Query Rewriting for Inconsistent DL-Lite Ontologies

Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati,
Marco Ruzzi, and Domenico Fabio Savo

Dipartimento di Informatica e Sistemistica Antonio Ruberti
Università di Roma "La Sapienza"
lastname@dis.uniroma1.it

Abstract. In this paper we study the problem of obtaining meaningful answers to queries posed over inconsistent *DL-Lite* ontologies. We consider different variants of inconsistency-tolerant semantics and show that for some of such variants answering unions of conjunctive queries (UCQs) is first-order (FOL) rewritable, i.e., it can be reduced to standard evaluation of a FOL/SQL query over a database. Since FOL-rewritability of query answering for UCQs over consistent ontologies under first-order logic semantics is one of the distinguishing features of *DL-Lite*, in this paper we actually identify some settings in which such property is preserved also under inconsistency-tolerant semantics. We therefore show that in such settings inconsistency-tolerant query answering has the same computational complexity of standard query answering and that it can rely on well-established relational database technology, as under standard DL semantics.

1 Introduction

In the last years there has been a continuously growing use of ontologies in many ICT applications and Description Logics (DLs) have been recognized as the best means for the formal specification of ontologies, for their ability of combining modeling power and decidability of reasoning [2]. For these characteristics, DLs constitute the logical underpinning of prominent ontology languages, such as OWL 2, the W3C standard language for ontology specification[1]. DL ontologies are constituted by a *TBox*, representing intensional knowledge, and an *ABox*, representing extensional knowledge. Motivated by the fact that the size of real world ontologies is scaling up and that the ability of dealing with ontologies with very large ABoxes has become a crucial requirement for many modern applications, various DLs have been recently proposed that allow for tractable reasoning. Among such DLs, the logics of the *DL-Lite* family [5, 19] present the distinguishing characteristic of enabling first-order (FOL) rewritability of query answering of unions of conjunctive queries (UCQs). This means that to answer a UCQ q in *DL-Lite* it is possible to first rewrite q into a first-order query q_r, only on the basis of the knowledge specified in the TBox, and then evaluate q_r over the ABox, which can be seen as a plain database. FOL-rewritability of UCQs is a notable property, since many practical applications require the expressivity of UCQs for query answering, and their FOL-rewritability allows for delegating the management of the ABox to a relational DBMS, because the FOL queries produced by the rewriting process are directly

[1] http://www.w3.org/TR/owl2-overview/

translatable into SQL. In other words, in this way we reduce a form of reasoning under incomplete information, i.e., query answering over an ontology, to classical evaluation of an SQL query. Notably, the ABox does not need to be touched during the rewriting phase, and no data preprocessing is needed (as for example required in [6, 14]). This turns out to be crucial, for instance, in all those applications in which ontologies, and in particular their intensional component, are used to access data stored in external repositories, such as in ontology-based data integration [19, 3].

In these applications, however, even though the TBox of the ontology is usually a consistent theory, its axioms may often be contradicted by assertions of the ABox, in general collected from various autonomous sources. This actually implies that the resulting ontology is inconsistent, and that reasoning over it is trivialized. The ability of dealing with such a form of inconsistency is of critical importance, in particular to obtain meaningful answers to queries posed over inconsistent ontologies.

In a previous paper [15], we have proposed various *inconsistency-tolerant* semantics that allow for such possibility, and have shown that answering UCQs under some of these semantics is tractable in *DL-Lite*$_{\mathcal{A}}$, one of the most expressive logics of the *DL-Lite* family, which is at the basis of OWL 2 QL[2], a standard tractable fragment of OWL 2. In [15], however, we left open the problem whether answering UCQs is in fact FOL-rewritable in the tractable cases we have identified. In the present paper we positively reply to this question, thus showing that in these cases inconsistency-tolerant query answering has the same computational complexity of standard query answering, and that it can rely on relational database technology, as under classical DL semantics.

More precisely, we consider here the *Intersection ABox Repair (IAR)* semantics and the *Intersection Closed ABox Repair (ICAR)* of [15], and provide algorithms for the FOL-rewritability of UCQs under such semantics. The notion of *repair* over which such semantics rely is borrowed from the database literature [1], and it is rooted in research on belief revision and updates [7, 9]. Roughly speaking, given an ontology $\mathcal{O}$ with TBox $\mathcal{T}$ and ABox $\mathcal{A}$, in the IAR semantics the (only) repair is obtained as the intersection of all ABoxes $\mathcal{A}'$ consistent with $\mathcal{T}$ that are contained in $\mathcal{A}$ and are maximal with respect to set containment. In the ICAR semantics, instead, we consider also ABox assertions that are logically implied by the TBox and by any subset of $\mathcal{A}$ that is consistent with $\mathcal{T}$. We call such set of ABox assertions the *closure of* $\mathcal{A}$ *with respect to* $\mathcal{T}$, denoted $clc(\mathcal{A}, \mathcal{T})$, and define the (only) repair in the ICAR semantics as the ABox obtained by the intersection of all ABoxes $\mathcal{A}'$ consistent with $\mathcal{T}$ that are contained in $clc(\mathcal{A}, \mathcal{T})$ and are maximal with respect to set containment. As shown in [15], there are cases in which the IAR semantics produces different repairs for ontologies that should be instead considered to some extent equivalent to the aims of certain applications, even though specified in a syntactically different form. The ICAR semantics does not present this behavior, for its ability of considering essentially only repairs that are "closed" with respect to the knowledge represented by the TBox, thus flattening differences in the syntactic specification of the ABox (for ontologies having the same TBox).

We notice that first-order rewritability of inconsistency-tolerant query answering has been already considered in the database literature [1, 8, 11]. These papers, however, provide only sufficient conditions for FOL-rewritability of some fragments of conjunc-

[2] `http://www.w3.org/TR/2008/WD-owl2-profiles-20081008/`

tive queries, specified over database schemas equipped with limited forms of integrity constraints (i.e, binary universal constraints [1], or single key dependencies on relations [8], possibly combined in a controlled way with exclusion dependencies [11], which impose disjunctions on projections of relations). The only paper considering FOL-rewriting of UCQs is [16], which provides some preliminary results for database schemas with only key dependencies. There is also an extensive literature that study inconsistency in ontologies, which however do not distinguish between inconsistency at the intensional and extensional level as we do our approach (in other words, also inconsistency in the TBox are considered) [18, 13, 12, 20, 17, 21]. Our approach is also deeply connected with the work on belief revision, since the ABox can be considered the initial knowledge, whereas the TBox represents the incoming knowledge. In this respect, we notice that the IAR and the ICAR semantics both conform to the WIDTIO principle of belief revision [7], but also that the special kind of theories that can be specified in *DL-Lite*$_\mathcal{A}$ has not been analyzed by works from these area.

The rest of the paper is organized as follows. In Section 2, we give some preliminaries, and in particular we introduce *DL-Lite*$_\mathcal{A}$ and the notion of FOL-rewritability. In Section 3, we recall the definition of the IAR and the ICAR semantics. In Section 4, we present FOL-rewritings for UCQs under the IAR and the ICAR semantics, and, in Section 5 we give a complete example. Finally, in Section 6 we conclude the paper.

2 Preliminaries

A Description Logic ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ consists of a TBox $\mathcal{T}$, representing intensional knowledge, and an ABox $\mathcal{A}$ representing extensional knowledge. $\Gamma_\mathcal{O}$ will denote the alphabet of the ontology, that is, the union of the predicate symbols occurring in $\mathcal{T}$ and $\mathcal{A}$, whereas Γ_C will denote the alphabet of constant symbols occurring in $\mathcal{A}$.

