

In the relational context, query rewriting has been studied for the case of conjunctive queries (with or without arithmetic comparisons) [7,27,28], disjunctive views [29], queries with aggregates [30,31], recursive queries and nonrecursive views [32], queries with negated goals [33], and in the presence of integrity constraints [34–36] and of limitations in accessing the views [37,38]. Rewriting techniques for query optimization are described, for example, in [39–41]. A comprehensive framework for view-based query answering in relational databases, as well as several interesting results, are presented in [42]. In [2], an analysis of the data complexity of the problem under various assumptions is carried out for the case where the views and the queries are expressed in terms of various languages (conjunctive queries, Datalog, first-order queries, etc.). In particular, in [2], the investigation has been carried out distinguishing between sound and exact view extensions: an extension e for a view v is called sound with respect to a database $\mathcal{D}$, if all the tuples in e satisfy the query v over $\mathcal{D}$, and is called exact if e is exactly the set of tuples satisfying v over $\mathcal{D}$. Further results in the presence of integrity constraints are reported in [36], and in the context of data exchange in [43,44].

View-based query processing has also been studied in the context of semi-structured databases. In the case of graph-based models, the problem has been studied for the class of regular path queries in [45–48]. In [49,50], a further distinction is proposed for characterizing the domain of the database (open vs. closed domain assumption), and the problem is studied for the case of regular-path queries, both with and without the inverse operator. In the case of XML-based model, results on both view-based query rewriting and view-based query answering are reported in [51–55] for several variants of the XPath query language.

While all the above mentioned articles study view-based query processing in the context of databases, our investigation is carried out in a setting where both views and queries are formulated over an ontology. When we move from databases to ontologies, we face the problem of dealing with a set of models (i.e., the set of interpretations that satisfy all the assertions)

Table 2
Data complexity of computing certain answers and of view-based query answering.

DL	Computing certain answers		View-based query answering with and without UNA	
	With UNA	Without UNA	Model-centered	TBox-centered
$DL\text{-}Lite_R$	in AC^0	in AC^0	in AC^0	coNP-complete
$DL\text{-}Lite_F$	in AC^0	PTime-complete	PTime-complete	coNP-complete
$\mathcal{EL}, \mathcal{ELH}$	PTime-complete	PTime-complete	PTime-complete	coNP-complete
$\mathcal{AL}, \mathcal{SHIQ}$	coNP-complete	coNP-complete	coNP-complete	Π_2^p-complete

rather than a single model (the interpretation structure corresponding to the database). To the best of our knowledge, the only articles dealing with view-based query processing, and in particular query answering, in this setting are [56] and [57]. The problem of view-based query rewriting is studied in [56] for the case of DL ontologies. The paper shows that if the view definitions do not contain existential variables, then it is always possible to find a rewriting that is a union of conjunctive queries, and furthermore, this rewriting produces the maximal set of answers possible from the views. On the other hand, if the views have existential variables, then it is not always possible to find a maximal rewriting. On the contrary, [57] studies the combined complexity (as opposed to data complexity) of view-based query answering for a very expressive DL, and under both the open and the closed world assumption on the view extensions.

Notice, however, that the semantics of view-based query answering adopted in those papers is different with respect to the one presented here. Indeed, both in [56] and in [57] the views V and their extensions E simply contribute, together with the TBox, to determining which are the models of the whole knowledge base. The models of the whole knowledge base are those models of the TBox where all tuples in the view extensions satisfy the corresponding view definition. As usual, the answers to a query provided by the system are those that are certain with respect to all such models. On the contrary, in our approach, the extensions of the views coincide with the certain answers to the queries corresponding to such views. So, in our approach each view v can be considered exact, in the sense that its extension e coincides with the set of certain answers of the query v over the underlying ontology. However, one should take into account that the underlying ontology may have models such that the evaluation of v over such models would yield more answers than those in the extension e.

7. Discussion and conclusions

The work presented in this paper constitutes a first systematic study of the semantics and the complexity of view-based query answering in DLs. The framework we have introduced distinguishes between different semantics for the problem, corresponding to different variants of the notion of solution. We have related view-based query answering to privacy-aware information access, and we have presented several algorithms and complexity results for various DLs, both in the model-centered and in the TBox-centered semantics.

We summarize in Table 2 the results we have presented about the data complexity in the various DLs, for the two semantics. To simplify the comparison with the problem of computing certain answers, the table also reports the data complexity of such a problem, in the two cases with UNA and without UNA.

Our work shows that, for all DLs, the complexity of view-based query answering under the model-centered semantics is the same as the complexity of computing certain answers without UNA. This means that view-based query answering has also the same complexity as computing certain answers without UNA in all DLs except for $DL\text{-}Lite_F$, where the data complexity jumps from AC^0 to PTime-complete.

