Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI-16)

Query-Driven Repairing of Inconsistent DL-Lite Knowledge Bases

Meghyn Bienvenu
CNRS, Univ. Montpellier, Inria
Montpellier, France

Camille Bourgaux
Univ. Paris-Sud, CNRS
Orsay, France

François Goasdoué
Univ. Rennes 1, CNRS
Lannion, France

Abstract

We consider the problem of query-driven repairing of inconsistent DL-Lite knowledge bases: query answers are computed under inconsistency-tolerant semantics, and the user provides feedback about which answers are erroneous or missing. The aim is to find a set of ABox modifications (deletions and additions), called a repair plan, that addresses as many of the defects as possible. After formalizing this problem and introducing different notions of optimality, we investigate the computational complexity of reasoning about optimal repair plans and propose interactive algorithms for computing such plans. For deletion-only repair plans, we also present a prototype implementation of the core components of the algorithm.

1 Introduction

Ontology-mediated query answering (OMQA) is a promising recent approach to data access in which conceptual knowledge provided by an ontology is exploited when querying incomplete data (see [Bienvenu and Ortiz, 2015] for a survey). As efficiency is a primary concern, significant research efforts have been devoted to identifying ontology languages with favorable computational properties. The DL-Lite family of description logics (DLs) [Calvanese *et al.*, 2007], which underlies the OWL 2 QL profile [Motik *et al.*, 2012], has garnered significant interest as it allows OMQA to be reduced, via query rewriting, to standard database query evaluation.

Beyond efficiency, it is important for OMQA systems to be robust to inconsistencies stemming from errors in the data. Inspired by work on consistent query answering in databases [Bertossi, 2011], several inconsistency-tolerant semantics have been developed for OMQA, with the aim of providing meaningful answers in the presence of inconsistencies. Of particular relevance to the present paper are the *brave semantics* [Bienvenu and Rosati, 2013], which returns all query answers that are supported by some internally consistent set of facts, and the more conservative *IAR semantics* [Lembo *et al.*, 2010] that requires that facts in the support not belong to *any* minimal inconsistent subset. Both semantics have appealing computational properties: for most DL-Lite dialects, including the dialect DL-Lite$_\mathcal{R}$ considered in this paper, conjunctive query answering is tractable in data complexity and can be implemented using query rewriting techniques [Lembo *et al.*, 2011; Bienvenu and Rosati, 2013].

While inconsistency-tolerant semantics are essential for returning useful results when consistency cannot be achieved, they by no means replace the need for tools for improving data quality. That is why in this paper we propose a complementary approach that exploits user feedback about query results to identify and correct errors. We consider the following scenario: a user interacts with an OMQA system, posing conjunctive queries and receiving the results, which are sorted into the possible answers (i.e., those holding under the weaker brave semantics) and the (almost) sure answers (holding under IAR semantics). When reviewing the results, the user can indicate that some of the retrieved answer tuples are erroneous, whereas other tuples should definitely be considered answers. Ideally, the unwanted tuples should not be returned as possible (brave) answers, and all of the desired tuples should be found among the sure (IAR) answers. The aim is thus to construct a set of atomic changes (deletions and additions of facts), called a *repair plan*, that achieves as many of these objectives as possible, subject to the constraint that the changes must be validated by the user.

There are several reasons to use queries to guide the repairing process. First, we note that it is typically impossible (for lack of time or information) to clean the entire dataset, and therefore reasonable to focus the effort on the parts of the data that are most relevant to users' needs. In the database arena, this observation has inspired work on integrating entity resolution into the querying process [Altwaijry *et al.*, 2013]. Second, expert users may have a good idea of which answers are expected for queries concerning their area of expertise, and thus queries provide a natural way of identifying flaws. Indeed, Kontokostas et al. (2014) recently proposed to use queries to search for errors and help evaluate linked data quality. Finally, even non-expert users may notice anomalies when examining query results, and it would be a shame not to capitalize on this information, and in this way, help distribute the costly and time-consuming task of improving data quality, as argued in [Bergman *et al.*, 2015].

The contributions of this paper are as follows. In Section 3, we formalize *query-driven repairing problems* and illustrate the main challenges, in particular, the fact that there may not

exist any repair plan that resolves all identified errors. This leads us to introduce in Section 4 different notions of optimal repair plan. Adopting DL-Lite$_\mathcal{R}$ as the ontology language, we study the complexity of reasoning about the different kinds of optimal repair plan and provide interactive algorithms for constructing such plans. In Section 5, we focus on the important special case of deletion-only repair plans, for which all of the optimality notions coincide. We take advantage of the more restricted search space to improve the general approach, and we analyze the complexity of the decision problems used in our algorithm. Finally, in Section 6, we present preliminary experiments about our implementation of the core components of the algorithm for the deletion-only case. We conclude with a discussion of related and future work.

Omitted proofs and additional information about the experiments can be found in [Bienvenu *et al.*, 2016b].

2 Preliminaries

Following the presentation of [Bienvenu *et al.*, 2016a], we recall the basics of DLs and inconsistency-tolerant semantics.