In this paper we consider ontologies specified in *DL-Lite*$_\mathcal{A}$, a member of the *DL-Lite* family of tractable Description Logics. *DL-Lite*$_\mathcal{A}$ distinguishes concepts from *value-domains*, which denote sets of (data) values, and roles from *attributes*, which denote binary relations between objects and values. Concepts, roles, attributes, and value-domains in this DL are formed according to the following syntax:

$$\begin{array}{lll} B \longrightarrow A \mid \exists Q \mid \delta(U) & & E \longrightarrow \rho(U) \\ C \longrightarrow B \mid \neg B & & F \longrightarrow \top_D \mid T_1 \mid \cdots \mid T_n \\ Q \longrightarrow P \mid P^- & & V \longrightarrow U \mid \neg U \\ R \longrightarrow Q \mid \neg Q & & \end{array}$$

In such rules, A, P, and U respectively denote an atomic concept (i.e., a concept name), an atomic role (i.e., a role name), and an attribute name, P^- denotes the inverse of an atomic role, whereas B and Q are called basic concept and basic role, respectively. Furthermore, $\delta(U)$ denotes the *domain* of U, i.e., the set of objects that U relates to values; $\rho(U)$ denotes the *range* of U, i.e., the set of values that U relates to objects; $\top_D$ is the universal value-domain; $T_1, \ldots, T_n$ are n pairwise disjoint unbounded value-domains. A *DL-Lite*$_\mathcal{A}$ TBox $\mathcal{T}$ is a finite set of assertions of the form

$$B \sqsubseteq C \qquad Q \sqsubseteq R \qquad E \sqsubseteq F \qquad U \sqsubseteq V \qquad (\mathsf{funct}\ Q) \qquad (\mathsf{funct}\ U)$$

From left to right, the first four assertions respectively denote inclusions between concepts, roles, value-domains, and attributes. In turn, the last two assertions denote functionality on roles and on attributes. In fact, in *DL-Lite*$_\mathcal{A}$ TBoxes we further impose that roles and attributes occurring in functionality assertions cannot be specialized (i.e., they cannot occur in the right-hand side of inclusions). Let B_1 and B_2 be basic concepts, and let Q_1 and Q_2 be basic roles. We call *positive inclusions (PIs)* assertions of the form $B_1 \sqsubseteq B_2$, and of the form $Q_1 \sqsubseteq Q_2$, whereas we call *negative inclusions (NIs)* assertions of the form $B_1 \sqsubseteq \neg B_2$ and $Q_1 \sqsubseteq \neg Q_2$.

A *DL-Lite*$_\mathcal{A}$ ABox $\mathcal{A}$ is a finite set of membership assertions of the forms $A(a)$, $P(a, b)$, and $U(a, v)$, where A, P, and U are as above, a and b belong to Γ_O, the subset of Γ_C containing object constants, and v belongs to Γ_V, the subset of Γ_C containing value constants, where $\{\Gamma_O, \Gamma_V\}$ is a partition of Γ_C.

The semantics of a *DL-Lite*$_\mathcal{A}$ ontology is given in terms of first-order logic (FOL) interpretations $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$. $\Delta^\mathcal{I}$ is a non-empty domain such that $\Delta^\mathcal{I} = \Delta_V \cup \Delta_O^\mathcal{I}$, where $\Delta_O^\mathcal{I}$ is the domain used to interpret object constants in Γ_O, and Δ_V is the fixed domain (disjoint from $\Delta_O^\mathcal{I}$) used to interpret data values. $\cdot^\mathcal{I}$ is an interpretation function defined as follows:

$$
\begin{array}{ll}
A^\mathcal{I} & \subseteq \Delta_O^\mathcal{I} \\
(\delta(U))^\mathcal{I} & = \{\ o \mid \exists v.\, (o, v) \in U^\mathcal{I}\ \} \\
(\exists Q)^\mathcal{I} & = \{\ o \mid \exists o'.\, (o, o') \in Q^\mathcal{I}\ \} \\
(\neg B)^\mathcal{I} & = \Delta_O^\mathcal{I} \setminus B^\mathcal{I} \\
\Gamma_D^\mathcal{I} & = \Delta_V \\
(\rho(U))^\mathcal{I} & = \{\ v \mid \exists o.\, (o, v) \in U^\mathcal{I}\ \}
\end{array}
\qquad
\begin{array}{ll}
P^\mathcal{I} & \subseteq \Delta_O^\mathcal{I} \times \Delta_O^\mathcal{I} \\
(P^-)^\mathcal{I} & = \{\ (o, o') \mid (o', o) \in P^\mathcal{I}\ \} \\
(\neg Q)^\mathcal{I} & = (\Delta_O^\mathcal{I} \times \Delta_O^\mathcal{I}) \setminus Q^\mathcal{I} \\
U^\mathcal{I} & \subseteq \Delta_O^\mathcal{I} \times \Delta_V \\
(\neg U)^\mathcal{I} & = (\Delta_O^\mathcal{I} \times \Delta_V) \setminus U^\mathcal{I}
\end{array}
$$

Notice that each $(T_i)^\mathcal{I}$ and each $(v)^\mathcal{I}$ are the same in every interpretation. An interpretation $\mathcal{I}$ satisfies a concept (resp., role) inclusion assertion $B \sqsubseteq C$ (resp., $Q \sqsubseteq R$) if $B^\mathcal{I} \subseteq C^\mathcal{I}$ (resp., $Q^\mathcal{I} \subseteq R^\mathcal{I}$). Furthermore, a role functionality assertion $(\mathsf{funct}\ Q)$ is satisfied by $\mathcal{I}$ if, for each $o, o', o'' \in \Delta_O^\mathcal{I}$, we have that $(o, o') \in Q^\mathcal{I}$ and $(o, o'') \in Q^\mathcal{I}$ implies $o' = o''$. The semantics for attribute and value-domain inclusion assertions, and for functionality assertions over attributes can be defined analogously. As for the semantics of ABox assertions, we say that $\mathcal{I}$ satisfies the ABox assertions $A(a)$, $P(a, b)$ and $U(a, v)$ if $a^\mathcal{I} \in A^\mathcal{I}$, $(a^\mathcal{I}, b^\mathcal{I}) \in P^\mathcal{I}$ and $(a^\mathcal{I}, v^\mathcal{I}) \in U^\mathcal{I}$, respectively. Furthermore, in *DL-Lite*$_\mathcal{A}$ the Unique Name Assumption (UNA) is adopted, i.e., in every interpretation $\mathcal{I}$, and for every pair $c_1, c_2 \in \Gamma_C$, if $c_1 \neq c_2$ then $c_1^\mathcal{I} \neq c_2^\mathcal{I}$.

We denote with *Mod*$(\mathcal{O})$ the set of models of an ontology $\mathcal{O}$, i.e., the set of FOL interpretations that satisfy both TBox and ABox assertions in $\mathcal{O}$. As usual, an ontology $\mathcal{O}$ entails a FOL sentence ϕ, denoted $\mathcal{O} \models \phi$, if $\phi^\mathcal{I}$ is true in every $\mathcal{I} \in$ *Mod*$(\mathcal{O})$.

An atomic concept A in $\mathcal{T}$ is unsatisfiable if $\mathcal{T} \models A \sqsubseteq \neg A$, i.e., if the interpretation of A is empty in every model of $\mathcal{T}$. Analogously, we say that an atomic role P is unsatisfiable in $\mathcal{T}$ if $\mathcal{T} \models P \sqsubseteq \neg P$, and a concept attribute U is unsatisfiable in $\mathcal{T}$ if $\mathcal{T} \models U \sqsubseteq \neg U$.

In the following, we we will mainly consider *boolean unions of conjunctive queries (UCQ)* expressed over a *DL-Lite*$_\mathcal{A}$ ontology, i.e., first order sentences of the form $\exists \boldsymbol{y_1}.conj_1(\boldsymbol{t_1}) \vee \cdots \vee \exists \boldsymbol{y_n}.conj_n(\boldsymbol{t_n})$, where $\boldsymbol{y_1}, \ldots, \boldsymbol{y_n}$ are variables, $\boldsymbol{t_1}, \ldots, \boldsymbol{t_n}$ are terms (i.e., constants or variables), such that each variable in $\boldsymbol{t_i}$ occurs also in $\boldsymbol{y_i}$, and

each $conj_i(\boldsymbol{t_i})$ is a conjunction of atoms of the form $A(z)$, $P(z, z')$ and $U(z, z')$ where A is a concept name, P is a role name and U is an attribute name, and z, z' are terms. Each $\exists \boldsymbol{y_i}.conj_i(\boldsymbol{t_i})$ is a *boolean conjunctive query (CQ)* over the ontology. A boolean conjunctive query q evaluates to true over an Abox $\mathcal{A}$ if there exists a substitution σ from the variables in q to constants of $\mathcal{A}$ such that the set containing all and only the atoms in $\sigma(q)$ is contained in $\mathcal{A}$. With a little abuse of notation we write such condition as $\sigma(q) \subseteq \mathcal{A}$. We call each such $\sigma(q)$ an image of q on $\mathcal{A}$. A boolean UCQ $Q = \bigvee_{i=1}^{n} q_i$ evaluates to true on an ABox $\mathcal{A}$ if there exists $i \in \{1, \ldots n\}$ such that q_i is true on $\mathcal{A}$.

