First-Order Expressibility Results for Queries over Inconsistent DL-Lite Knowledge Bases

Meghyn Bienvenu

First-Order Expressibility Results for Queries over Inconsistent DL-Lite Knowledge Bases

Meghyn Bienvenu

CNRS & Université Paris Sud, France
meghyn@lri.fr

1 Introduction

In recent years, there has been growing interest in ontology-based data access, in which information in the ontology is used to derive additional answers to queries posed over instance data. The $DL\text{-}Lite$ family of description logics [3, 2]) is considered especially well-suited for such applications due to the fact that query answering can be performed by first incorporating the relevant information from the ontology into the query, and then posing the modified query to the bare data. This property, known as first-order rewritability, means that query answering over $DL\text{-}Lite$ ontologies has very low data complexity, which is considered key to scalability.

An important problem which arises in ontology-based data access is how to handle inconsistencies. This problem is especially relevant in an information integration setting where the data comes from multiple sources and one generally lacks the ability to modify the data so as to remove the inconsistency. In the database community, the related problem of querying databases which violate integrity constraints has been extensively studied (cf. [4] for a survey) under the name of consistent query answering. The standard approach is based on the notion of a repair, which is a database which satisfies the integrity constraints and is as similar as possible to the original database. Consistent answers are defined as those answers which hold in all repairs. A similar strategy can be used for description logics by replacing the integrity constraints with the ontology.

Consistent query answering for the $DL\text{-}Lite$ family of description logics was recently studied in [8, 7]. Unfortunately, the obtained complexity results are rather negative: consistent query answering is co-NP-hard in data complexity, even for instance queries and the simplest dialect $DL\text{-}Lite_{core}$. In the database community, negative results were also encountered, but this spurred a line of research [5, 6, 9] aimed at identifying cases where consistent query answering is feasible, and in particular, can be done using query rewriting. We propose to carry out a similar investigation for $DL\text{-}Lite$ ontologies, with the aim of better understanding the cases in which query rewriting can be profitably used. In this paper, we make some first steps towards this goal. Specifically, we formulate general conditions which can be used to prove that a consistent rewriting does or does not exist for a given $DL\text{-}Lite_{core}$ TBox and instance query.

The paper is organized as follows. After some preliminaries, we introduce in Sections 3 and 4 the problem of consistent query answering and some useful notions and terminology. Our main results are presented in Sections 4, 5, and 6, where we present general conditions which yield co-NP-hardness, first-order inexpressiblity, or first-order expressiblity of consistent instance checking in $DL\text{-}Lite_{core}$. Finally, in Section 7, we

show that query rewriting is always possible if we adopt a previously studied alternative semantics. Note that proofs have been omitted for lack of space but can be found in [1].

2 Preliminaries

Syntax. $DL\text{-}Lite_{core}$ knowledge bases (KBs) are built up from a set $\mathsf{N_I}$ of constants, called *individuals*, a set $\mathsf{N_C}$ of unary predicates, called *atomic concepts*, and a set $\mathsf{N_R}$ binary predicates, called *atomic roles*. Complex concept and role expressions are constructed using the following syntax:

$$B \rightarrow A \mid \exists R \qquad C \rightarrow B \mid \neg B \qquad R \rightarrow P \mid P^-$$

where $A \in \mathsf{N_C}$ and $P \in \mathsf{N_R}$. Here B (resp. R) denotes a *basic concept* (resp. *basic role*), and C denotes a *general concept*. A *TBox* is a finite set of *inclusions* of the form $B \sqsubseteq C$ (with B, C as above). An *ABox* is a finite set of *assertions* of the form $B(a)$ ($B \in \mathsf{N_C}$) or $R(a, b)$ ($R \in \mathsf{N_R}$) where $a, b \in \mathsf{N_I}$. A KB consists of a TBox and an ABox.

Notational conventions We use $\mathsf{lhs}(\Gamma)$ (resp. $\mathsf{rhs}(\Gamma)$) to refer to the basic concept appearing on the left (resp. right) side of an inclusion Γ, e.g. $\mathsf{lhs}(\exists P \sqsubseteq \neg D) = \exists P$ and $\mathsf{rhs}(\exists P \sqsubseteq \neg D) = D$. We sometimes use R^- to mean P^- if $R = P \in \mathsf{N_R}$ and P if $R = P^-$, and write $R(a, b)$ to mean $P(a, b)$ if $R = P$ and $R(b, a)$ if $R = P^-$.

Semantics An *interpretation* is $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I}$ is a non-empty set and $\cdot^\mathcal{I}$ maps each $a \in \mathsf{N_I}$ to $a^\mathcal{I} \in \Delta^\mathcal{I}$, each $A \in \mathsf{N_C}$ to $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, and each $P \in \mathsf{N_R}$ to $P^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$. The function $\cdot^\mathcal{I}$ is straightforwardly extended to general concepts and roles, e.g. $(\exists S)^\mathcal{I} = \{c \mid \exists d : (c, d) \in S^\mathcal{I}\}$. $\mathcal{I}$ satisfies $G \sqsubseteq H$ if $G^\mathcal{I} \subseteq H^\mathcal{I}$; it satisfies $A(a)$ (resp. $P(a, b)$) if $a^\mathcal{I} \in A^\mathcal{I}$ (resp. $(a^\mathcal{I}, b^\mathcal{I}) \in P^\mathcal{I}$). We write $\mathcal{I} \models \alpha$ if $\mathcal{I}$ satisfies inclusion/assertion α. $\mathcal{I}$ is a *model* of $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ if $\mathcal{I}$ satisfies all inclusions in $\mathcal{T}$ and assertions in $\mathcal{A}$. A KB $\mathcal{K}$ is *satisfiable/consistent* if it has a model; otherwise it is *unsatisfiable/inconsistent* ($\mathcal{K} \models \bot$). We say that $\mathcal{K}$ *entails* α, written $\mathcal{K} \models \alpha$, if every model of $\mathcal{K}$ is a model of α. The *closure of* $\mathcal{T}$, written $cl(\mathcal{T})$, consists of all inclusions which are entailed from $\mathcal{T}$. Given an ABox $\mathcal{A}$, we denote by $\mathcal{I}_\mathcal{A}$ the interpretation which has as its domain the individuals in $\mathcal{A}$ and which makes true precisely the assertions in $\mathcal{A}$.