Syntax A DL *knowledge base (KB)* consists of an ABox and a TBox, both constructed from a set $\mathsf{N_C}$ of *concept names* (unary predicates), a set $\mathsf{N_R}$ of *role names* (binary predicates), and a set $\mathsf{N_I}$ of *individuals* (constants). The *ABox* (dataset) is a finite set of *concept assertions* $A(a)$ and *role assertions* $R(a,b)$, where $A \in \mathsf{N_C}$, $R \in \mathsf{N_R}$, $a, b \in \mathsf{N_I}$. The *TBox* (ontology) is a finite set of axioms whose form depends on the particular DL. In DL-Lite$_\mathcal{R}$, TBox axioms are either *concept inclusions* $B \sqsubseteq C$ or *role inclusions* $P \sqsubseteq S$ built according to the following syntax (where $A \in \mathsf{N_C}$ and $R \in \mathsf{N_R}$):

$$B := A \mid \exists P,\ C := B \mid \neg B,\ P := R \mid R^-,\ S := P \mid \neg P$$

Semantics An *interpretation* has the form $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, 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 $R \in \mathsf{N_R}$ to $R^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$. The function $\cdot^\mathcal{I}$ is extended to general concepts and roles in the standard way, e.g. $(R^-)^\mathcal{I} = \{(d,e) \mid (e,d) \in R^\mathcal{I}\}$ and $(\exists P)^\mathcal{I} = \{d \mid \exists e:\ (d,e) \in P^\mathcal{I}\}$. An interpretation $\mathcal{I}$ satisfies an inclusion $G \sqsubseteq H$ if $G^\mathcal{I} \subseteq H^\mathcal{I}$; it satisfies $A(a)$ (resp. $R(a,b)$) if $a^\mathcal{I} \in A^\mathcal{I}$ (resp. $(a^\mathcal{I}, b^\mathcal{I}) \in R^\mathcal{I}$). We call $\mathcal{I}$ a *model* of $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ if $\mathcal{I}$ satisfies all axioms in $\mathcal{T}$ and assertions in $\mathcal{A}$. A KB is *consistent* if it has a model, and an ABox $\mathcal{A}$ is *$\mathcal{T}$-consistent* if the KB $(\mathcal{T}, \mathcal{A})$ is consistent.

Example 1. As a running example, we consider a simple KB $\mathcal{K}_\mathsf{ex} = (\mathcal{T}_\mathsf{ex}, \mathcal{A}_\mathsf{ex})$ about the university domain that contains concepts for postdoctoral researchers ($\mathsf{Postdoc}$), professors (Pr) of two levels of seniority ($\mathsf{APr}, \mathsf{FPr}$), PhD holders ($\mathsf{PhD}$), and graduate courses (GradC), as well as roles to link advisors to their students (Adv), instructors to their courses (Teach) and student to the courses they attend (TakeC). The ABox $\mathcal{A}_\mathsf{ex}$ provides information about an individual a:

$$\begin{aligned}\mathcal{T}_\mathsf{ex} =& \{\mathsf{Postdoc} \sqsubseteq \mathsf{PhD}, \mathsf{Pr} \sqsubseteq \mathsf{PhD}, \mathsf{Postdoc} \sqsubseteq \neg\mathsf{Pr},\\ & \mathsf{FPr} \sqsubseteq \mathsf{Pr}, \mathsf{APr} \sqsubseteq \mathsf{Pr}, \mathsf{APr} \sqsubseteq \neg\mathsf{FPr}, \exists\mathsf{Adv} \sqsubseteq \mathsf{Pr}\}\\ \mathcal{A}_\mathsf{ex} =& \{\mathsf{Postdoc}(a), \mathsf{APr}(a), \mathsf{Adv}(a,b), \mathsf{Teach}(a,c)\}\end{aligned}$$

Observe that $\mathcal{A}_\mathsf{ex}$ is $\mathcal{T}_\mathsf{ex}$-inconsistent. ◀

Queries We focus on *conjunctive queries* (CQs) which take the form $q(\vec{x}) = \exists \vec{y}\, \psi(\vec{x}, \vec{y})$, where ψ is a conjunction of atoms of the forms $A(t)$ or $R(t,t')$, with t, t' individuals or variables from $\vec{x} \cup \vec{y}$. A CQ is called *Boolean* (BCQ) if it has no free variables (i.e. $\vec{x} = \emptyset$). Given a CQ q with free variables $\vec{x} = (x_1, \ldots, x_k)$ and a tuple of individuals $\vec{a} = (a_1, \ldots, a_k)$, we use $q(\vec{a})$ to denote the BCQ resulting from replacing each x_i by a_i. A tuple $\vec{a}$ is a *certain answer* to q over $\mathcal{K}$, written $\mathcal{K} \models q(\vec{a})$, iff $q(\vec{a})$ holds in every model of $\mathcal{K}$. When we use the generic term *query*, we mean a CQ.

Causes and Conflicts A *cause* for a BCQ q w.r.t. KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is a minimal $\mathcal{T}$-consistent subset $\mathcal{C} \subseteq \mathcal{A}$ such that $(\mathcal{T}, \mathcal{C}) \models q$. We use $\mathsf{causes}(q, \mathcal{K})$ to refer to the set of causes for q. A *conflict* for $\mathcal{K}$ is a minimal $\mathcal{T}$-inconsistent subset of $\mathcal{A}$, and $\mathsf{confl}(\mathcal{K})$ denotes the set of conflicts for $\mathcal{K}$.