UCQs is a particularly interesting class of queries since entailment of a boolean UCQ by a *DL-Lite*$_\mathcal{A}$ ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ can be reduced to standard evaluation of a FOL query, in fact a UCQs, over the ABox $\mathcal{A}$, i.e., entailment of UCQs in *DL-Lite*$_\mathcal{A}$ is *FOL-rewritable* [5, 19]. More formally, query answering of UCQs is *FOL-rewritable* if, for every union of conjunctive queries q and every *DL-Lite*$_\mathcal{A}$ TBox $\mathcal{T}$, there exists a FOL query q_r, over the alphabet of $\mathcal{T}$, such that for every non-empty ABox $\mathcal{A}$ it holds that $\langle \mathcal{T}, \mathcal{A} \rangle \models q$ if and only if q_r evaluates to true over $\mathcal{A}$, i.e., $\langle \emptyset, \mathcal{A} \rangle \models q$. The query q_r is called the *perfect FOL reformulation* of q w.r.t. $\mathcal{T}$. An algorithm for computing such reformulation, called *PerfectRef*, is provided in [5, 19]. In a nutshell, *PerfectRef* takes as input a UCQ q and a *DL-Lite*$_\mathcal{A}$ TBox $\mathcal{T}$ and compiles in q the knowledge of $\mathcal{T}$ useful for answering q, returning another UCQs over $\mathcal{T}$ which is the perfect FOL reformulation of q w.r.t. $\mathcal{T}$.

Obviously, if a *DL-Lite*$_\mathcal{A}$ ontology $\mathcal{O}$ is unsatisfiable, query answering is trivialized, and in particular every boolean query over $\mathcal{O}$ is entailed by $\mathcal{O}$.[3] To avoid this, inconsistency-tolerant semantics are needed, which ensure the existence of models even in the presence of contradictions in the ontology. In the following, we consider this problem and study FOL-rewritability of UCQs under some inconsistency-tolerant semantics. All results we achieve on boolean UCQs can be easily extended in the usual way to the presence of free variables in queries (see e.g. [10]).

3 Inconsistency-tolerant semantics

In this section we recall the IAR and ICAR semantics of [15]. Such semantics are based on the notion of *repair*. Intuitively, given a DL ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$, a repair $\mathcal{A}_R$ for $\mathcal{O}$ is an ABox such that the ontology $\langle \mathcal{T}, \mathcal{A}_R \rangle$ is satisfiable under the first-order semantics, and $\mathcal{A}_R$ "minimally" differs from $\mathcal{A}$. The IAR and the ICAR semantics differ indeed on the notion of minimality they adopt. Formally, the IAR semantics is defined as follows:

Definition 1. *Let $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a DL ontology. The* Intersection ABox Repair (IAR) *of $\mathcal{O}$ is the set IAR-Rep$(\mathcal{O})$ obtained by the intersection of all sets $\mathcal{A}'$ of membership assertions such that:*

1. $\mathcal{A}' \subseteq \mathcal{A}$
2. *Mod*$(\langle \mathcal{T}, \mathcal{A}' \rangle) \neq \emptyset$

[3] The algorithm *PerfectRef* given in [5, 19] assumes in fact that the ontology of interest is satisfiable. Extending it to obtain also trivial answers entailed by unsatisfiable ontologies is easy (see, e.g., [4]), but it is not of practical interest.

3. *there does not exist* $\mathcal{A}''$ *such that* $\mathcal{A}' \subset \mathcal{A}'' \subseteq \mathcal{A}$ *and* $Mod(\langle \mathcal{T}, \mathcal{A}'' \rangle) \neq \emptyset$

An interpretation $\mathcal{I}$ is an *Intersection ABox repair model*, or simply an IAR*-model*, of $\mathcal{O}$ if $\mathcal{I} \models \langle \mathcal{T}, \mathcal{A}' \rangle$, with $\mathcal{A}' =$ *IAR-Rep*$(\mathcal{O})$. The set of ABox repair models is denoted by *IAR-Mod*$(\mathcal{O})$. Furthermore, let $\mathcal{O}$ be a DL ontology, and let ϕ be a first-order sentence, we say that ϕ is IAR*-consistently entailed*, or simply IAR*-entailed*, by $\mathcal{O}$, written $\mathcal{O} \models_{IAR} \phi$, if $\mathcal{I} \models \phi$ for every $\mathcal{I} \in$ *IAR-Mod*$(\mathcal{O})$.

As shown in [15], the IAR-semantics is a sound approximation of another semantics, called AR-semantics, whose repairs actually coincide with the subset of the ABox satisfying conditions 1, 2, and 3 of Definition 1. In other words, *IAR-Rep* $= \bigcap_{\mathcal{A}_i \in \textit{AR-Rep}(\mathcal{O})} \mathcal{A}_i$, where *AR-Rep* denotes the set of the AR-repairs. In the same paper, it has been also shown that the AR-semantics, and consequently the IAR-semantics, has the characteristic to be "syntax-dependent", in the sense that two ontologies $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ and $\mathcal{O}' = \langle \mathcal{T}, \mathcal{A}' \rangle$ that differ simply because $\mathcal{A}'$ includes assertions that logically follow, using $\mathcal{T}$, from a consistent subset of $\mathcal{A}$, may give rise to different AR- and IAR- repairs. This may be considered an unwanted behavior in some applications, and has led to the introduction of the CAR-semantics and of its sound approximation called the ICAR-semantics, whose definition is based on the notion of *consistent logical consequences of* $\mathcal{O}$, denoted *clc*$(\mathcal{O})$. Formally, we have that *clc*$(\mathcal{O}) = \{\alpha \mid \alpha \in$ *HB*$(\mathcal{O})$ and there exists $S \subseteq \mathcal{A}$ such that *Mod*$(\langle \mathcal{T}, S \rangle) \neq \emptyset$ and $\langle \mathcal{T}, S \rangle \models \alpha\}$, where *HB*$(\mathcal{O})$ denotes the *Herbrand Base of* $\mathcal{O}$, i.e., the set of ground atoms that can be built over the alphabet of $\mathcal{O}$. With this notion in place we can now recall the definition of Intersection Closed ABox Repair.[4]

Definition 2. *Let* $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ *be a DL ontology. The* Intersection Closed ABox Repair ($ICAR$) *of* $\mathcal{O}$ *is the set ICAR-Rep*$(\mathcal{O})$ *obtained by the intersection of all sets* $\mathcal{A}'$ *of membership assertions such that:*

1. $\mathcal{A}' \subseteq$ *clc*$(\mathcal{O})$
2. *Mod*$(\langle \mathcal{T}, \mathcal{A}' \rangle) \neq \emptyset$
3. *there does not exist* $\mathcal{A}''$ *such that* $\mathcal{A}' \subset \mathcal{A}'' \subseteq$ *clc*$(\mathcal{O})$ *and Mod*$(\langle \mathcal{T}, \mathcal{A}'' \rangle) \neq \emptyset$

The set of $ICAR$-models of an ontology $\mathcal{O}$, denoted *ICAR-Mod*$(\mathcal{O})$, is defined analogously to IAR-models. Also, $ICAR$-entailment, denoted $\models_{ICAR}$, is defined in a way analogous to IAR-entailment.

4 Perfect reformulation of UCQs in *DL-Lite*$_{\mathcal{A}}$

In this section we show that answering UCQs over *DL-Lite*$_{\mathcal{A}}$ is FOL-rewritable under both the IAR and the ICAR semantics. First of all, we notice that a *DL-Lite*$_{\mathcal{A}}$ TBox is always satisfiable, i.e., it admits always a model under standard FOL semantics, and that in a *DL-Lite*$_{\mathcal{A}}$ ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ an inconsistency may arise only for one of the following reasons:

[4] The definition provided here of the CAR-semantics (and hence of the ICAR-semantics) is a slight simplification of the one originally proposed in [15]: this modification, however, does not affect any of the computational results presented in [15].