Queries A *query* is a formula of first-order logic with equality (FOL), whose atoms are of the form $A(t)$, $P(t, t')$, or $t = t'$ with t, t' *terms*, i.e., variables or individuals. *Conjunctive queries* are queries which do not contain $\forall$, $\neg$, or $=$. *Instance queries* (IQs) are queries consisting of a single atom with no variables (i.e. ABox assertions). A *Boolean query* is a query with no free variables. For a Boolean query q, we write $\mathcal{I} \models q$ when q holds in the interpretation $\mathcal{I}$, and $\mathcal{K} \models q$ when $\mathcal{I} \models q$ for all models $\mathcal{I}$ of $\mathcal{K}$.

3 Consistent query answering

The most commonly used approach to query answering over inconsistent KBs is known as *consistent query answering* and relies on the notion of a repair:

Definition 1. *A* repair *of a knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is an inclusion-maximal subset $\mathcal{B}$ of $\mathcal{A}$ consistent with $\mathcal{T}$. We use $Rep(\mathcal{K})$ to denote the set of repairs of $\mathcal{K}$.*

Consistent query answering consists in performing standard query answering on each of the repairs and intersecting the answers. For Boolean queries, we have:

Definition 2. *A Boolean query q is said to be* consistently entailed *from $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, written $\mathcal{K} \models_{cons} q$, if $\mathcal{T}, \mathcal{B} \models q$ for every repair $\mathcal{B} \in Rep(\mathcal{K})$.*

Just as with standard query entailment, we can ask whether consistent query entailment can be tested by rewriting the query and evaluating it over the data.

Definition 3. *A first-order query q' is a* consistent rewriting *of a Boolean query q w.r.t. a TBox $\mathcal{T}$ if for every ABox $\mathcal{A}$, we have $\mathcal{T}, \mathcal{A} \models_{cons} q$ if and only if $\mathcal{I}_\mathcal{A} \models q'$.*

We illustrate the notion of consistent rewriting on an example.

Example 1. Consider the query $q = R(a, b)$ and the TBox $\mathcal{T} = \{ \exists R \sqsubseteq \neg D, \exists R \sqsubseteq \neg\exists S^-, \exists R^- \sqsubseteq \neg B\}$. We claim $q' = R(a, b) \wedge \neg D(a) \wedge \neg\exists x S(x, a) \wedge \neg B(b)$ is a consistent rewriting of q w.r.t. $\mathcal{T}$. To see why, note that if a repair implies q, then it must contain $R(a, b)$. Moreover, if the ABox $\mathcal{A}$ contains any assertion that contradicts $R(a, b)$ then we can build a repair which does not contain $R(a, b)$. Thus, $R(a, b)$ is consistently entailed just in the case that $R(a, b) \in \mathcal{A}$ and there are no assertions in $\mathcal{A}$ which conflict with $R(a, b)$, which is precisely what q' states.

The method used in Example 1 can be generalized to show that a consistent rewriting exists for all role instance queries[1]. Unfortunately, the same is not true for concept IQs. Indeed, in [7], it was shown that consistent instance checking in DL-Lite$_{core}$ is co-NP-hard in data complexity. We present the reduction in the following example.

Example 2. Consider an instance $\varphi = c_1 \wedge \ldots \wedge c_m$ of UNSAT, where each c_i is a propositional clause. Let $v_1, \ldots, v_k$ be the propositional variables appearing in φ. We define the $DL\text{-}Lite_{core}$ knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ as follows:

$$\begin{aligned} \mathcal{T} &= \{ \exists P^- \sqsubseteq \neg\exists N^-, \exists P \sqsubseteq \neg\exists U^-, \quad \exists N \sqsubseteq \neg\exists U^-, \exists U \sqsubseteq A \} \\ \mathcal{A} &= \{ U(a, c_i) \mid 1 \leq i \leq m \} \cup \{ P(c_i, v_j) \mid v_j \in c_i \} \cup \{ N(c_i, v_j) \mid \neg v_j \in c_i \} \end{aligned}$$

It is not hard to verify that φ is unsatisfiable if and only if $\mathcal{K} \models_{cons} A(a)$. The basic idea is that, because of the inclusion $\exists P^- \sqsubseteq \neg\exists N^-$, each repair corresponds to a valuation of the variables, with v_j assigned true if it has an incoming P-edge in the repair.

The focus in this paper will be on distinguishing between hard and easy instances of the consistent query answering problem. More specifically, we will be interested in the problem of deciding for a given TBox and IQ whether a consistent rewriting exists.

4 Causes and conflicts

In formulating our results, it will be convenient to introduce some terminology for referring to assertions which participate in the entailment of another assertion or its negation.

[1] Obviously this is no longer the case if we consider a logic with role inclusions like $DL\text{-}Lite_\mathcal{R}$.

Definition 4. *Let α, β be assertions and Υ an inclusion. We say* β causes (or is a cause of) α *given Υ, written $\beta \overset{\Upsilon}{\longmapsto} \alpha$, if $\{\Upsilon\}, \{\beta\} \models \alpha$. We say* β conflicts with (or is a conflict for) α *given Υ, written $\beta \overset{\Upsilon}{\bullet\!\!\longrightarrow} \alpha$, if $\Upsilon = B_1 \sqsubseteq \neg B_2$ and $\beta \models B_1(a)$ and $\alpha \models B_2(a)$ for some a. Sometimes we omit Υ if its identity is not relevant.*

The following straightforward proposition characterizes consistent instance checking in terms of causes and conflicts.