The situation is different in the case of the TBox-centered semantics. Indeed, in this semantics, a counterexample to a tuple $\vec{t}$ being a certain answer to q over $\langle \mathcal{T}, V, E \rangle$ must be represented by an ontology $\langle \mathcal{T}, \mathcal{A} \rangle$ expressed in $\mathcal{L}$. This requirement makes the problem harder: in all the DLs considered here, the data complexity of view-based query answering under the TBox-centered semantics is higher than under the model-centered semantics. In particular, in the DLs where view-based query answering is polynomially tractable under the model-centered semantics, the problem under the TBox-centered semantics becomes intractable, whereas in the DLs where it is coNP-complete under the model-centered semantics, it jumps one level up in the polynomial hierarchy under the TBox-centered semantics.

The work presented in this paper will be continued along different directions. In particular, we aim at studying other variant of solutions, such as the one called "language-centered" in [20]. In such a semantics, a counterexample to a tuple being a certain answer to q over $\langle \mathcal{T}, V, E \rangle$ is any subset of $Mod(\mathcal{T})$ that is expressible as a knowledge base $\langle \mathcal{T}', \mathcal{A} \rangle$ in the language $\mathcal{L}$, where, differently from the case of TBox-centered semantics, $\mathcal{T}'$ need not be equivalent to $\mathcal{T}$. Another interesting issue is to investigate the impact of both extending and restricting the language used to express the query and the views on the complexity of view-based query answering, with the goal of singling out more cases where the problem is tractable.

Finally, it would be very interesting to exploit the relationship between Description Logics and disjunctive databases in the study of view-based query answering. In particular, as shown in [58], it is possible to reduce the usual reasoning tasks in Description Logics (e.g., instance checking and query answering) to reasoning in Disjunctive Datalog programs. The computational results presented in this paper are compatible with those that would be obtained by encoding view-based

query answering in Disjunctive Datalog. Hence, it would be interesting to explore whether the correspondence between Description Logics and disjunctive databases can be extended to view-based query answering.

Acknowledgments

References

[1] J.D. Ullman, Information integration using logical views, in: Proc. of the 6th Int. Conf. on Database Theory, ICDT'97, in: Lecture Notes in Comput. Sci., vol. 1186, Springer, 1997, pp. 19–40.

[2] S. Abiteboul, O. Duschka, Complexity of answering queries using materialized views, in: Proc. of the 17th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems, PODS'98, 1998, pp. 254–265.

[3] A.Y. Halevy, Answering queries using views: A survey, Very Large Database J. 10 (4) (2001) 270–294.

[4] Z. Zhang, A. Mendelzon, Authorization views and conditional query containment, in: Proc. of the 10th Int. Conf. on Database Theory, ICDT 2005, in: Lecture Notes in Comput. Sci., vol. 3363, Springer, 2005, pp. 259–273.

[5] S. Rizvi, A.O. Mendelzon, S. Sudarshan, P. Roy, Extending query rewriting techniques for fine-grained access control, in: Proc. of the ACM SIGMOD Int. Conf. on Management of Data, 2004, pp. 551–562.

[6] P. Stouppa, T. Studer, A formal model of data privacy, in: I. Virbitskaite, A. Voronkov (Eds.), Ershov Memorial Conference, in: Lecture Notes in Comput. Sci., vol. 4378, Springer, 2007, pp. 400–408.

[7] A.Y. Levy, A.O. Mendelzon, Y. Sagiv, D. Srivastava, Answering queries using views, in: Proc. of the 14th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems, PODS'95, 1995, pp. 95–104.

[8] D. Calvanese, G. De Giacomo, M. Lenzerini, M.Y. Vardi, View-based query processing and constraint satisfaction, in: Proc. of the 15th IEEE Symp. on Logic in Computer Science, LICS 2000, 2000, pp. 361–371.

[9] D. Calvanese, G. De Giacomo, M. Lenzerini, M.Y. Vardi, View-based query processing: On the relationship between rewriting, answering and losslessness, Theoret. Comput. Sci. 371 (3) (2007) 169–182.

[10] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P.F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation and Applications, Cambridge University Press, 2003.

[11] I. Horrocks, Ontologies and the Semantic Web, Commun. ACM 51 (12) (2008) 58–67.

[12] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, R. Rosati, Tractable reasoning and efficient query answering in description logics: The DL-Lite family, J. Automat. Reason. 39 (3) (2007) 385–429.

[13] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, R. Rosati, Linking data to ontologies, J. Data Semantics X (2008) 133–173.

[14] F. Baader, S. Brandt, C. Lutz, Pushing the $\mathcal{EL}$ envelope, in: Proc. of the 19th Int. Joint Conf. on Artificial Intelligence, IJCAI 2005, 2005, pp. 364–369.