1. there exist an atomic concept A (resp. an atomic role or an attribute Z) in the TBox alphabet $\Gamma_{\mathcal{O}}$ and a constant d (resp. a pair of constants d_1 and d_2) in the alphabet of constants Γ_C such that $\mathcal{T} \models A \sqsubseteq \neg A$ and $A(d) \in \mathcal{A}$ (resp. $\mathcal{T} \models Z \sqsubseteq \neg Z$ and $Z(d_1, d_2) \in \mathcal{A}$);
2. there exist an atomic role $P \in \Gamma_{\mathcal{O}}$ and a constant $d \in \Gamma_C$ such that $\mathcal{T} \models P \sqsubseteq \neg P^-$ or $\mathcal{T} \models \exists P \sqsubseteq \neg \exists P^-$ and $P(d, d) \in \mathcal{A}$;
3. there exist an attribute $U \in \Gamma_{\mathcal{O}}$, a constant $d \in \Gamma_C$, and a constant $v \in \Gamma_V$ such that $\mathcal{T} \models \rho(U) \sqsubseteq T$, $U(d, v) \in \mathcal{A}$ and the interpretation of v does not belong to the interpretation set of T;
4. there is a negative inclusion γ such that $\mathcal{T} \models \gamma$ and γ is contradicted by assertions of the ABox. For example, $\mathcal{T} \models A \sqsubseteq \neg \exists P$ and $\{A(d), P(d, c)\} \subseteq \mathcal{A}$ (and analogously for all the various possible forms of negative inclusions);
5. there is an atomic role $P \in \Gamma_{\mathcal{O}}$ and constants d, d_1, d_2 belonging to Γ_C such that $(\mathsf{funct}\ P) \in \mathcal{T}$ (resp. $(\mathsf{funct}\ P^-) \in \mathcal{T}$) and $\{P(d, d_1), P(d, d_2)\} \subseteq \mathcal{A}$ (resp. $\{P(d_1, d), P(d_2, d)\} \subseteq \mathcal{A}$);
6. there is an attribute U, a constant $d \in \Gamma_C$ and constants d_1, d_2 belonging to Γ_V such that $(\mathsf{funct}\ U) \in \mathcal{T}$ and $\{U(d, d_1), U(d, d_2)\} \subseteq \mathcal{A}$.

Interestingly, each possible violation of a TBox axiom involves either one membership assertion (cases 1, 2, and 3) or two membership assertions (cases 4, 5, and 6). It can be shown that no other violations than those described above are indeed possible in *DL-Lite*$_{\mathcal{A}}$. We also call each such violation a *clash* in the ontology. In the following, we will not consider violations described at point 3 (type violations). It is indeed possible to easily extend our reformulation technique to deal with this kind of violations, but due to lack of space we omit details on this aspect.

Roughly speaking, in the reformulation of a query q over a *DL-Lite*$_{\mathcal{A}}$ TBox $\mathcal{T}$, we encode into a FOL formula all violations that can involve membership assertions belonging to images of q on any ABox $\mathcal{A}$. Indeed, this can be done by reasoning only on the TBox, and considering each query atom separately. Intuitively, we deal with inconsistency by rewriting each atom at of q into a FOL formula at_r in such a way that at_r evaluates to true over $\mathcal{A}$ only if there exists a substitution σ of the variables in q to constants of $\mathcal{A}$ such that $\sigma(q) \subseteq \mathcal{A}$ and $\sigma(at)$ is not involved in any of the violations of TBox assertions mentioned above.

This inconsistency-driven rewriting is then suitably casted into the final reformulation, which takes into account also positive knowledge of the TBox, i.e., positive inclusions. As we will show later in this section, this can be done by means of the algorithm *PerfectRef* of [5, 19], which is used in the perfect rewriting of UCQs under both the IAR and the ICAR semantics (even though in different ways).

To formalize the above idea, we need to introduce some preliminary definitions. The first definition that we give can be used to establish whether a certain atom is consistent with the TBox axioms (cf. cases 1 and 2 above). We distinguish below the various possible cases. Let A be an atomic concept in $\Gamma_{\mathcal{O}}$ and t a term (i.e., either a constant or a variable symbol), we pose

$$ConsAtom_A^{\mathcal{T}}(t) = \begin{cases} false & \text{if } \mathcal{T} \models A \sqsubseteq \neg A \\ true & \text{otherwise} \end{cases}$$

In words, $ConsAtom^{\mathcal{T}}_A(t)$ is *false* if the concept A is unsatisfiable, *true* otherwise. The analogous holds for an attribute $U \in \Gamma_{\mathcal{O}}$ and terms t and t':

$$ConsAtom^{\mathcal{T}}_U(t,t') = \begin{cases} \mathit{false} & \text{if } \mathcal{T} \models U \sqsubseteq \neg U \\ \mathit{true} & \text{otherwise} \end{cases}$$

For an atomic role $P \in \Gamma_{\mathcal{O}}$ and terms t, t', we instead have:

$$ConsAtom^{\mathcal{T}}_P(t,t') = \begin{cases} \mathit{false} & \text{if } \mathcal{T} \models P \sqsubseteq \neg P \\ t \neq t' & \text{if } \mathcal{T} \models P \sqsubseteq \neg P^- \vee \mathcal{T} \models \exists P \sqsubseteq \neg \exists P^- \\ \mathit{true} & \text{otherwise} \end{cases}$$

Here, besides taking into account the case when P is an unsatisfiable role, we need to also deal with violations caused by an assertion of the form $P(a,a)$ when P is antisymmetric, which is considered by the second option. The conditions generated by $ConsAtom$ will be used in our reformulation in conjunction with the corresponding query atoms. For example, given the atom $\exists x, y.P(x,y)$ and the TBox $\mathcal{T} = \{\exists P \sqsubseteq \neg \exists P^-\}$, we have that $ConsAtom^{\mathcal{T}}_P(x,y) = x \neq y$, which is used to generate $\exists x, y.P(x,y) \wedge x \neq y$. A membership assertion of the form $P(d,d)$ can be an image for $\exists x, y.P(x,y)$ in any ABox, but not for $\exists x, y.P(x,y) \wedge x \neq y$, i.e., through the rewriting we are able to filter out images of q that are inconsistent with the TBox.

Now we deal with possible clashes involving negative inclusions, which are also called *disjointnesses*. Let B be a basic concept built from an atomic concept or an atomic role of $\Gamma_{\mathcal{O}}$, and let t be a term. Then, we define $NotDisjClash^{\mathcal{T}}_B(t)$ as the following FOL formula:

$$\begin{aligned} &\bigwedge_{A \in DisjConcepts(B,\mathcal{T})} \neg(A(t) \wedge ConsAtom^{\mathcal{T}}_A(t)) \wedge \\ &\bigwedge_{P \in DisjRoleDom(B,\mathcal{T})} \neg(\exists y.P(t,y) \wedge ConsAtom^{\mathcal{T}}_P(t,y)) \wedge \\ &\bigwedge_{P \in DisjRoleRan(B,\mathcal{T})} \neg(\exists y.P(y,t) \wedge ConsAtom^{\mathcal{T}}_P(y,t)) \wedge \\ &\bigwedge_{U \in DisjAttrDom(B,\mathcal{T})} \neg(\exists y.U(t,y) \wedge ConsAtom^{\mathcal{T}}_U(t,y)) \end{aligned}$$

where y is a variable symbol such that $y \neq t$ [5], $DisjConcepts$, $DisjRoleDom$, $DisjRoleRan$, and $DisjAttrDom$ are defined as follows:

$$\begin{aligned} &DisjConcepts(B,\mathcal{T}) = \{A \mid A \text{ is an atomic concept of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models B \sqsubseteq \neg A\} \\ &DisjRoleDom(B,\mathcal{T}) = \{P \mid P \text{ is an atomic role of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models B \sqsubseteq \neg \exists P\} \\ &DisjRoleRan(B,\mathcal{T}) = \{P \mid P \text{ is an atomic role of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models B \sqsubseteq \neg \exists P^-\} \\ &DisjAttrDom(B,\mathcal{T}) = \{U \mid U \text{ is an attribute of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models B \sqsubseteq \neg \delta(U)\} \end{aligned}$$