When $\mathcal{K}$ is a DL-Lite$_\mathcal{R}$ KB, every conflict for $\mathcal{K}$ has at most two assertions. We can thus define the set of *conflicts of a set of assertions* $\mathcal{C} \subseteq \mathcal{A}$ as follows:

$$\mathsf{confl}(\mathcal{C}, \mathcal{K}) = \{\beta \mid \exists \alpha \in \mathcal{C}, \{\alpha, \beta\} \in \mathsf{confl}(\mathcal{K})\}.$$

Inconsistency-Tolerant Semantics A *repair* of $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is an inclusion-maximal subset of $\mathcal{A}$ that is $\mathcal{T}$-consistent. We consider two previously studied inconsistency-tolerant semantics based upon repairs. Under *IAR semantics*, a tuple $\vec{a}$ is an answer to q over $\mathcal{K}$, written $\mathcal{K} \models_{\text{IAR}} q(\vec{a})$, just in the case that $(\mathcal{T}, \mathcal{B}_\cap) \models q(\vec{a})$, where $\mathcal{B}_\cap$ is the *intersection of all repairs* of $\mathcal{K}$ (equivalently, $\mathcal{B}_\cap$ contains some cause for $q(\vec{a})$). If there exists *some repair* $\mathcal{B}$ such that $(\mathcal{T}, \mathcal{B}) \models q(\vec{a})$ (equivalently: $\mathsf{causes}(q(\vec{a}), \mathcal{K}) \neq \emptyset$), then $\vec{a}$ is an answer to q under *brave semantics*, written $\mathcal{K} \models_{\text{brave}} q(\vec{a})$.

Example 2. There are two repairs of the example KB $\mathcal{K}_\mathsf{ex}$:

$$\{\mathsf{Postdoc}(a), \mathsf{Teach}(a,c)\}$$

$$\{\mathsf{APr}(a), \mathsf{Adv}(a,b), \mathsf{Teach}(a,c)\}$$

Evaluating the queries $q_1 = \exists y \mathsf{Teach}(x,y)$ and $q_2 = \mathsf{Prof}(x)$ on $\mathcal{K}_\mathsf{ex}$ yields the following results: $\mathcal{K}_\mathsf{ex} \models_{\text{IAR}} q_1(a)$, $\mathcal{K}_\mathsf{ex} \not\models_{\text{IAR}} q_2(a)$, but $\mathcal{K}_\mathsf{ex} \models_{\text{brave}} q_2(a)$. Indeed, intersecting the repairs yields $\{\mathsf{Teach}(a,c)\}$, which contains a cause for $q_1(a)$ but no cause for $q_2(a)$. On the other hand, the second repair contains two causes for $q_2(a)$ (namely $\{\mathsf{APr}(a)\}$ and $\{\mathsf{Adv}(a,b)\}$), which shows $\mathcal{K}_\mathsf{ex} \models_{\text{brave}} q_2(a)$. ◀

In DL-Lite$_\mathcal{R}$, CQ answering under IAR or brave semantics is in P w.r.t. data complexity (i.e. in the size of the ABox) [Lembo *et al.*, 2010; Bienvenu and Rosati, 2013].

3 Query-Driven Repairing

A user poses questions to a possibly inconsistent KB and receives the sets of possible answers (i.e. those holding under brave semantics) and almost sure answers (those holding under IAR semantics). When examining the results, he detects some *unwanted answers*, which should not have been retrieved, and identifies *wanted answers*, which should be present. To fix the detected problems and improve the quality of the data, the objective is to modify the ABox in such a way that the unwanted answers do not hold under the (more liberal) brave semantics and the wanted answers hold under the more cautious IAR semantics.

A first way of repairing the data is to delete assertions from the ABox that lead to undesirable consequences, either because they contribute to the derivation of an unwanted answer or because they conflict with causes of some wanted answer.

Example 3. Reconsider the KB $\mathcal{K} = (\mathcal{T}_{\mathsf{ex}}, \mathcal{A}_{\mathsf{ex}})$, and suppose a is an unwanted answer for $\mathsf{Pr}(x)$ but a wanted answer for $\mathsf{PhD}(x)$. Deleting the assertions $\mathsf{APr}(a)$ and $\mathsf{Adv}(a, b)$ achieve the objectives since $(\mathcal{T}_{\mathsf{ex}}, \{\mathsf{Postdoc}(a), \mathsf{Teach}(a, c)\}) \not\models_{\text{brave}} \mathsf{Pr}(a)$ and $(\mathcal{T}_{\mathsf{ex}}, \{\mathsf{Postdoc}(a), \mathsf{Teach}(a, c)\}) \models_{\text{IAR}} \mathsf{PhD}(a)$. ◀

The next example shows that, due to data incompleteness, it can also be necessary to add new assertions.

Example 4. Consider $\mathcal{K} = (\mathcal{T}_{\mathsf{ex}}, \{\mathsf{APr}(a)\})$ with the same wanted and unwanted answers as in Ex. 3. The assertion $\mathsf{APr}(a)$ has to be removed to satisfy the unwanted answer, but then there remains no cause for the wanted answer. This is due to the fact that the only cause of $\mathsf{PhD}(a)$ in $\mathcal{K}$ contains an erroneous assertion: there is no 'good' reason in the initial ABox for $\mathsf{PhD}(a)$ to hold. A solution is for the user to add a cause he knows for $\mathsf{PhD}(a)$, such as $\mathsf{Postdoc}(a)$. ◀

We now provide a formal definition of the query-driven repairing problem investigated in this paper.

Definition 1. A *query-driven repairing problem (QRP)* consists of a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ to repair and two sets $\mathcal{W}, \mathcal{U}$ of BCQs that $\mathcal{K}$ should entail ($\mathcal{W}$) or not entail ($\mathcal{U}$). A *repair plan (for $\mathcal{A}$)* is a pair $\mathcal{R} = (\mathcal{E}_-, \mathcal{E}_+)$ such that $\mathcal{E}_- \subseteq \mathcal{A}$ and $\mathcal{E}_+ \cap \mathcal{A} = \emptyset$; if $\mathcal{E}_+ = \emptyset$, we say that $\mathcal{R}$ is *deletion-only*.