Proposition 1. *Let $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ be a $DL\text{-}Lite_{core}$ KB and α an instance query. Then $\mathcal{K} \not\models_{cons} \alpha$ if and only if there exists a subset $\mathcal{A}' \subseteq \mathcal{A}$ which is consistent with $\mathcal{T}$ and such that for every $\beta \in \mathcal{A}$ which causes α (given some axiom in $cl(\mathcal{T})$), there is $\gamma \in \mathcal{A}'$ which conflicts with β (given some axiom in $cl(\mathcal{T})$).*

In other words, consistent instance checking comes down to deciding existence of a consistent subset of the ABox which contradicts all causes of the instance query.

We now introduce the notion of a cause-conflict chain. The intuition is as follows. Suppose that we have an assertion μ_0 in the ABox which causes the IQ α. Then to show $\mathcal{K} \not\models_{cons} \alpha$, Proposition 1 says we must select some assertion ρ_0 which conflicts with μ_0. But if ρ_0 conflicts with an assertion λ_1 which is a conflict of another cause μ_1, then this forces us to choose a different conflict ρ_1 for μ_1 which is consistent with ρ_0. The presence of ρ_1 may in turn attack a conflict of a third cause μ_3, leading us to select a conflict ρ_3 for μ_3, and so on.

Definition 5. *A* cause-conflict chain *(for TBox $\mathcal{T}$ and IQ α) is a sequence $\mu_0\rho_0\lambda_1\mu_1\rho_1 \lambda_2\mu_2\rho_2 \ldots \lambda_n\mu_n\rho_n\lambda_{n+1}\mu_{n+1}$ of distinct assertions, together with a sequence $\Upsilon_0\Gamma_0\Sigma_1 \Omega_1\Upsilon_1\Gamma_1\Sigma_2 \ldots \Omega_n\Upsilon_n\Gamma_n\Sigma_{n+1}\Omega_{n+1}\Upsilon_{n+1}$ of inclusions from $cl(\mathcal{T})$, which satisfy:*

- *for every i: $\mu_i \overset{\Upsilon_i}{\longmapsto} \alpha$, $\mu_i \overset{\Gamma_i}{\bullet\!\!\longrightarrow} \rho_i$, $\rho_i \overset{\Sigma_{i+1}}{\bullet\!\!\longrightarrow} \lambda_{i+1}$, and $\mu_i \overset{\Omega_i}{\bullet\!\!\longrightarrow} \lambda_i$*
- *if $j < i$, then we do not have $\mu_i \bullet\!\!\longrightarrow \rho_j$*
- *$\{\rho_0, \rho_1, \ldots, \rho_n\}$ is consistent with $\mathcal{T}$*

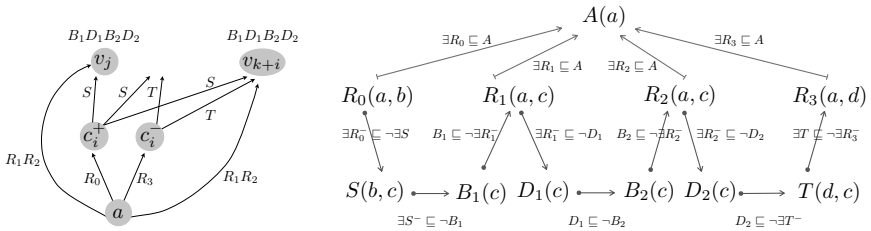
Examples of cause-conflict chains can be found in Figure 1a(b) and 2(b). In the following sections, we will consider particular types of cause-conflict chains and see how they are related to the presence of a consistent rewriting.

5 General co-NP-hardness result

In this section, we formulate a general condition which can be used to show co-NP-hardness of consistent instance checking. We begin by giving a more elaborate reduction from UNSAT, and then we analyze what is needed to make the proof go through.

Example 3. Consider the following TBox $\mathcal{T}$:

$$\{\, \exists R_0 \sqsubseteq A, \exists R_1 \sqsubseteq A, \exists R_2 \sqsubseteq A, \exists R_3 \sqsubseteq A, \exists R_0 \sqsubseteq \neg\exists S, \exists S^- \sqsubseteq \neg B_1, B_1 \sqsubseteq \neg\exists R_1^-,$$
$$\exists R_1^- \sqsubseteq \neg D_1, D_1 \sqsubseteq \neg B_2, B_2 \sqsubseteq \neg\exists R_2^-, \exists R_2^- \sqsubseteq \neg D_2, D_2 \sqsubseteq \neg\exists T^-, \exists T^- \sqsubseteq \neg\exists R_3^- \,\}$$



(a) ABox (b) Cause-conflict chain

Fig. 1: ABox and type-1 cause-conflict chain for Example 3.

We show using a reduction from UNSAT that deciding whether $\mathcal{T}, \mathcal{A} \models_{cons} A(a)$ is co-NP-hard in data complexity. Given a propositional CNF $\varphi = c_1 \wedge \ldots \wedge c_m$ over $v_1, \ldots, v_k$, we define $\mathcal{A}$ as follows (see Figure 1(a) for a pictorial representation):

$$\{\, R_0(a, c_i^+), R_3(a, c_i^-) \,|\, 1 \leq i \leq m \,\} \cup \{\, R_1(a, v_j), R_2(a, v_j) \,|\, 1 \leq j \leq k+m \,\} \cup$$
$$\{\, S(c_i^+, v_j) \,|\, v_j \in c_i \} \cup \{\, T(c_i^-, v_j) \,|\, \neg v_j \in c_i \} \cup \{\, S(c_i^+, v_{k+i}) \,|\, 1 \leq i \leq m \} \cup$$
$$\{\, T(c_i^-, v_{k+i}) \,|\, 1 \leq i \leq m \} \cup \{\, B_1(v_j), B_2(v_j), D_1(v_j), D_2(v_j) \,|\, 1 \leq j \leq k+m \}$$