Notice that if t is a variable, $B(t)$ represents the atoms $\exists t.A(t)$, $\exists t, y.P(t,y)$, $\exists y, t.P(y,t)$, or $\exists t, y.U(t,y)$, for B equal to A, $\exists P$, $\exists P^-$, or $\delta(U)$, respectively. The

[5] Notice that, if t is a variable symbol, then t is a free variable in $NotDisjClash^{\mathcal{T}}_B(t)$

case in which t is a constant is analogous. Intuitively, we will use $NotDisjClash_B^{\mathcal{T}}(t)$ to reformulate a query atom of the form $B(t)$, in order to properly manage all possible clashes that can be caused by assertions of an ABox and that involve TBox axioms of the form $B \sqsubseteq \neg B'$, where B' is a basic concept. In $NotDisjClash_B^{\mathcal{T}}(t)$, the use of $ConsAtom$ formulas guarantees that we do not consider as real clashes the clashes on negative inclusions that involve a membership assertion α such that $\langle \mathcal{T}, \{\alpha\}\rangle$ is unsatisfiable. To clarify this point, consider the following example: Given the TBox $\mathcal{T} = \{A_1 \sqsubseteq \neg A_2, A_2 \sqsubseteq \neg A_2\}$ and the ABox $\mathcal{A} = \{A_1(d), A_2(d)\}$, we have that $A_1(d)$ and $A_2(d)$ clashes on $A_1 \sqsubseteq \neg A_2$, but since $A_2(d)$ alone violates $A_2 \sqsubseteq \neg A_2$, the first one is not a "real" clash, indeed both the IAR and the ICAR semantics include $A_1(d)$ in their repair.

Let us now consider disjointness clashes for roles. Let P be a role name from $\Gamma_{\mathcal{O}}$ and let t, t' be terms, we define the formula $NotDisjClash_P^{\mathcal{T}}(t, t')$ as follows:

$$\bigwedge_{S \in DisjRoles(P,\mathcal{T})} \neg(S(t,t') \wedge ConsAtom_S^{\mathcal{T}}(t,t')) \wedge NotDisjClash_{\exists P}^{\mathcal{T}}(t) \wedge$$

$$\bigwedge_{S \in DisjInvRoles(P,\mathcal{T})} \neg(S(t',t) \wedge ConsAtom_S^{\mathcal{T}}(t',t)) \wedge NotDisjClash_{\exists P^-}^{\mathcal{T}}(t')$$

where, again, if either t or t' are variable symbols, then they are free variables, and the sets $DisjRoles(P, \mathcal{T})$ and $DisjInvRoles(P, \mathcal{T})$ are defined as follows:

$$DisjRoles(P,\mathcal{T}) = \{S \mid S \text{ is a role name of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models P \sqsubseteq \neg S\}$$
$$DisjInvRoles(P,\mathcal{T}) = \{S \mid S \text{ is a role name of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models P \sqsubseteq \neg S^-\}.$$

Intuitively, $NotDisjClash_P^{\mathcal{T}}(t, t')$ will be used in the reformulation to deal with possible violations of negative inclusions involving P. This means considering role inclusions, through the sets $DisjRoles(P, \mathcal{T})$ and $DisjInvRoles(P, \mathcal{T})$, and concept inclusions of the form $\exists P \sqsubseteq \neg B$ and of the form $\exists P^- \sqsubseteq \neg B$, through the use of $NotDisjClash_{\exists P}^{\mathcal{T}}(t)$ and $NotDisjClash_{\exists P^-}^{\mathcal{T}}(t')$, respectively. $ConsAtom_S^{\mathcal{T}}(t, t')$ plays here a role analogous to the one played by $ConsAtom$ formulas in $NotDisjClash_B^{\mathcal{T}}(t)$.

Let us now consider disjontnesses on attributes. Let U be an atomic attribute from $\Gamma_{\mathcal{O}}$ and let t, t' be terms. We define $NotDisjClash_U^{\mathcal{T}}(t, t')$ as the following FOL formula:

$$\bigwedge_{T \in DisjAttributes(U,\mathcal{T})} \neg(T(t,t') \wedge ConsAtom_U^{\mathcal{T}}(t,t')) \wedge NotDisjClash_{\delta(U)}^{\mathcal{T}}(t)$$

where, again, if t or t' are variable symbols they are free variables, and $DisjAttributes$ is defined as follows:

$$DisjAttributes(U,\mathcal{T}) = \{W \mid W \text{ is an atomic attribute of } \Gamma_{\mathcal{O}} \text{ and } \mathcal{T} \models U \sqsubseteq \neg W\}$$

Notice that $NotDisjClash_U^{\mathcal{T}}(t, t')$ is similar to $NotDisjClash_R^{\mathcal{T}}(t, t')$, but simpler, according to the syntax of TBox inclusions involving attributes.

Finally, we consider clashes on functionalities and define $NotFunctClash_P^{\mathcal{T}}(t, t')$ as the following FOL formula:

- if $(\mathsf{funct}\ P) \notin \mathcal{T}$ and $(\mathsf{funct}\ P^-) \notin \mathcal{T}$, then $NotFunctClash_P^{\mathcal{T}}(t, t') = true$;

- if (funct P) $\in \mathcal{T}$ and (funct P^-) $\notin \mathcal{T}$, then $NotFunctClash^{\mathcal{T}}_P(t,t') = \neg(\exists y.P(t,y) \wedge y \neq t' \wedge ConsAtom^{\mathcal{T}}_P(t,y))$;
- if (funct P) $\notin \mathcal{T}$ and (funct P^-) $\in \mathcal{T}$, then $NotFunctClash^{\mathcal{T}}_P(t,t') = \neg(\exists y.P(y,t') \wedge y \neq t \wedge ConsAtom^{\mathcal{T}}_P(y,t))$;
- if (funct P) $\in \mathcal{T}$ and (funct P^-) $\in \mathcal{T}$, then $NotFunctClash^{\mathcal{T}}_P(t,t') = \neg(\exists y.P(t,y) \wedge y \neq t' \wedge ConsAtom^{\mathcal{T}}_P(t,y)) \wedge \neg(\exists y.P(y,t') \wedge y \neq t \wedge ConsAtom^{\mathcal{T}}_P(y,t))$.

Analogously, $NotFunctClash^{\mathcal{T}}_U(t,t')$ is as follows:

- if (funct U) $\notin \mathcal{T}$ then $NotFunctClash^{\mathcal{T}}_U(t,t') = true$;
- if (funct U) $\in \mathcal{T}$ then $NotFunctClash^{\mathcal{T}}_U(t,t') = \neg(\exists y.U(t,y) \wedge y \neq t' \wedge ConsAtom^{\mathcal{T}}_U(t,y))$.

As usual, $ConsAtom$ formulas are used in $NotFunctClash$ formulas to guarantee that only "real" clashes on functionalities are taken into account.

We are finally able to define for each *DL-Lite*$_{\mathcal{A}}$ construct the formula that combines together the various formulas we have introduced for dealing with the various possible clashes.

- $NotClash^{\mathcal{T}}_A(t) = NotDisjClash^{\mathcal{T}}_A(t)$ for an atomic concept name A and term t;
- $NotClash^{\mathcal{T}}_Z(t,t') = NotDisjClash^{\mathcal{T}}_Z(t,t') \wedge NotFunctClash^{\mathcal{T}}_Z(t,t')$ for a role or attribute name Z and terms t, t'.

4.1 Perfect reformulation of UCQs in *DL-Lite*$_{\mathcal{A}}$ under the IAR-semantics

Let q be a CQ of the form

$$\exists x_1, \ldots, x_k. \bigwedge_{i=1}^{n} A_i(t_i^1) \wedge \bigwedge_{i=1}^{m} P_i(t_i^2, t_i^3) \wedge \bigwedge_{i=1}^{\ell} U_i(t_i^4, t_i^5) \tag{1}$$

where every A_i is an atomic concept, every P_i is an atomic role, every U_i is an attribute, and every $t_i^1, t_i^2, t_i^3, t_i^4, t_i^5$ is either a constant or a variable x_j with $1 \leq j \leq k$.