[15] A. Krisnadhi, C. Lutz, Data complexity in the $\mathcal{EL}$ family of description logics, in: Proc. of the 14th Int. Conf. on Logic for Programming, Artificial Intelligence, and Reasoning, LPAR 2007, 2007, pp. 333–347.

[16] R. Rosati, On conjunctive query answering in $\mathcal{EL}$, in: Proc. of the 20th Int. Workshop on Description Logic, DL 2007, CEUR Electronic Workshop Proceedings, vol. 250, 2007, pp. 451–458, http://ceur-ws.org/.

[17] I. Horrocks, U. Sattler, S. Tobies, Reasoning with individuals for the description logic $\mathcal{SHIQ}$, in: D. McAllester (Ed.), Proc. of the 17th Int. Conf. on Automated Deduction, CADE 2000, in: Lecture Notes in Comput. Sci., vol. 1831, Springer, 2000, pp. 482–496.

[18] I. Horrocks, P.F. Patel-Schneider, F. van Harmelen, From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language, J. Web Semantics 1 (1) (2003) 7–26.

[19] B. Glimm, I. Horrocks, C. Lutz, U. Sattler, Conjunctive query answering for the description logic $\mathcal{SHIQ}$, J. Artificial Intelligence Res. 31 (2008) 151–198.

[20] D. Calvanese, G. De Giacomo, M. Lenzerini, R. Rosati, View-based query answering over description logic ontologies, in: Proc. of the 11th Int. Conf. on the Principles of Knowledge Representation and Reasoning, KR 2008, 2008, pp. 242–251.

[21] S. Abiteboul, R. Hull, V. Vianu, Foundations of Databases, Addison Wesley Publ. Co., 1995.

[22] F. Baader, W. Nutt, Basic description logics, in: F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P.F. Patel-Schneider (Eds.), The Description Logic Handbook: Theory, Implementation and Applications, Cambridge University Press, 2003, pp. 43–95, Ch. 2.

[23] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, R. Rosati, Data complexity of query answering in description logics, in: Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning, KR 2006, 2006, pp. 260–270.

[24] M. Ortiz, D. Calvanese, T. Eiter, Data complexity of query answering in expressive description logics via tableaux, J. Automat. Reason. 41 (1) (2008) 61–98.

[25] A. Artale, D. Calvanese, R. Kontchakov, M. Zakharyaschev, The DL-Lite family and relations, J. Artificial Intelligence Res. 36 (2009) 1–69.

[26] D. Maier, A.O. Mendelzon, Y. Sagiv, Testing implications of data dependencies, ACM Trans. Database Systems 4 (4) (1979) 455–469.

[27] A. Rajaraman, Y. Sagiv, J.D. Ullman, Answering queries using templates with binding patterns, in: Proc. of the 14th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems, PODS'95, 1995, pp. 105–112.

[28] R. Pottinger, A.Y. Levy, A scalable algorithm for answering queries using views, in: Proc. of the 26th Int. Conf. on Very Large Data Bases, VLDB 2000, 2000, pp. 484–495.

[29] F.N. Afrati, M. Gergatsoulis, T. Kavalieros, Answering queries using materialized views with disjunction, in: Proc. of the 7th Int. Conf. on Database Theory, ICDT'99, in: Lecture Notes in Comput. Sci., vol. 1540, Springer, 1999, pp. 435–452.

[30] D. Srivastava, S. Dar, H.V. Jagadish, A. Levy, Answering queries with aggregation using views, in: Proc. of the 22nd Int. Conf. on Very Large Data Bases, VLDB'96, 1996, pp. 318–329.

[31] S. Cohen, W. Nutt, A. Serebrenik, Rewriting aggregate queries using views, in: Proc. of the 18th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems, PODS'99, 1999, pp. 155–166.

[32] O.M. Duschka, M.R. Genesereth, Answering recursive queries using views, in: Proc. of the 16th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems, PODS'97, 1997, pp. 109–116.

[33] S. Flesca, S. Greco, Rewriting queries using views, IEEE Trans. on Knowledge and Data Engineering 13 (6) (2001) 980–995.

[34] J. Gryz, Query folding with inclusion dependencies, in: Proc. of the 14th IEEE Int. Conf. on Data Engineering, ICDE'98, 1998, pp. 126–133.

[35] O.M. Duschka, A.Y. Levy, Recursive plans for information gathering, in: Proc. of the 15th Int. Joint Conf. on Artificial Intelligence, IJCAI'97, 1997, pp. 778–784.

[36] A. Calì, D. Lembo, R. Rosati, Query rewriting and answering under constraints in data integration systems, in: Proc. of the 18th Int. Joint Conf. on Artificial Intelligence, IJCAI 2003, 2003, pp. 16–21.

[37] C. Li, E. Chang, On answering queries in the presence of limited access patterns, in: Proc. of the 8th Int. Conf. on Database Theory, ICDT 2001, 2001, pp. 219–233.