The sets $\mathcal{U}$ and $\mathcal{W}$ correspond to the unwanted and wanted answers in our scenario: $q(\vec{a}) \in \mathcal{U}$ (resp. $\mathcal{W}$) means that $\vec{a}$ is an unwanted (resp. wanted) answer for q. Slightly abusing terminology, we will use the term *unwanted (resp. wanted) answers* to refer to the BCQs in $\mathcal{U}$ (resp. $\mathcal{W}$). We say that a repair plan $(\mathcal{E}_-, \mathcal{E}_+)$ *addresses all defects* of a QRP $(\mathcal{K}, \mathcal{W}, \mathcal{U})$ if the KB $\mathcal{K}' = (\mathcal{T}, (\mathcal{A} \backslash \mathcal{E}_-) \cup \mathcal{E}_+)$ is such that $\mathcal{K}' \models_{\text{IAR}} q$ for every $q \in \mathcal{W}$, and $\mathcal{K}' \not\models_{\text{brave}} q$ for every $q \in \mathcal{U}$.

The next example shows that by considering several answers at the same time, we can exploit the interaction between the different answers to reduce the search space.

Example 5. Consider the KB $\mathcal{K} = (\mathcal{T}_{\mathsf{ex}}, \mathcal{A})$ with ABox $\mathcal{A} = \{\mathsf{Pr}(a), \mathsf{APr}(b), \mathsf{FPr}(b), \mathsf{Teach}(a, c), \mathsf{Teach}(b, c), \mathsf{GrC}(c), \mathsf{TakeC}(s, c)\}$. It is easy to see that $\mathcal{K}$ is inconsistent, and its two repairs are obtained by removing either $\mathsf{APr}(b)$ or $\mathsf{FPr}(b)$. Evaluating the queries $q_1(x) = \mathsf{PhD}(x)$ and $q_2(x) = \exists yz \mathsf{Pr}(x) \wedge \mathsf{Teach}(x, y) \wedge \mathsf{GrC}(y) \wedge \mathsf{TakeC}(z, y)$ over this KB yields:

$$\mathcal{K} \models_{\text{brave}} q_1(b) \quad \mathcal{K} \models_{\text{brave}} q_2(b) \quad \mathcal{K} \models_{\text{IAR}} q_2(a).$$

We consider the QRP $(\mathcal{K}, \mathcal{W}, \mathcal{U})$ with wanted answers $\mathcal{W} = \{q_1(b), q_2(a)\}$ and unwanted answers $\mathcal{U} = \{q_2(b)\}$.

Two deletion-only repair plans address all defects: $\{\mathsf{APr}(b), \mathsf{Teach}(b, c)\}$ and $\{\mathsf{FPr}(b), \mathsf{Teach}(b, c)\}$. Indeed, we must delete exactly one of $\mathsf{APr}(b)$ and $\mathsf{FPr}(b)$ for $q_1(b)$ to be entailed under IAR semantics, and we cannot remove $\mathsf{GrC}(c)$ or $\mathsf{TakeC}(s, c)$ without losing the wanted answer $q_2(a)$. Thus, the only way to get rid of $q_2(b)$ is to delete $\mathsf{Teach}(b, c)$.

If we consider only $\mathcal{U}$ (i.e. $\mathcal{W} = \emptyset$), there are additional possibilities such as $\{\mathsf{GrC}(c)\}$ and $\{\mathsf{TakeC}(s, c)\}$, and there is no evidence that $\mathsf{Teach}(b, c)$ has to be deleted. ◀

If we want to avoid introducing new errors, a fully automated repairing process is impossible: we need the user to validate every assertion that is removed or added in order to remove (resp. add) only assertions that are false (resp. true).

Example 6. Reconsider the problem from Ex. 5, and suppose that the user knows that $\mathsf{TakeC}(s, c)$ is false and every other assertion in $\mathcal{A}$ is true. An automatic repairing will remove the true assertion $\mathsf{Teach}(b, c)$. The problem is due to the absence of a 'good' cause for the wanted answer $q_2(a)$ in $\mathcal{A}$. ◀

Since we will be studying an *interactive* repairing process, in which users must validate changes, we will also need to formalize the user's knowledge. For the purposes of this paper, we assume that the user's knowledge is consistent with the considered TBox $\mathcal{T}$, and so can be captured as a set $\mathcal{M}_{\mathsf{user}}$ of models of $\mathcal{T}$. Instead of using $\mathcal{M}_{\mathsf{user}}$ directly, it will be more convenient to work with the *function* user *induced from* $\mathcal{M}_{\mathsf{user}}$ that assigns truth values to BCQs in the obvious way: $\mathsf{user}(q) = \mathsf{true}$ if q is true in every $\mathcal{I} \in \mathcal{M}_{\mathsf{user}}$, $\mathsf{user}(q) = \mathsf{false}$ if q is false in every $\mathcal{I} \in \mathcal{M}_{\mathsf{user}}$, and $\mathsf{user}(q) = \mathsf{unknown}$ otherwise. We will assume throughout the paper the following *truthfulness condition*: $\mathsf{user}(q) = \mathsf{false}$ for every $q \in \mathcal{U}$, and $\mathsf{user}(q) = \mathsf{true}$ for every $q \in \mathcal{W}$.

We now formalize the requirement that repair plans only contain changes that are sanctioned by the user.

Definition 2. A repair plan $(\mathcal{E}_-, \mathcal{E}_+)$ is *validatable w.r.t.* user[1] just in the case that $\mathsf{user}(\alpha) = \mathsf{false}$ for every $\alpha \in \mathcal{E}_-$ and $\mathsf{user}(\alpha) = \mathsf{true}$ for every $\alpha \in \mathcal{E}_+$.

Unfortunately, it may be the case that there is no validatable repair plan addressing all defects. This may happen, for instance, if the user knows some answer is wrong but cannot pinpoint which assertion is at fault, as we illustrate next.