Then, we define *IncRewriting*$_{IAR}(q, \mathcal{T})$ as the following FOL sentence

$$\begin{array}{l}
\exists x_1, \ldots, x_k. \bigwedge_{i=1}^{n} A_i(t_i^1) \wedge ConsAtom^{\mathcal{T}}_{A_i}(t_i^1) \wedge NotClash^{\mathcal{T}}_{A_i}(t_i^1) \wedge \\
\bigwedge_{i=1}^{m} P_i(t_i^2, t_i^3) \wedge ConsAtom^{\mathcal{T}}_{P_i}(t_i^2, t_i^3) \wedge NotClash^{\mathcal{T}}_{P_i}(t_i^2, t_i^3) \\
\bigwedge_{i=1}^{\ell} U_i(t_i^4, t_i^5) \wedge ConsAtom^{\mathcal{T}}_{U_i}(t_i^4, t_i^5) \wedge NotClash^{\mathcal{T}}_{U_i}(t_i^4, t_i^5)
\end{array}$$

In words, for each atom $A_i(t_i^1)$ we specify the conditions that each membership assertion in every image of $A_i(t_i^1)$ over an ABox $\mathcal{A}$ has not to be inconsistent with the TBox (condition $ConsAtom^{\mathcal{T}}_{A_i}(t_i^1)$), and has not be involved in any clash with some other assertion of $\mathcal{A}$ on any negative inclusion (condition $NotClash^{\mathcal{T}}_{A_i}(t_i^1)$). Similarly for atoms of the form $P_i(t_i^2, t_i^3)$ and $U_i(t_i^4, t_i^5)$.

Let Q be the UCQ $q_1 \vee \ldots \vee q_n$. Then, we define

$$\mathit{IncRewritingUCQ}_{IAR}(Q, \mathcal{T}) = \bigvee_{i=1}^{n} \mathit{IncRewriting}_{IAR}(q_i, \mathcal{T})$$

We are then able to give our final results on reformulation of UCQs under the IAR-semantics.

Theorem 1. *Let $\mathcal{T}$ be a DL-Lite$_\mathcal{A}$ TBox and let Q be a UCQ. Then, for every ABox $\mathcal{A}$, $\langle \mathcal{T}, \mathcal{A} \rangle \models_{IAR} Q$ iff $\langle \emptyset, \mathcal{A} \rangle \models \mathit{IncRewritingUCQ}_{IAR}(\mathit{PerfectRef}(Q, \mathcal{T}), \mathcal{T})$.*

We call $\mathit{IncRewritingUCQ}_{IAR}(\mathit{PerfectRef}(Q, \mathcal{T}), \mathcal{T})$ the perfect rewriting of Q with respect to $\mathcal{T}$ under the IAR-semantics, and denote it with $\mathit{PerfectRef}_{IAR}(Q, \mathcal{T})$. In the theorem, $\mathit{PerfectRef}(Q, \mathcal{T})$ denotes the perfect reformulation of a UCQ Q with respect to a *DL-Lite*$_\mathcal{A}$ TBox $\mathcal{T}$ under standard semantics [5, 19]. This algorithm returns a UCQ specified over $\mathcal{T}$. Through such reformulation we first preprocess the query according to "positive" knowledge of the TBox, thus obtaining a new UCQ. Such a UCQ is then rewritten according to functionalities and negative inclusions implied by the TBox in order to realize inconsistency tolerance of query answering under the IAR semantics.

The following complexity result is a direct consequence of the above theorem, since establishing whether $\langle \emptyset, \mathcal{A} \rangle \models \mathit{PerfectRef}_{IAR}(Q, \mathcal{T})$ simply amounts to evaluating a FOL query over the ABox $\mathcal{A}$, which is in AC^0 in data complexity, that is, the complexity computed w.r.t. the size of the ABox only.

Corollary 1. *Let $\mathcal{O}$ be a DL-Lite$_\mathcal{A}$ ontology and let Q be a UCQ. Deciding whether $\mathcal{O} \models_{IAR} Q$ is in AC^0 in data complexity.*

4.2 Perfect reformulation of UCQs in *DL-Lite*$_\mathcal{A}$ under $ICAR$-semantics

Given a CQ q of the form (1), and a *DL-Lite*$_\mathcal{A}$ TBox $\mathcal{T}$, we denote by $\mathit{IncRewriting}_{ICAR}(q, \mathcal{T})$ the following FOL sentence

$$\exists x_1, \ldots, x_k. \bigwedge_{i=1}^{n} \mathit{PerfectRef}(A_i(t_i^1), \mathcal{T}) \wedge \mathit{ConsAtom}^{\mathcal{T}}_{A_i}(t_i^1) \wedge \mathit{NotClash}^{\mathcal{T}}_{A_i}(t_i^1) \wedge$$
$$\bigwedge_{i=1}^{m} \mathit{PerfectRef}(P_i(t_i^2, t_i^3)) \wedge \mathit{ConsAtom}^{\mathcal{T}}_{P_i}(t_i^2, t_i^3) \wedge \mathit{NotClash}^{\mathcal{T}}_{P_i}(t_i^2, t_i^3)$$
$$\bigwedge_{i=1}^{\ell} \mathit{PerfectRef}(U_i(t_i^4, t_i^5)) \wedge \mathit{ConsAtom}^{\mathcal{T}}_{U_i}(t_i^4, t_i^5) \wedge \mathit{NotClash}^{\mathcal{T}}_{U_i}(t_i^4, t_i^5)$$

Differently from what we have done for the IAR-semantics, the algorithm *PerfectRef* is used here to rewrite each atom α of q, considering all variables occurring in α as bound terms. This is done to take into account the presence in the ICAR repair of atoms belonging to $\mathit{clc}(\mathcal{O})$.

Let Q be the UCQ $q_1 \vee \ldots \vee q_n$. Then, $\mathit{IncRewritingUCQ}_{ICAR}(Q, \mathcal{T})$ is the FOL sentence

$$\mathit{IncRewritingUCQ}_{ICAR}(Q, \mathcal{T}) = \bigvee_{i=1}^{n} \mathit{IncRewriting}_{ICAR}(q_i, \mathcal{T})$$

Theorem 2. *Let $\mathcal{T}$ be a DL-Lite$_\mathcal{A}$ TBox and let Q be a UCQ. Then, for every ABox $\mathcal{A}$, $\langle \mathcal{T}, \mathcal{A} \rangle \models_{ICAR} Q$ iff $\langle \emptyset, \mathcal{A} \rangle \models$ IncRewritingUCQ$_{ICAR}$(PerfectRef$(Q, \mathcal{T}), \mathcal{T})$.*

We call *IncRewritingUCQ*$_{ICAR}$(*PerfectRef*$(Q, \mathcal{T}), \mathcal{T})$ the perfect rewriting of Q with respect to $\mathcal{T}$ under the ICAR-semantics, and denote it with *PerfectRef*$_{ICAR}(Q, \mathcal{T})$. As for the IAR-semantics, the following corollary is a direct consequence of the above theorem.

Corollary 2. *Let $\mathcal{O}$ be a DL-Lite$_\mathcal{A}$ ontology and let Q be a UCQ. Deciding whether $\mathcal{O} \models_{ICAR} Q$ is in AC^0 in data complexity.*

5 Example

This example aims to highlight the differences between the IAR semantics and the $ICAR$ semantics and to clarify the algorithms for query reformulation presented before. Let us consider a simple *DL-Lite*$_\mathcal{A}$ ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ describing a wine catalogue. The TBox $\mathcal{T}$ is constituted by the following assertions:

RedWine $\sqsubseteq$ Wine	WhiteWine $\sqsubseteq$ Wine	RedWine $\sqsubseteq \neg$WhiteWine
Wine $\sqsubseteq \neg$Beer	Wine $\sqsubseteq \exists$producedBy	$\exists$producedBy $\sqsubseteq$ Wine
$\exists$producedBy$^-$ $\sqsubseteq$ Winery	Wine $\sqsubseteq \neg$Winery	Beer $\sqsubseteq \neg$Winery
$(funct$ producedBy$)$		

In words, $\mathcal{T}$ specifies that red wines (RedWine) and white wines (WhiteWine) are wines (Wine), but red wines are not white wines. Every wine is produced by (producedBy) a winery (Winery). Moreover the role producedBy has Wine as domain and Winery as range, and it is also functional, i.e., every wine can be produced by at most one winery. Finally, a wine is neither a beer (Beer) nor a winery, and a beer is not a winery.