We show that $\varphi \models \bot$ if and only if $\mathcal{T}, \mathcal{A} \models_{cons} A(a)$. For the first direction, suppose we have a satisfying valuation for φ, and let V be the set of variables which are affected to true. We assume without loss of generality that if a variable v_j appears only positively (resp. negatively) in φ then $v_j \in V$ (resp. $v_j \notin V$). Define the subset $\mathcal{B}$ of $\mathcal{A}$ as follows:

$$\{\, S(c_i^+, v_j), D_1(v_j), D_2(v_j) \in \mathcal{A} \,|\, v_j \in V, 1 \leq j \leq k \} \cup$$
$$\{\, T(c_i^-, v_j), B_1(v_j), B_2(v_j) \in \mathcal{A} \,|\, v_j \notin V, 1 \leq j \leq k \} \cup$$
$$\{\, T(c_i^-, v_{k+i}), B_1(v_{k+i}), B_2(v_{k+i}) \in \mathcal{A} \,|\, \exists v_j \in V \text{ with } v_j \in c_i \} \cup$$
$$\{\, S(c_i^+, v_{k+i}), D_1(v_{k+i}), D_2(v_{k+i}) \in \mathcal{A} \,|\, \forall v_j \in V : v_j \notin c_i \}$$

It is easy to check that $\mathcal{B}$ is consistent with $\mathcal{T}$ and that $\mathcal{T}, \mathcal{B} \not\models A(a)$. It can also be verified that adding any additional assertions from $\mathcal{A}$ to $\mathcal{B}$ leads to a contradiction. In particular, note that either a clause c_i has some positive variable $v_j \in V$, in which case $S(c_i^+, v_j), T(c_i^-, v_{k+i}) \in \mathcal{B}$, or it contains no such v_j, in which case $S(c_i^+, v_{k+i}), T(c_i^-, v_j) \in \mathcal{B}$. In either case, both $R_0(a, c_i^+)$ and $R_3(a, c_i^-)$ conflict with an assertion in $\mathcal{B}$. Thus, $\mathcal{B}$ is a repair of $\mathcal{A}$ w.r.t. $\mathcal{T}$ which does not entail $A(a)$.

For the other direction, let $\mathcal{B}$ be a repair with $\mathcal{T}, \mathcal{B} \not\models A(a)$. It follows that none of the role assertions in $\mathcal{A}$ involving R_0, R_1, R_2, R_3 appear in $\mathcal{B}$. The absence of R_1- and R_2-assertions and the consistency of $\mathcal{B}$ with $\mathcal{T}$ together imply that for each v_j, we have either B_1 and B_2 or both D_1 and D_2. This means each v_j has either incoming S-edges or incoming T-edges, but not both. We create a valuation in which v_j is affected to true if and only if v_j has an incoming S-edge. Clearly if c_i has a positive literal v_j which is affected to true, then it will be satisfied by this valuation. If instead all of the positive literals in c_i are affected to false, then the absence of $R_0(a, c_i^+)$ can only be explained by the presence in $\mathcal{B}$ of the assertion $S(c_i^+, v_{k+i})$. But this implies in turn the absence

of $T(c_i^-, v_{k+i})$ in $\mathcal{B}$. As $R_3(a, c_i^-) \notin \mathcal{B}$, there must be some assertion in $\mathcal{B}$ of the form $T(c_i^-, v_\ell)$ ($1 \leq \ell \leq k$). This means v_ℓ will be affected to false by our valuation, and hence the clause will be satisfied. Thus, the formula φ is satisfiable.

To understand how the preceding reduction can be generalized, it is helpful to consider the cause-conflict chain pictured in Figure 1(b). This chain contains the essential structure used in the reduction, with individuals b, c, and d playing the roles of c_i^+, v_j, and c_ℓ^-. We first notice that at the start and end of the chain, there is a switch of individuals, which corresponds to moving from c_i^+ to v_j and then back to c_ℓ^-. Next remark that in order to show consistency of the constructed $\mathcal{B}$, we needed consistency of the sets of "forward" assertions $\{S(b,c), D_1(c), D_2(c)\}$ and "backward" assertions $\{B_1(c), B_2(c), T(d,c)\}$. Also note that in order to use a repair to construct a satisfying valuation, we had to prove that no v_j had both incoming S- and T-edges. This involved showing that the only way to simultaneously contradict all R_i assertions while retaining consistency was to choose all of the forward (D_i) or all of the backward (B_i) assertions. Key to this reasoning was the fact that for each $R_i(a, v_j)$ assertion, we were forced to choose either $B_i(v_j)$ or $D_i(v_j)$. If we could use some $B_\ell(v_j)$ or $D_\ell(v_j)$ with $\ell \neq j$, the line of reasoning fails. Finally we note that none of the conflicts in the chain involves the query individual a. This is important because if we used some assertion $C(a)$ to contradict $R_i(a, v_j)$, then we would also contradict $R_i(a, v_\ell)$ when $\ell \neq j$, making it impossible to independently choose truth values for each variable.

The preceding analysis leads us to define the notion of a position (to be able to talk about switching to a new individual) and the notion of type-1 cause-conflict chains.

Definition 6. *Concepts of the forms A or $\exists P$ (resp. $\exists P^-$) are said to have* position *1 (resp. 2). An inclusion Υ* begins (resp. concludes) on position *p, written $\mathsf{bpos}(\Upsilon) = p$ (resp. $\mathsf{cpos}(\Upsilon) = p$), if p is the position associated with $\mathsf{lhs}(\Upsilon)$ (resp. $\mathsf{rhs}(\Upsilon)$).*

Definition 7. *A cause-conflict chain for $\mathcal{T}$ and α defined by the sequence of assertions $\mu_0\rho_0\lambda_1\mu_1 \ldots \rho_n\lambda_{n+1}\mu_{n+1}$ and sequence of inclusions $\Upsilon_0\Gamma_0\Sigma_1\Omega_1\Upsilon_1 \ldots \Sigma_{n+1}\Omega_{n+1}\Upsilon_{n+1}$ is said to be of* type-1 *if it satisfies the following conditions:*