Example 7. Consider the QRP given by:

$$\mathcal{K} = (\mathcal{T}_{\mathsf{ex}}, \{\mathsf{FPr}(a), \mathsf{Teach}(a, c), \mathsf{GrC}(c)\})$$
$$\mathcal{U} = \{\exists x \mathsf{Pr}(a) \wedge \mathsf{Teach}(a, x) \wedge \mathsf{GrC}(x)\}, \ \mathcal{W} = \{\mathsf{Pr}(a)\}$$

Suppose that $\mathsf{user}(\mathsf{FPr}(a)) = \mathsf{false}$, $\mathsf{user}(\mathsf{Teach}(a, c)) = \mathsf{unknown}$, $\mathsf{user}(\mathsf{GrC}(c)) = \mathsf{unknown}$, $\mathsf{user}(\mathsf{APr}(a)) = \mathsf{true}$. It is not possible to satisfy the wanted and unwanted answers at the same time, since adding the true assertion $\mathsf{APr}(a)$ creates a cause for the unwanted answer that does not contains any assertion α with $\mathsf{user}(\alpha) = \mathsf{false}$: the user does not know which of $\mathsf{Teach}(a, c)$ and $\mathsf{GrC}(c)$ is erroneous. ◀

As validatable repair plans addressing all defects are not guaranteed to exist, our aim will be to find repair plans that are optimal in the sense that they address as many of the defects as possible, subject to the constraint that the changes must be validated by the user.

4 Optimal Repair Plans

To compare repair plans, we consider the answers from $\mathcal{U}$ and $\mathcal{W}$ that are satisfied by the resulting KBs, where:

- $q \in \mathcal{U}$ is *satisfied by* $\mathcal{K}$ if $\mathcal{K} \not\models_{\text{brave}} q$;
- $q \in \mathcal{W}$ is *satisfied by* $\mathcal{K}$ if there exists $\mathcal{C} \in \mathsf{causes}(q, \mathcal{K})$ such that $\mathsf{confl}(\mathcal{C}, \mathcal{K}) = \emptyset$ and there is no $\alpha \in \mathcal{C}$ with $\mathsf{user}(\alpha) = \mathsf{false}$.

[1] In what follows, we often omit 'w.r.t. user' and leave it implicit.

Remark 1. Observe that for $q \in \mathcal{W}$ to be satisfied by $\mathcal{K}$, we require not only that $\mathcal{K} \models_{\text{IAR}} q$, but also that there exists a cause for q that does not contain any assertions known to be false, i.e. $\mathcal{K} \models_{\text{IAR}} q$ should hold 'for a good reason'. We impose this additional requirement to avoid counterintuitive situations, e.g. preferring repair plans that remove fewer false assertions in order to retain a conflict-free (but erroneous) cause for a wanted answer.

We say that a repair plan $\mathcal{R} = (\mathcal{E}_-, \mathcal{E}_+)$ *satisfies* $q \in \mathcal{U} \cup \mathcal{W}$ if the KB $\mathcal{K}_\mathcal{R} = (\mathcal{T}, (\mathcal{A} \backslash \mathcal{E}_-) \cup \mathcal{E}_+)$ satisfies q, and we use $\mathcal{S}(\mathcal{R})$ (resp. $\mathcal{S}_\mathcal{U}(\mathcal{R})$, $\mathcal{S}_\mathcal{W}(\mathcal{R})$) to denote the sets of answers (resp. unwanted answers, wanted answers) satisfied by $\mathcal{R}$.

Two repair plans $\mathcal{R}$ and $\mathcal{R}'$ can be compared w.r.t. the sets of unwanted and wanted answers that they satisfy: for $A \in \{\mathcal{U}, \mathcal{W}\}$, we define the preorder $\preceq_A$ by setting $\mathcal{R} \preceq_A \mathcal{R}'$ iff $\mathcal{S}_A(\mathcal{R}) \subseteq \mathcal{S}_A(\mathcal{R}')$, and obtain the corresponding strict order ($\prec_A$) and equivalence relations ($\sim_A$) in the usual way. If the two criteria are equally important, we can combine them using the Pareto principle: $\mathcal{R} \preceq_{\{\mathcal{U},\mathcal{W}\}} \mathcal{R}'$ iff $\mathcal{R} \preceq_\mathcal{U} \mathcal{R}'$ and $\mathcal{R} \preceq_\mathcal{W} \mathcal{R}'$. Alternatively, we can use the lexicographic method to give priority either to the wanted answers ($\preceq_{\mathcal{W},\mathcal{U}}$) or unwanted answers ($\preceq_{\mathcal{U},\mathcal{W}}$): $\mathcal{R} \preceq_{A,B} \mathcal{R}'$ iff $\mathcal{R} \prec_A \mathcal{R}'$ or $\mathcal{R} \sim_A \mathcal{R}'$ and $\mathcal{R} \preceq_B \mathcal{R}'$, where $\{A, B\} = \{\mathcal{U}, \mathcal{W}\}$.

For each of the preceding preference relations $\preceq$, we can define the corresponding notions of $\preceq$-optimal repair plan.

Definition 3. A repair plan $(\mathcal{E}_-, \mathcal{E}_+)$ is *globally (resp. locally) $\preceq$-optimal w.r.t.* user iff it is validatable w.r.t. user and there is no other validatable repair plan $(\mathcal{E}'_-, \mathcal{E}'_+)$ such that $(\mathcal{E}_-, \mathcal{E}_+) \prec (\mathcal{E}'_-, \mathcal{E}'_+)$ (resp. such that $\mathcal{E}_- \subseteq \mathcal{E}'_-$, $\mathcal{E}_+ \subseteq \mathcal{E}'_+$ and $(\mathcal{E}_-, \mathcal{E}_+) \prec (\mathcal{E}'_-, \mathcal{E}'_+)$).