The ABox $\mathcal{A}$ is formed by the following set of assertions:

$\mathcal{A}$: { RedWine(wine$_1$), WhiteWine(wine$_1$), WhiteWine(wine$_2$),
Beer(wine$_3$), producedBy(wine$_3$, winr) }

This ABox states that wine$_1$ is both a red wine and a white wine, wine$_2$ is a white wine, wine$_3$ is a beer, and wine$_3$ is produced by winr. Notice that this implies that wine$_3$ is a wine and that winr is a winery. It is easy to see that $\mathcal{O}$ is unsatisfiable, since wine$_1$ violates the disjointness between red wine and white wine, and since wine$_3$ violates the disjointness between beer and wine.

Suppose we are interested in asking if a wine exists in the catalogue and to know if winr is a winery, that is to evaluate respectively the queries $q = \exists x.\text{Wine}(x)$ and $q' = \text{Winery}(\text{winr})$.

Let us start considering the IAR-semantics. The first step to perform the perfect reformulation of q and q' under the IAR-semantics consists in computing the classical perfect reformulation of q and q' with respect to $\mathcal{T}$. We obtain:

$$\begin{aligned} \textit{PerfectRef}(q, \mathcal{T}) = \; & (\exists x.\text{Wine}(x)) \vee (\exists x.\text{RedWine}(x)) \vee (\exists x.\text{WhiteWine}(x)) \vee \\ & (\exists x, y.\text{producedBy}(x, y)) \\ \textit{PerfectRef}(q', \mathcal{T}) = \; & \text{Winery}(\text{winr}) \vee (\exists x.\text{producedBy}(x, \text{winr})) \end{aligned}$$

The corresponding perfect reformulation of the queries under IAR-semantics are:

$$
\begin{array}{ll}
\textit{PerfectRef}_{IAR}(q, \mathcal{T}) = & \\
(\exists x.\mathsf{Wine}(x) \wedge \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) \wedge \neg(\exists y.\mathsf{producedBy}(y, x) \wedge x \neq y)) \vee & (d1) \\
(\exists x.\mathsf{RedWine}(x) \wedge \neg\mathsf{WhiteWine}(x) \wedge \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) \wedge & (d2) \\
\quad \neg(\exists y.\mathsf{producedBy}(y, x) \wedge x \neq y)) \vee & \\
(\exists x.\mathsf{WhiteWine}(x) \wedge \neg\mathsf{RedWine}(x) \wedge \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) \wedge & (d3) \\
\quad \neg(\exists y.\mathsf{producedBy}(y, x) \wedge x \neq y)) \vee & \\
(\exists x, y.\mathsf{producedBy}(x, y) \wedge x \neq y \wedge & (d4) \\
\quad \neg(\exists z.\mathsf{producedBy}(x, z) \wedge z \neq y \wedge x \neq z) \wedge & \\
\quad \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) \wedge \neg(\exists w.\mathsf{producedBy}(w, x) \wedge w \neq x) \wedge & \\
\quad \neg\mathsf{Beer}(y) \wedge \neg\mathsf{Wine}(y) \wedge \neg(\exists k.\mathsf{producedBy}(y, k) \wedge y \neq k)) &
\end{array}
$$

and

$$
\begin{array}{ll}
\textit{PerfectRef}_{IAR}(q', \mathcal{T}) = & \\
(\mathsf{Winery}(\mathsf{winr}) \wedge \neg\mathsf{Beer}(\mathsf{winr}) \wedge \neg\mathsf{Wine}(\mathsf{winr}) \wedge & (d5) \\
\quad \neg(\exists y.\mathsf{producedBy}(\mathsf{winr}, y) \wedge \mathsf{winr} \neq y)) \vee & \\
(\exists x.\mathsf{producedBy}(x, \mathsf{winr}) \wedge x \neq \mathsf{winr} \wedge & (d6) \\
\quad \neg(\exists z.\mathsf{producedBy}(x, z) \wedge z \neq \mathsf{winr} \wedge x \neq z) \wedge & \\
\quad \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) \wedge \neg(\exists w.\mathsf{producedBy}(w, x) \wedge w \neq x) \wedge & \\
\quad \neg\mathsf{Beer}(\mathsf{winr}) \wedge \neg\mathsf{Wine}(\mathsf{winr}) \wedge \neg(\exists k.\mathsf{producedBy}(\mathsf{winr}, k) \wedge k \neq \mathsf{winr})) &
\end{array}
$$

It is easy to verify that $\langle \emptyset, \mathcal{A} \rangle \models \textit{PerfectRef}_{IAR}(q, \mathcal{T})$, since the disjunct $(d3)$ evaluates to true on the ABox $\mathcal{A}$. On the contrary, the perfect rewriting of q' under the IAR-semantics evaluates to false on $\mathcal{A}$. Observe that this is exactly what we expect since, by Definition 1, we have that the Intersection ABox Repair of $\mathcal{O}$ is the set $\textit{IAR-Rep}(\mathcal{O}) = \{\mathsf{WhiteWine}(\mathsf{wine}_2)\}$.

Let us now show how the queries q and q' are reformulated and evaluated under the $ICAR$-semantics. The perfect reformulation $\textit{PerfectRef}_{ICAR}(q, \mathcal{T})$ corresponds to the following sentence[6]:

$$
\begin{array}{ll}
(\exists x.\mathsf{Wine}(x) \wedge \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) \wedge \neg(\exists y.\mathsf{producedBy}(y, x) \wedge x \neq y)) \vee & (d7) \\
(\exists x.\mathsf{RedWine}(x) \wedge \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) & (d8) \\
\quad \wedge \neg(\exists y.\mathsf{producedBy}(y, x) \wedge x \neq y)) \vee & \\
(\exists x.\mathsf{WhiteWine}(x) \wedge \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) & (d9) \\
\quad \wedge \neg(\exists y.\mathsf{producedBy}(y, x) \wedge x \neq y)) \vee & \\
(\exists x.\mathsf{producedBy}(x, y) \wedge \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) & (d10) \\
\quad \wedge \neg(\exists y.\mathsf{producedBy}(y, x) \wedge x \neq y)) \vee & \\
(\exists x.\mathsf{RedWine}(x) \wedge \neg\mathsf{WhiteWine}(x) \wedge \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) \wedge & (d11) \\
\quad \neg(\exists y.\mathsf{producedBy}(y, x) \wedge x \neq y)) \vee & \\
(\exists x.\mathsf{WhiteWine}(x) \wedge \neg\mathsf{RedWine}(x) \wedge \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) \wedge & (d12) \\
\quad \neg(\exists y.\mathsf{producedBy}(y, x) \wedge x \neq y)) \vee & \\
(\exists x, y.\mathsf{producedBy}(x, y) \wedge x \neq y \wedge & (d13) \\
\quad \neg(\exists y.\mathsf{producedBy}(x,) \wedge z \neq y \wedge x \neq z) \wedge & \\
\quad \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) \wedge \neg(\exists w.\mathsf{producedBy}(w, x) \wedge w \neq x) \wedge & \\
\quad \neg\mathsf{Beer}(y) \wedge \neg\mathsf{Wine}(y) \wedge \neg(\exists k.\mathsf{producedBy}(y, k) \wedge y \neq k)) &
\end{array}
$$

[6] For the sake of exposition, we have slightly transformed the sentence produced by $\textit{PerfectRef}_{ICAR}(q, \mathcal{T})$ by distributing the logic operator $\wedge$ over the logic operator $\vee$.

which evaluates to true on $\mathcal{A}$ since $(d7)$, $(d8)$, and $(d9)$ evaluate to true on $\mathcal{A}$, and therefore $\mathcal{O} \models_{ICAR} q$.