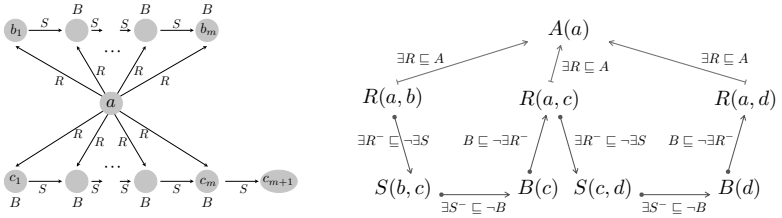
(C1) $\mathsf{bpos}(\Upsilon_i) \neq \mathsf{bpos}(\Gamma_i)$ *and* $\mathsf{bpos}(\Upsilon_i) \neq \mathsf{cpos}(\Omega_i)$ *for all i*
(C2) $\mathsf{cpos}(\Gamma_0) \neq \mathsf{bpos}(\Sigma_1)$
(C3) $\mathsf{cpos}(\Sigma_{n+1}) \neq \mathsf{bpos}(\Omega_{n+1})$
(C4) $\{\lambda_1, \ldots, \lambda_{n+1}\}$ *is consistent with $\mathcal{T}$*
(C5) *if $j > i$, then we do not have $\mu_i \bullet\!\!\longrightarrow \lambda_j$*

Condition C1 of the definition states that the query individual is not used in the conflicts, whereas C2 and C3 make sure there is a switch to a new individual at the start and end of the chain. Condition C4 guarantees consistency of the "backward" conflict assertions, and C5 ensures that when reading the chain from right to left all causes are relevant (i.e. not already contradicted by one of the previous choices).

Example 4. If $B_1 \sqsubseteq \neg B_2$ were added to the TBox from Example 3, then the chain from Figure 1(b) would not be type-1, since $B_1(c)$ and $B_2(c)$ would conflict (violating C4).

The next result shows that the presence of a type-1 cause-conflict chain is sufficient to show co-NP-hardness (and *a fortiori*, the lack of a consistent rewriting). The proof generalizes the reduction from Example 3.

Theorem 1. *If a type-1 cause-conflict chain for $\mathcal{T}$ and α exists, then the problem of deciding whether $\mathcal{T}, \mathcal{A} \models_{cons} \alpha$ is co-NP-hard in data complexity.*



(a) ABox $\mathcal{A}_1$ (b) Cause-conflict chain

Fig. 2: ABox and Type-2 cause-conflict chain used in Example 5.

6 General first-order inexpressibility result

In this section, we use Ehrenfeucht-Fraïssé games to prove nonexistence of a consistent rewriting. As in the previous section, we start with an illustrative example, before formulating the general condition.

Example 5. Consider the following $DL\text{-}Lite_{core}$ TBox $\mathcal{T}$:

$$\mathcal{T} \quad = \quad \{\, \exists R \sqsubseteq A,\, \exists R^- \sqsubseteq \neg\exists S,\, \exists R^- \sqsubseteq \neg B, \exists S^- \sqsubseteq \neg B \,\}$$

We show using Ehrenfeucht-Fraïssé games that there is no consistent first-order rewriting of the query $A(a)$ w.r.t. $\mathcal{T}$. Consider some $k \in \mathbb{N}$, and let $m = 2^k + 1$. We construct two ABoxes $\mathcal{A}_1$ and $\mathcal{A}_2$ as follows ($\mathcal{A}_1$ is pictured in Figure 2(a)):

$$\begin{aligned}
\mathcal{A}_1 \quad &= \quad \{\, R(a,b_i), R(a,c_i), B(c_i), S(c_i,c_{i+1}) \,|\, 1 \leq i \leq m\} \cup \\
&\qquad \{\, B(b_i) \,|\, 2 \leq i \leq m \,\} \cup \{\, S(b_i,b_{i+1}),\, |\, 1 \leq i \leq m-1 \,\} \\
\mathcal{A}_2 \quad &= \quad \mathcal{A}_1 \setminus \{B(c_1)\} \cup \{B(b_1)\}
\end{aligned}$$

We show that $\mathcal{T}, \mathcal{A}_1 \models_{cons} A(a)$ and $\mathcal{T}, \mathcal{A}_2 \not\models_{cons} A(a)$. For the first point, suppose for a contradiction that there is a repair $\mathcal{B}$ of $\mathcal{A}_1$ w.r.t. $\mathcal{T}$ such that $\mathcal{T}, \mathcal{B} \not\models A(a)$. Then there can be no assertions in $\mathcal{B}$ of the form $R(a,b_i)$, and hence each such assertion must provoke a contradiction when added to $\mathcal{B}$. In order for $\mathcal{B} \cup \{R(a,b_1)\}$ to be inconsistent with $\mathcal{T}$, we must have $S(b_1,b_2) \in \mathcal{B}$, as $S(b_1,b_2)$ is the only assertion in $\mathcal{A}$ which conflicts with $R(a,b_1)$. But this means that $B(b_2) \notin \mathcal{B}$, and hence that $S(b_2,b_3) \in \mathcal{B}$, or else we could add $R(a,b_2)$ to $\mathcal{B}$ without provoking a contradiction. Continuing in this manner, we find that $S(b_{m-1},b_m) \in \mathcal{B}$, and so $B(b_m) \notin \mathcal{B}$. But in this case, $\mathcal{B} \cup \{R(a,b_m)\}$ is consistent with $\mathcal{T}$, which contradicts the maximality of $\mathcal{B}$. For the second point, we remark that the set $\mathcal{B} = \{B(b_i), S(c_i,c_{i+1}) \mid 1 \leq i \leq m\}$ is a repair of $\mathcal{A}_2$ w.r.t. $\mathcal{T}$ such that $\mathcal{T}, \mathcal{B} \not\models A(a)$.