Globally $\preceq$-optimal repair plans are those that are maximal with respect to the preference relation $\preceq$, whereas locally $\preceq$-optimal repair plans are those that cannot be improved in the $\preceq$ ordering by adding further assertions to $\mathcal{E}_-$ or $\mathcal{E}_+$.

Remark 2. If a repair plan is validatable and addresses all defects of a QRP, then it is globally $\preceq_\mathcal{U}$-optimal. If it additionally satisfies every $q \in \mathcal{W}$ (ensuring that there is a 'good' cause for every $q \in \mathcal{W}$), then it is globally $\preceq$-optimal for every $\preceq \in \{\preceq_\mathcal{W}, \preceq_{\{\mathcal{U},\mathcal{W}\}}, \preceq_{\mathcal{U},\mathcal{W}}, \preceq_{\mathcal{W},\mathcal{U}}\}$.

The following example illustrates the difference between local and global optimality.

Example 8. Consider the QRP $((\mathcal{T}_{\mathsf{ex}}, \mathcal{A}), \mathcal{W}, \mathcal{U})$ where

$$\begin{aligned}
\mathcal{A} =& \{\mathsf{Teach}(a,e), \mathsf{Adv}(a,b), \mathsf{TakeC}(b,c), \mathsf{TakeC}(b,e), \\
& \mathsf{GrC}(e)\} \\
\mathcal{W} =& \{\exists x \mathsf{Teach}(a,x), \exists x \mathsf{takeC}(b,x) \wedge \mathsf{GrC}(x)\} \\
\mathcal{U} =& \{\exists xy \mathsf{Teach}(a,x) \wedge \mathsf{Adv}(a,y) \wedge \mathsf{TakeC}(y,x) \wedge \mathsf{GrC}(x)\}
\end{aligned}$$

Suppose that $\mathsf{user}(\mathsf{Teach}(a,e)) = \mathsf{user}(\mathsf{GrC}(e)) = \mathsf{false}$, $\mathsf{user}(\alpha) = \mathsf{unknown}$ for the other $\alpha \in \mathcal{A}$, and the user knows that $\mathsf{Teach}(a,c)$, $\mathsf{Teach}(a,d)$ and $\mathsf{GrC}(c)$ are true.

It can be verified that the repair plan $\mathcal{R}_1 = (\{\mathsf{Teach}(a,e), \mathsf{GrC}(e)\}, \{\mathsf{Teach}(a,c)\})$ satisfies the first answer in $\mathcal{W}$ and the (only) answer in $\mathcal{U}$. It is locally $\preceq_{\{\mathcal{U},\mathcal{W}\}}$-optimal since the only way to satisfy the second wanted answer would be to add $\mathsf{GrC}(c)$, which would create a cause for the unwanted answer, which could not be repaired by removing additional assertions as the user does not know which of $\mathsf{Adv}(a,b)$ and $\mathsf{takeC}(b,c)$ is false. However, $\mathcal{R}_1$ is not globally $\preceq_{\{\mathcal{U},\mathcal{W}\}}$-optimal because $\mathcal{R}_2 = (\{\mathsf{Teach}(a,e), \mathsf{GrC}(e)\}, \{\mathsf{Teach}(a,d), \mathsf{GrC}(c)\})$ satisfies all answers in $\mathcal{W} \cup \mathcal{U}$. ◀

In order to gain a better understanding of the computational properties of the different ways of ranking repair plans, we study the complexity of deciding if a given repair plan is optimal w.r.t. the different criteria. Since validatability of a repair plan depends on user, in this section, we measure the complexity w.r.t. $|\mathcal{A}|$, $|\mathcal{U}|$, $|\mathcal{W}|$, as well as the size of the set

$$\begin{aligned}
True^{rel}_{\mathsf{user}} = \{&\alpha \in True_{\mathsf{user}} \mid \text{there exists } q \in \mathcal{W} \text{ such that} \\
&\alpha \in \mathcal{C} \text{ for some } \mathcal{C} \in \mathsf{causes}(q, \mathcal{A} \cup True_{\mathsf{user}})\}
\end{aligned}$$

where $True_{\mathsf{user}} = \{\alpha \mid \mathsf{user}(\alpha) = \mathsf{true}\}$. We make the reasonable assumption that $True_{\mathsf{user}}$ (hence $True^{rel}_{\mathsf{user}}$) is finite.

Theorem 1. *Deciding if a repair plan is globally $\preceq$-optimal is co*NP*-complete for $\preceq \in \{\preceq_{\{\mathcal{U},\mathcal{W}\}}, \preceq_{\mathcal{U},\mathcal{W}}, \preceq_{\mathcal{W},\mathcal{U}}\}$, and in* P *for $\preceq \in \{\preceq_\mathcal{W}, \preceq_\mathcal{U}\}$. Deciding if a repair plan is locally $\preceq$-optimal is in* P *for $\preceq \in \{\preceq_\mathcal{U}, \preceq_\mathcal{W}, \preceq_{\{\mathcal{U},\mathcal{W}\}}, \preceq_{\mathcal{U},\mathcal{W}}, \preceq_{\mathcal{W},\mathcal{U}}\}$.*

For the coNP upper bounds, we note that to show that $\mathcal{R}$ is *not* $\preceq$-optimal (for $\preceq \in \{\preceq_{\{\mathcal{U},\mathcal{W}\}}, \preceq_{\mathcal{U},\mathcal{W}}, \preceq_{\mathcal{W},\mathcal{U}}\}$), we can guess another repair plan $\mathcal{R}'$ and verify in P that both plans are validatable and that $\mathcal{R}'$ satisfies more answers than $\mathcal{R}$. The lower bounds are by reduction from (variants of) UNSAT.