[38] A. Calì, D. Calvanese, D. Martinenghi, Dynamic query optimization under access limitations and dependencies, J. Univers. Comput. Sci. 15 (1) (2009) 33–62.

[39] S. Chaudhuri, S. Krishnamurthy, S. Potarnianos, K. Shim, Optimizing queries with materialized views, in: Proc. of the 11th IEEE Int. Conf. on Data Engineering, ICDE'95, 1995, pp. 190–200.

[40] S. Adali, K.S. Candan, Y. Papakonstantinou, V.S. Subrahmanian, Query caching and optimization in distributed mediator systems, in: Proc. of the ACM SIGMOD Int. Conf. on Management of Data, 1996, pp. 137–148.

[41] O.G. Tsatalos, M.H. Solomon, Y.E. Ioannidis, The GMAP: A versatile tool for physical data independence, Very Large Database J. 5 (2) (1996) 101–118.

[42] G. Grahne, A.O. Mendelzon, Tableau techniques for querying information sources through global schemas, in: Proc. of the 7th Int. Conf. on Database Theory, ICDT'99, in: Lecture Notes in Comput. Sci., vol. 1540, Springer, 1999, pp. 332–347.

[43] P.G. Kolaitis, Schema mappings, data exchange, and metadata management, in: Proc. of the 24rd ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems, PODS 2005, 2005, pp. 61–75.

[44] L. Libkin, Data exchange and incomplete information, in: Proc. of the 25th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems, PODS 2006, 2006, pp. 60–69.

[45] D. Calvanese, G. De Giacomo, M. Lenzerini, M.Y. Vardi, Rewriting of regular expressions and regular path queries, in: Proc. of the 18th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems, PODS'99, 1999, pp. 194–204.

[46] D. Calvanese, G. De Giacomo, M. Lenzerini, M.Y. Vardi, Rewriting of regular expressions and regular path queries, J. Comput. System Sci. 64 (3) (2002) 443–465.

[47] G. Grahne, A. Thomo, Query containment and rewriting using views for regular path queries under constraints, in: Proc. of the 22nd ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems, PODS 2003, 2003, pp. 111–122.

[48] G. Grahne, A. Thomo, Algebraic rewritings for optimizing regular path queries, Theoret. Comput. Sci. 296 (3) (2003) 453–471.

[49] D. Calvanese, G. De Giacomo, M. Lenzerini, M.Y. Vardi, Answering regular path queries using views, in: Proc. of the 16th IEEE Int. Conf. on Data Engineering, ICDE 2000, 2000, pp. 389–398.

[50] D. Calvanese, G. De Giacomo, M. Lenzerini, M.Y. Vardi, Query processing using views for regular path queries with inverse, in: Proc. of the 19th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems, PODS 2000, 2000, pp. 58–66.

[51] M.F. Fernandez, D. Suciu, Optimizing regular path expressions using graph schemas, in: Proc. of the 14th IEEE Int. Conf. on Data Engineering, ICDE'98, 1998, pp. 14–23.

[52] T. Milo, D. Suciu, Index structures for path expressions, in: Proc. of the 7th Int. Conf. on Database Theory, ICDT'99, in: Lecture Notes in Comput. Sci., vol. 1540, Springer, 1999, pp. 277–295.

[53] Y. Papakonstantinou, V. Vassalos, Query rewriting for semistructured data, in: Proc. of the ACM SIGMOD Int. Conf. on Management of Data, 1999, pp. 455–466.

[54] F.N. Afrati, R. Chirkova, M. Gergatsoulis, B. Kimelfeld, V. Pavlaki, Y. Sagiv, On rewriting XPath queries using views, in: Proc. of the 12th Int. Conf. on Extending Database Technology, EDBT 2009, 2009, pp. 168–179.

[55] D. Calvanese, G. De Giacomo, M. Lenzerini, M.Y. Vardi, An automata-theoretic approach to Regular XPath, in: Proc. of the 12th Int. Symp. on Database Programming Languages, DBPL 2009, in: Lecture Notes in Comput. Sci., vol. 5708, Springer, 2009, pp. 18–35.

[56] C. Beeri, A.Y. Levy, M.-C. Rousset, Rewriting queries using views in description logics, in: Proc. of the 16th ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems, PODS'97, 1997, pp. 99–108.

[57] D. Calvanese, G. De Giacomo, M. Lenzerini, Answering queries using views over description logics knowledge bases, in: Proc. of the 17th Nat. Conf. on Artificial Intelligence, AAAI 2000, 2000, pp. 386–391.

[58] U. Hustadt, B. Motik, U. Sattler, Reasoning in description logics by a reduction to Disjunctive Datalog, J. Automat. Reason. 39 (3) (2007) 351–384.