We now must show that duplicator has a k-round winning strategy in the Ehrenfeucht-Fraïssé game based on interpretations $\mathcal{I}_{\mathcal{A}_1}$ and $\mathcal{I}_{\mathcal{A}_2}$. The basic idea is as follows (we defer the full argument to [1]). Whenever spoiler selects a point which is "closer" to the side of b_m/c_{m+1} in $\mathcal{I}_{\mathcal{A}_1}$, duplicator responds with the identical point in $\mathcal{I}_{\mathcal{A}_2}$. When spoiler plays "closer" to the b_1/c_1 side, then duplicator plays c_i if b_i was played, and b_i if c_i was played. The important thing is to make sure there is sufficient distance between

the indices j where duplicator copies spoiler and those where he chooses differently. This can be done by keeping track of the rightmost point where the choices differ and the leftmost point where they coincide and ensuring that the distance between these points is always at least 2^{k-i}, where i is the the current round of play.

Figure 2(b) presents a cause-conflict chain for the preceding example. Most of the conditions we identified in the previous section continue to hold for this chain. The only exception is that we do not have a switch of individuals at the end of the chain. Instead, we can remark that the initial cause-type is repeated further down the chain and can be contradicted in the same way, and this is what we use to create the long chain structure required in the proof. This leads us to define a second class of cause-conflict chains, in which we replace C3 with a new condition which captures this repetition.

Definition 8. *A cause-conflict chain for $\mathcal{T}$ and α whose sequence of inclusions is $\Upsilon_0\Gamma_0$ $\Sigma_1\Omega_1\Upsilon_1 \dots \Sigma_{n+1}\Omega_{n+1}\Upsilon_{n+1}$ is said to be* type-2 *if it satisfies C1, C2, C4, C5, and C6:*

(C6) *$\Upsilon_0 = \Upsilon_n$ and $\Gamma_0 = \Gamma_n$*

The following theorem states that type-2 cause-conflict chains witness nonexistence of a consistent rewriting. The proof generalizes the argument outlined in Example 5.

Theorem 2. *If there exists a type-2 cause-conflict chain for $\mathcal{T}$ and α, then there is no consistent first-order rewriting for α w.r.t. $\mathcal{T}$.*

We next establish the relationship between type-1 and type-2 chains.

Theorem 3. *If there exists a type-1 cause-conflict chain for $\mathcal{T}$ and α, then there also exists a type-2 cause-conflict chain. The converse does not hold (assuming P≠NP).*

Proof (Sketch). For the first point, the idea to take a second copy of the type-1 chain, reverse it, and append it to the original. For the second point, we show that consistent instance checking for the TBox and IQ from Example 5 can be done in polynomial time by iteratively applying the following rule: if $R(a, c) \in \mathcal{A}$ and there is no $S(c, d) \in \mathcal{A}$, then delete all incoming S-edges to c. We continue until either we find $R(a, c) \in \mathcal{A}$ such that neither $B(c)$ nor any $S(c, d)$ belongs to $\mathcal{A}$ (in which case $A(a)$ is consistently entailed), or the rule is no longer applicable (and $A(a)$ is not consistently entailed).

7 Rewriting Procedure

In this section, we develop a procedure which is guaranteed to produce a consistent rewriting whenever the TBox $\mathcal{T}$ and query $\alpha = A(a)$ satisfy the following two criteria:

Ordering There exists a total order $<$ on $\mathsf{CauseT}(A)$ such that whenever a cause-conflict chain begins with inclusion $B_1 \sqsubseteq A$, ends with inclusion $B_2 \sqsubseteq A$, and satisfies conditions C1 and C3, we have $B_2 < B_1$.

No loops Every cause-conflict chain for $\mathcal{T}, \alpha$ of length $n+1$ which satisfies $\mathsf{cpos}(\Sigma_i) = \mathsf{bpos}(\Omega_i)$ for every $1 \leq i \leq n+1$ is such that $\Upsilon_i \neq \Upsilon_j$ for all $i \neq j < n+1$.

where $\mathsf{CauseT}(A) = \{D \mid D \sqsubseteq A \in cl(\mathcal{T})\}$ is the set of *cause-types* of A. We define the set of *conflict-types* of A analogously: $\mathsf{ConflT}(A) = \{D \mid D \sqsubseteq \neg A \in cl(\mathcal{T})\}$.

Algorithm 1 `Rewrite`

Input: TBox $\mathcal{T}$, IQ $A(a)$ **Output:** a first-order query φ

Initialize φ to $\bot$ and initialize $\mathcal{G}$ to the set of all tuples $(\mathcal{C}, \mathcal{D})$ which satisfy:

(a) $\mathcal{C} = \{C \in \mathsf{CauseT}(A) \mid \exists D \in \mathcal{D} \text{ with } D \in \mathsf{ConflT}(C)\}$

(b) for all $D \in \mathcal{D}$, there exists $C \in \mathcal{C}$ such that $D \in \mathsf{ConflT}(C)$

(c) there do not exist $D_1, D_2 \in \mathcal{D}$ with $D_2 \in \mathsf{ConflT}(D_1)$

For every $(\mathcal{C}, \mathcal{D}) \in \mathcal{G}$ *// choose which cause-types to treat globally*

Let $\mathcal{D} = \{B_1, \ldots, B_k, \exists P_1, \ldots, \exists P_\ell, \exists P_{\ell+1}^-, \ldots, \exists P_m^-\}$ ($B_i \in \mathsf{N_C}$, $P_i \in \mathsf{N_R}$)

$S = \{B_i(a)\}_{i=1}^k \cup \{P_i(a, w_i)\}_{i=1}^\ell \cup \{P_i(w_i, a)\}_{i=\ell+1}^m$ *// realize concepts in $\mathcal{D}$ at a*

// compute inequalities needed to ensure consistency (treating variables as individuals)

$I = \{v_i \neq v_j \mid v_i, v_j \in \{a, w_1, \ldots, w_m\} \text{ and } \mathcal{T}, S \cup \{v_i = v_j\} \models \bot\}$