To establish the tractability results from Theorem 1, we provide characterizations of optimal plans in terms of the notion of *satisfiability* of answers, defined next.

Definition 4. An answer $q \in \mathcal{U} \cup \mathcal{W}$ is *satisfiable* if there exists a validatable repair plan that satisfies q. We say that q is *satisfiable w.r.t. a validatable repair plan* $\mathcal{R} = (\mathcal{E}_-, \mathcal{E}_+)$ if there exists a validatable repair plan $\mathcal{R}' = (\mathcal{E}'_-, \mathcal{E}'_+)$ such that $\mathcal{E}_- \subseteq \mathcal{E}'_-$, $\mathcal{E}_+ \subseteq \mathcal{E}'_+$, $q \in \mathcal{S}(\mathcal{R}')$, and $\mathcal{R} \preceq_{\{\mathcal{U},\mathcal{W}\}} \mathcal{R}'$.

Proposition 1. *Deciding if an answer is satisfied, satisfiable, or satisfiable w.r.t. a repair plan is in* P.

Combining Prop. 1 with the following characterizations yields polynomial-time procedures for optimality testing.

Proposition 2. *A validatable repair plan $\mathcal{R}$ is:*

- *globally $\preceq_\mathcal{U}$- (resp. $\preceq_\mathcal{W}$-) optimal iff it satisfies every satisfiable $q \in \mathcal{U}$ (resp. $q \in \mathcal{W}$).*
- *locally $\preceq_{\mathcal{U},\mathcal{W}}$-optimal iff it is locally $\preceq_{\{\mathcal{U},\mathcal{W}\}}$-optimal iff it satisfies every $q \in \mathcal{U} \cup \mathcal{W}$ that is satisfiable w.r.t. $\mathcal{R}$.*
- *locally $\preceq_{\mathcal{W},\mathcal{U}}$-optimal iff it satisfies every satisfiable $q \in \mathcal{W}$ and every $q \in \mathcal{U}$ that is satisfiable w.r.t. $\mathcal{R}$.*

Our complexity analysis reveals that the notions of global optimality based upon the preference relations $\preceq_{\{\mathcal{U},\mathcal{W}\}}$, $\preceq_{\mathcal{U},\mathcal{W}}$, and $\preceq_{\mathcal{W},\mathcal{U}}$ have undesirable computational properties: even when provided with all relevant user knowledge, it is intractable to decide whether a given plan is optimal. Moreover, while plans globally $\preceq_\mathcal{U}$- (resp. $\preceq_\mathcal{W}$-) optimal can be interactively constructed in a monotonic fashion by removing further false assertions (resp. and adding further true assertions), building a globally optimal plan for a preference relation that involves both $\mathcal{U}$ and $\mathcal{W}$ may require backtracking over answers already satisfied (cf. the situation in Example 8).

ALGORITHM $\mathsf{OptRP}_{\mathcal{U}}$
Input: QRP $(\mathcal{K}{=}(\mathcal{T},\mathcal{A}), \mathcal{U}, \mathcal{W})$ *Output:* repair plan

A. $\mathcal{E}_- \leftarrow \emptyset, \mathcal{E}_+ \leftarrow \emptyset$

B. Display the assertions of $\bigcup_{q\in\mathcal{U}\cup\mathcal{W}} \mathsf{causes}(q,\mathcal{K})$ and $\bigcup_{q\in\mathcal{W},\mathcal{C}\in\mathsf{causes}(q,\mathcal{K})} \mathsf{confl}(\mathcal{C},\mathcal{K})$
   1. Ask user to mark *all* false (F) and true (T) assertions
   2. $\mathcal{E}_- \leftarrow \mathcal{E}_- \cup F \cup \mathsf{confl}(T,\mathcal{K})$

C. While $\mathcal{W}' = \mathcal{W}\backslash\mathcal{S}_{\mathcal{W}}(\mathcal{E}_-,\mathcal{E}_+) \neq \emptyset$: $q \leftarrow$ first$(\mathcal{W}')$
   1. Ask the user for true assertions T_q (not already provided) to complete (or create) a cause for q
   2. If $T_q = \emptyset$ (nothing to add): $\mathcal{W} \leftarrow \mathcal{W}\backslash\{q\}$, go to C.
   3. $\mathcal{E}_+ \leftarrow \mathcal{E}_+ \cup T_q$, $\mathcal{E}_- \leftarrow \mathcal{E}_- \cup \mathsf{confl}(T_q, (\mathcal{T}, \mathcal{A}\cup T_q))$
   4. Show assertions of every cause $\mathcal{C}$ of q such that $T_q \cap \mathcal{C} \neq \emptyset$ and its conflicts: user indicates false, true assertions F', T': $\mathcal{E}_- \leftarrow \mathcal{E}_- \cup F' \cup \mathsf{confl}(T',\mathcal{K})$
   5. Show assertions of causes of every $q'\in\mathcal{U}$ in $\mathcal{A}\backslash\mathcal{E}_- \cup \mathcal{E}_+$: user gives false assertions F'': $\mathcal{E}_- \leftarrow \mathcal{E}_- \cup F''$
   6. If there is $q'' \in \mathcal{U}$ such that $(\mathcal{T}, \mathcal{A}\backslash\mathcal{E}_-\cup\mathcal{E}_+) \models_{\text{brave}} q''$ and $(\mathcal{T}, \mathcal{A}\backslash\mathcal{E}_-) \not\models_{\text{brave}} q''$: $\mathcal{E}_+ \leftarrow \mathcal{E}_+\backslash T_q$ (revert $\mathcal{E}_+$)