Moreover, we have also that $\mathcal{O} \not\models_{ICAR} q'$. Indeed, $\mathit{PerfectRef}_{ICAR}(q', \mathcal{T})$ corresponds to

$$
\begin{array}{ll}
\mathit{PerfectRef}_{IAR}(q', \mathcal{T}) = & \\
(\mathsf{Winery}(\mathsf{winr}) \wedge \neg\mathsf{Beer}(\mathsf{winr}) \wedge \neg\mathsf{Wine}(\mathsf{winr}) & (d14) \\
\quad \wedge\neg(\exists y.\mathsf{producedBy}(\mathsf{winr}, y) \wedge \mathsf{winr} \neq y))\vee & \\
(\exists x.\mathsf{producedBy}(x, \mathsf{winr}) \wedge x \neq \mathsf{winr})\wedge & (d15) \\
\quad \neg\mathsf{Beer}(\mathsf{winr}) \wedge \neg\mathsf{Wine}(\mathsf{winr}) \wedge \neg(\exists y.\mathsf{producedBy}(\mathsf{winr}, y) \wedge \mathsf{winr} \neq y))\vee & \\
(\exists x.\mathsf{producedBy}(x, \mathsf{winr}) \wedge x \neq \mathsf{winr}\wedge & (d6) \\
\quad \neg(\exists z.\mathsf{producedBy}(x, z) \wedge z \neq \mathsf{winr} \wedge x \neq z)\wedge & \\
\quad \neg\mathsf{Beer}(x) \wedge \neg\mathsf{Winery}(x) \wedge \neg(\exists w.\mathsf{producedBy}(w, x) \wedge w \neq x)\wedge & \\
\quad \neg\mathsf{Beer}(\mathsf{winr}) \wedge \neg\mathsf{Wine}(\mathsf{winr}) \wedge \neg(\exists k.\mathsf{producedBy}(\mathsf{winr}, k) \wedge k \neq \mathsf{winr})) &
\end{array}
$$

where the disjunct $(d15)$, and therefore the reformulated query, evaluates to true on $\mathcal{A}$.

The differences with respect to the IAR-semantics are due to the use in the ICAR-semantics of the consistent logical consequences of $\mathcal{O}$, which is:

$$
\begin{array}{l}
\mathit{clc}(\mathcal{O}) = \{\ \mathsf{RedWine}(\mathsf{wine}_1), \mathsf{WhiteWine}(\mathsf{wine}_1), \mathsf{WhiteWine}(\mathsf{wine}_2), \\
\qquad\quad \mathsf{Beer}(\mathsf{wine}_3), \mathsf{producedBy}(\mathsf{wine}_3, \mathsf{winr}), \mathsf{Wine}(\mathsf{wine}_1), \\
\qquad\quad \mathsf{Wine}(\mathsf{wine}_2), \mathsf{Wine}(\mathsf{wine}_3), \mathsf{Winery}(\mathsf{winr})\ \}
\end{array}
$$

It follows from Definition 2 that the Intersection Closed ABox Repair of $\mathcal{O}$ is the set $\mathit{ICAR\text{-}Rep}(\mathcal{O}) = \{\ \mathsf{WhiteWine}(\mathsf{wine}_2), \mathsf{Wine}(\mathsf{wine}_1), \mathsf{Wine}(\mathsf{wine}_2), \mathsf{Winery}(\mathsf{winr})\}$.

6 Conclusions

In this paper we have shown that inconsistency-tolerant query answering of UCQs for *DL-Lite*$_\mathcal{A}$ ontologies under both the IAR and the ICAR semantics is FOL-rewritable, and therefore it is in the complexity class AC^0 with respect to data complexity. By virtue of this property, tools offering such reasoning task can rely over well-established relational database technology, exactly as it is done in *DL-Lite*$_\mathcal{A}$ for the classical DL semantics.

The results of the present paper actually refine the results given in [15], where answering UCQs over *DL-Lite*$_\mathcal{A}$ ontologies has been shown to be in PTIME in data complexity under both the IAR and ICAR semantics. In [15], it has been also shown (in Lemma 2) that answering atomic ground queries, a.k.a., instance checking, under the ICAR semantics in fact coincides with instance checking under the CAR semantics (cf. Section 2). This actually means that instance checking is FOL-rewritable under the CAR semantics. A direct consequence of this result is that query answering of ground CQs is FOL rewritable under the CAR semantics.

Acknowledgments. This research has been partially supported by the ICT Collaborative Project ACSI (Artifact-Centric Service Interoperation), funded by the EU under FP7 ICT Call 5, 2009.1.2, grant agreement n. FP7-257593, and by Regione Lazio under the project "Integrazione semantica di dati e servizi per le aziende in rete".

References

1. M. Arenas, L. E. Bertossi, and J. Chomicki. Consistent query answers in inconsistent databases. In *Proc. of PODS'99*, pages 68–79, 1999.
2. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2nd edition, 2007.
3. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodriguez-Muro, R. Rosati, M. Ruzzi, and D. F. Savo. The Mastro system for ontology-based data access. *Semantic Web Journal*, 2(1):43–53, 2011.
4. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Data complexity of query answering in description logics. In *Proc. of KR 2006*, pages 260–270, 2006.
5. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. of Automated Reasoning*, 39(3):385–429, 2007.
6. J. Dolby, A. Fokoue, A. Kalyanpur, L. Ma, E. Schonberg, K. Srinivas, and X. Sun. Scalable grounded conjunctive query evaluation over large and expressive knowledge bases. In *Proc. of ISWC 2008*, volume 5318 of *LNCS*, pages 403–418. Springer, 2008.
7. T. Eiter and G. Gottlob. On the complexity of propositional knowledge base revision, updates and counterfactuals. *Artificial Intelligence*, 57:227–270, 1992.
8. A. Fuxman and R. J. Miller. First-order query rewriting for inconsistent databases. *J. of Computer and System Sciences*, 73(4):610–635, 2007.
9. P. Gärdenfors and H. Rott. Belief revision. In *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 4, pages 35–132. Oxford University Press, 1995.
10. B. Glimm, I. Horrocks, C. Lutz, and U. Sattler. Conjunctive query answering for the description logic $\mathcal{SHIQ}$. In *Proc. of IJCAI 2007*, pages 399–404, 2007.
11. L. Grieco, D. Lembo, M. Ruzzi, and R. Rosati. Consistent query answering under key and exclusion dependencies: Algorithms and experiments. In *Proc. of CIKM 2005*, pages 792–799, 2005.
12. P. Haase, F. van Harmelen, Z. Huang, H. Stuckenschmidt, and Y. Sure. A framework for handling inconsistency in changing ontologies. In *Proc. of ISWC 2005*, volume 3729 of *LNCS*, pages 353–367. Springer, 2005.
13. Z. Huang, F. van Harmelen, and A. ten Teije. Reasoning with inconsistent ontologies. In *Proc. of IJCAI 2005*, pages 454–459, 2003.
14. R. Kontchakov, C. Lutz, D. Toman, F. Wolter, and M. Zakharyaschev. The combined approach to query answering in *DL-Lite*. In *Proc. of KR 2010*, 2010.
15. D. Lembo, M. Lenzerini, R. Rosati, M. Ruzzi, and D. F. Savo. Inconsistency-tolerant semantics for description logics. In *Proc. of RR 2010*, 2010.
16. D. Lembo, R. Rosati, and M. Ruzzi. On the first-order reducibility of unions of conjunctive queries over inconsistent databases. In *Proc. of the Int. Workshop on Inconsistency and Incompleteness in Databases (IIDB 2006)*, 2006.
17. Y. Ma and P. Hitzler. Paraconsistent reasoning for OWL 2. In *Proc. of RR 2009*, pages 197–211, 2009.
18. B. Parsia, E. Sirin, and A. Kalyanpur. Debugging OWL ontologies. In *Proc. of WWW 2005*, pages 633–640, 2005.
19. A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati. Linking data to ontologies. *J. on Data Semantics*, X:133–173, 2008.
20. G. Qi and J. Du. Model-based revision operators for terminologies in description logics. In *Proc. of IJCAI 2009*, pages 891–897, 2009.
21. Z. Wang, K. Wang, and R. W. Topor. A new approach to knowledge base revision in *DL-Lite*. In *Proc. of AAAI 2010*. AAAI Press, 2010.