$U = \mathsf{CauseT}(A) \setminus \mathcal{C}$ *// cause-types not yet treated*

$\varphi = \varphi \vee \exists w_1 \ldots w_m \bigwedge_{\beta \in S} \beta \wedge \bigwedge_{\gamma \in I} \gamma \wedge \bigwedge_{C \in U} (\forall x \, \texttt{auxRewrite}(\mathcal{T}, A(a), C, x, S))$

Output $\neg\varphi$

Our algorithm `Rewrite` creates a big disjunction, where each disjunct corresponds to a choice of a set of cause-types to be conflicted *globally*, i.e. one single assertion involving the query individual is used to conflict all causes of that type. For each disjunct, we first fix the assertions which realize these global conflicts, and then invoke subroutine `auxRewrite` to build one conjunct per untreated cause-type whose purpose is to see whether for each cause of that type there is an assertion which conflicts with it and can safely be added to the repair under construction. These conjuncts have a tree-like structure whose "paths" are cause-conflict chains which satisfy $\mathsf{cpos}(\Sigma_i) = \mathsf{bpos}(\Omega_i)$ for all i. Property **No Loops** can thus be applied to show that the recursion depth of `auxRewrite` is no more than $|\mathsf{CauseT}(A)| + 1$, ensuring termination. The difficult step in the correctness proof is to show $\mathcal{I}_\mathcal{A} \not\models \texttt{Rewrite}(\mathcal{T}, q)$ implies $\mathcal{T}, \mathcal{A} \not\models_{cons} q$. The basic idea is to use the way the negation of the formula is satisfied to direct our construction of a repair which conflicts with every cause of q. **Ordering** is used to decide in which order we should treat the causes. We illustrate this idea on a concrete example:

Example 6. Let $q = A(a)$ and $\mathcal{T}$ be the following TBox:

$$\{\, \exists R_0 \sqsubseteq A, \exists R_1 \sqsubseteq A, \exists R_2 \sqsubseteq A, \exists R_0^- \sqsubseteq \neg\exists S, \exists S^- \sqsubseteq \neg B_1, B_1 \sqsubseteq \neg\exists R_1^-,$$
$$\exists R_1^- \sqsubseteq \neg D_1, D_1 \sqsubseteq \neg\exists T^-, B_1 \sqsubseteq \neg\exists T^-, \exists T \sqsubseteq \neg\exists R_2^- \,\}$$

It can be verified that the negation of $\texttt{Rewrite}(\mathcal{T}, q)$ consists of a single disjunct:

$$\begin{aligned} & \forall x \, R_0(a, x) \rightarrow \exists y (S(x, y) \wedge (R_1(a, y) \rightarrow D_1(y))) \\ \wedge \quad & \forall x \, R_1(a, x) \rightarrow (B_1(x) \vee D_1(x)) \\ \wedge \quad & \forall x \, R_2(a, x) \rightarrow \exists y (T(x, y) \wedge \neg R_1(a, y)) \end{aligned}$$

We show that if this formula is satisfied in $\mathcal{I}_\mathcal{A}$, then we can construct a repair $\mathcal{B}$ of $\mathcal{A}$ w.r.t. $\mathcal{T}$ which does not entail $A(a)$. First we fix an order on $\mathsf{CauseT}(A)$ satisfying the conditions in **Ordering**: $\exists R_0 < \exists R_2 < \exists R_1$. This means we start by considering causes via $\exists R_0$. If $R_0(a, b) \in \mathcal{A}$, then the first conjunct allows us to find c such that $S(b, c) \in \mathcal{A}$ and $R_1(a, c) \in \mathcal{A}$ implies $D_1(c) \in \mathcal{A}$. We add $S(b, c)$ to $\mathcal{B}$, and also add $D_1(c)$ if $R_1(a, c) \in \mathcal{A}$. We then move on to the next cause-type in the order, $\exists R_2$. If we have $R_2(a, b) \in \mathcal{A}$, then we use the third conjunct to find c such that $T(b, c) \in \mathcal{A}$

Algorithm 2 `auxRewrite`

Input: TBox $\mathcal{T}$, IQ $A(a)$, $C \in \mathsf{CauseT}(A)$, variable x, S set of atoms
Output: a first-order query χ
If $C \in \mathsf{N_C}$, output $\neg C(a)$
Set $\alpha = R(a, x)$, $\chi = \neg\alpha$, and $B = \exists R^-$ where $C = \exists R$ *// R basic role*
For each $D \in \mathsf{ConflT}(B)$ *// Consider different ways to contradict α on x*
  Set $\beta = D(x)$ if $D \in \mathsf{N_C}$ and $\beta = T(x, y)$ [y fresh variable] if $D = \exists T$
  If β is necessarily inconsistent with S given $\mathcal{T}$, exit the for-loop
  Else, let ϵ be the inequalities needed to ensure $\{\beta\} \cup S$ is consistent with $\mathcal{T}$
  // Compute untreated causes which are affected by choice of β
  Initialize Δ to $\emptyset$
  For all $\exists V \in \mathsf{CauseT}(A)$ such that $\mathcal{T}, S \cup \{\beta\} \cup \{V(a, x)\} \not\models \bot$ and
  $\mathsf{ConflT}(\exists V^-) \cap \mathsf{ConflT}(D) \neq \emptyset$
    Add $(\exists V, x)$ to Δ *// need to find conflict for cause $V(a, x)$*
  If $D = \exists T$, then for all $\exists V \in \mathsf{CauseT}(A)$ with $\mathcal{T}, S \cup \{\beta\} \cup \{V(a, y)\} \not\models \bot$
  and $\mathsf{ConflT}(\exists V^-) \cap \mathsf{ConflT}(\exists T^-) \neq \emptyset$
    Add $(\exists V, y)$ to Δ *// need to find conflict for cause $V(a, y)$*
  $\chi = \chi \vee (\exists y)(\beta \wedge \epsilon \wedge \bigwedge_{(H,v)\in\Delta}$ `auxRewrite`$(\mathcal{T}, A(a), H, v, S \cup \{\beta\}))$
Output χ

and $R_1(a, c) \notin \mathcal{A}$, and we add $T(b, c)$ to $\mathcal{B}$. Finally we turn to the final cause-type $\exists R_1$, and let $R_1(a, b) \in \mathcal{A}$. Possibly we have already added $D_1(b)$ when dealing with the first conjunct, in which case we do nothing. Otherwise, because of the second conjunct, we have either $B_1(b) \in \mathcal{A}$ or $D_1(b) \in \mathcal{A}$, which we can add to $\mathcal{B}$. The set $\mathcal{B}$ is still consistent with $\mathcal{T}$ after this step, since if $T(e, b) \in \mathcal{B}$ then we would have $R_1(a, b) \notin \mathcal{A}$, and if $S(e, b) \in \mathcal{B}$, then we would have already added a conflict for $R_1(a, b)$. We have thus found a set $\mathcal{B}$ which is consistent with $\mathcal{T}$ and contradicts every assertion which could cause entailment of $A(a)$. By Proposition 1, we have $\mathcal{T}, \mathcal{A} \not\models_{cons} A(a)$.

Theorem 4. *If a TBox $\mathcal{T}$ and IQ q satisfy conditions **Ordering** and **No Loops**, then* `Rewrite`*$(\mathcal{T}, q)$ terminates and outputs a consistent rewriting of q w.r.t. $\mathcal{T}$.*

Theorem 4 can be used to derive simpler sufficient conditions, like the following:

Corollary 1. `Rewrite`*$(\mathcal{T}, A(a))$ terminates with the correct output if there do not exist basic roles R, S with $\mathcal{T} \models \exists R \sqsubseteq A$ and $\mathcal{T} \models \exists R^- \sqsubseteq \neg\exists S$.*

8 Approximating Consistent Query Answering

In order to obtain a more generally applicable positive result, we consider a sound approximation of consistent query answering, which we term *cautious query answering*.

Definition 9. *A query q is* cautiously entailed *by a knowledge base $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, written $\mathcal{K} \models_{caut} q$, if $\mathcal{T}, \cap_{\mathcal{B}\in Rep(\mathcal{K})}\mathcal{B} \models q$.*

In [7], cautious conjunctive query answering (there called Intersection ABox Repair semantics) was shown to be tractable for $DL\text{-}Lite_{\mathcal{R}}$ KBs. The proposed algorithm first deletes all assertions involved in some conflict, and then queries the resulting ABox. It was left open whether query rewriting techniques could be used instead. We answer this question in the affirmative and thus obtain an improved upper bound of AC_0.

Theorem 5. *Cautious conjunctive query answering is in AC_0 for $DL\text{-}Lite_{core}$.*

Proof (Sketch). Given a $DL\text{-}Lite_{core}$ TBox $\mathcal{T}$ and a CQ q, we first compute (in the standard manner) a UCQ $q' = q_1 \vee ... \vee q_n$ such that for all ABoxes $\mathcal{A}$, we have $\mathcal{T}, \mathcal{A} \models q$ if and only if $\mathcal{I}_\mathcal{A} \models q'$. Then to each disjunct we add the negation of each atomic query which could contradict one of the atoms in the disjunct.

Example 7. If $q = \exists y\, B(x) \wedge R(x, y)$ and $\mathcal{T} = \{A \sqsubseteq B, A \sqsubseteq \exists R, B \sqsubseteq \neg D, \exists R^- \sqsubseteq \neg\exists S^-\}$, standard rewriting yields $A(x) \vee \exists y\, B(x) \wedge R(x, y)$. We then add $\neg\exists z S(z, y)$ to the second disjunct and $\neg D(x)$ to both to obtain the cautious rewriting.

Theorem 5 is easily extended to other $DL\text{-}Lite$ logics enjoying FO-rewritability.

9 Conclusion and Future Work

In this paper, we took a closer look at the problem of consistent instance checking in $DL\text{-}Lite$ and identified some general conditions which can be used to prove the absence or existence of a consistent rewriting. While our results were formulated for $DL\text{-}Lite_{core}$, we expect they can be easily lifted to more expressive $DL\text{-}Lite$ dialects.

The main objective for future work is to strengthen our results so as to be able to decide for every TBox and instance query whether a consistent rewriting exists. We conjecture that the absence of a type-2 cause-conflict chain is both a necessary and sufficient condition for existence of a consistent rewriting. Extending our investigation to conjunctive queries would be interesting but quite challenging, as it would likely involve confronting longstanding open problems from the database community, where a full characterization of rewritable cases remains elusive [9].

References

1. `http://www.lri.fr/~meghyn/BienvenuDL11-long.pdf`.
2. Allessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. The DL-Lite family and relations. *Journal of Artificial Intelligence Research*, 36:1–69, 2009.
3. Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *Journal of Automated Reasoning*, 39(3):385–429, 2007.
4. Jan Chomicki. Consistent query answering: Five easy pieces. In *Proc. of ICDT*, pages 1–17, 2007.
5. Ariel Fuxman and Renée J. Miller. First-order query rewriting for inconsistent databases. In *Proc. of ICDT*, pages 337–351, 2005.
6. Luca Grieco, Domenico Lembo, Riccardo Rosati, and Marco Ruzzi. Consistent query answering under key and exclusion dependencies: algorithms and experiments. In *Proc. of CIKM*, pages 792–799, 2005.
7. Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. Inconsistency-tolerant semantics for description logics. In *Proc. of RR (Web Reasoning and Rule Systems)*, pages 103–117, 2010.
8. Domenico Lembo and Marco Ruzzi. Consistent query answering over description logic ontologies. In *Proc. of RR (Web Reasoning and Rule Systems)*, pages 194–208, 2007.
9. Jef Wijsen. On the first-order expressibility of computing certain answers to conjunctive queries over uncertain databases. In *Proc. of PODS*, pages 179–190, 2010.