D. Return $(\mathcal{E}_-, \mathcal{E}_+)$

Figure 1: Algorithm for constructing a globally $\preceq_{\mathcal{U}}$ and locally $\preceq_{\{\mathcal{U},\mathcal{W}\}}$-optimal repair plan

For the preceding reasons, we target validatable repair plans that are both *globally optimal for* $\preceq_{\mathcal{U}}$ *or* $\preceq_{\mathcal{W}}$ (depending which is preferred) and *locally optimal for* $\preceq_{\{\mathcal{U},\mathcal{W}\}}$. In Fig. 1, we give an interactive algorithm $\mathsf{OptRP}_{\mathcal{U}}$ for building such a repair plan when $\mathcal{U}$ is preferred; if $\mathcal{W}$ is preferred, we use the algorithm $\mathsf{OptRP}_{\mathcal{W}}$ obtained by removing Step C.6 from $\mathsf{OptRP}_{\mathcal{U}}$. The algorithms terminate provided the user knows only a finite number of assertions that may be inserted. In this case, the algorithms output optimal repair plans:

Theorem 2. *The output of $\mathsf{OptRP}_{\mathcal{U}}$ (resp. $\mathsf{OptRP}_{\mathcal{W}}$) is globally $\preceq_{\mathcal{U}}$ (resp. $\preceq_{\mathcal{W}}$) and locally $\preceq_{\{\mathcal{U},\mathcal{W}\}}$-optimal.*

Proof. We give the proof for $\mathsf{OptRP}_{\mathcal{U}}$. First observe that at every point during the execution of the algorithm, the current repair plan is validatable, since only true assertions are added to $\mathcal{E}_+$ and false assertions are added to $\mathcal{E}_-$ (they are either marked as false by the user or are in conflict with assertions that have been marked as true).

Step B adds to $\mathcal{E}_-$ all assertions known to be false that belong to a cause of some $q \in \mathcal{U}\cup\mathcal{W}$ or a conflict of some cause of $q \in \mathcal{W}$. Thus, at the end of this step, $\mathcal{E}_-$ satisfies every satisfiable answer in $\mathcal{U}$, that is, every answer in $\mathcal{U}$ every cause of which contains at least one false assertion. Hence $(\mathcal{E}_-, \mathcal{E}_+)$ is globally $\preceq_{\mathcal{U}}$-optimal at the end of step B. Moreover, every false assertion that occurs in a cause or conflict of a cause of a wanted answer has been removed, so if $q \in \mathcal{W}$ is not satisfied at this point, then it has no cause without any conflict in $\mathcal{A}\backslash\{\alpha \mid \mathsf{user}(\alpha) = \mathsf{false}\}$.

The purpose of Step C is to add new true assertions to create causes for the wanted answers not satisfied after Step B, while preserving $\mathcal{S}_{\mathcal{U}}(\mathcal{E}_-,\mathcal{E}_+)$. For every $q \in \mathcal{W}$, while q is not satisfied, the user is asked to input true assertions to complete a cause for q in Step C.1. If he is unable to do so, at Step C.2, we remove q from $\mathcal{W}$ (since it cannot be satisfied w.r.t. user); otherwise, we update $\mathcal{E}_-$ and $\mathcal{E}_+$ using T_q (C.3). Note that since T_q contains only true assertions, we can remove its conflicts without affecting already satisfied wanted answers; this step is necessary because T_q may conflict with assertions of $\mathcal{A}$ that are not involved in the causes and conflicts presented at Step B. In Step C.4, we remove false assertions appearing in a new cause for q or its conflicts (such assertions may not have been examined in Step B). Step C.5 removes false assertions of new causes of unwanted answers, and Step C.6 undoes the addition of T_q if it affects $\mathcal{S}_{\mathcal{U}}(\mathcal{E}_-,\mathcal{E}_+)$. Thus, at the end of Step C, for every wanted answer, either it is satisfied, or the user is unable to supply a cause that does not deteriorate $\mathcal{S}_{\mathcal{U}}(\mathcal{E}_-,\mathcal{E}_+)$. It follows that $(\mathcal{E}_-,\mathcal{E}_+)$ is locally $\preceq_{\{\mathcal{U},\mathcal{W}\}}$-optimal. □

5 Optimal Deletion-Only Repair Plans

In this section, we restrict our attention to constructing optimal deletion-only repair plans. In this simpler setting, all of the previously introduced notions of optimality collapse into the one characterized in the following proposition.

Proposition 3. *A validatable deletion-only plan is* optimal *iff it satisfies every $q \in \mathcal{U}$ such that every $\mathcal{C} \in \mathsf{causes}(q,\mathcal{K})$ has $\alpha \in \mathcal{C}$ with $\mathsf{user}(\alpha) = \mathsf{false}$, and every $q \in \mathcal{W}$ for which there exists $\mathcal{C} \in \mathsf{causes}(q,\mathcal{K})$ such that $\mathsf{user}(\alpha) \neq \mathsf{false}$ for every $\alpha \in \mathcal{C}$ and $\mathsf{user}(\beta) = \mathsf{false}$ for every $\beta \in \mathsf{confl}(\mathcal{C},\mathcal{K})$.*

Constructing such repair plans can be done with one of the preceding algorithms, omitting Step C that adds facts. However, it is possible to further assist the user by taking advantage of the fact that subsets of the ABox whose removal addresses all defects of the QRP can be automatically identified, and then interactively transformed into optimal repair plans. We call such subsets *potential solutions*.

An assertion is said to be *relevant* if it appears in a cause of some $q \in \mathcal{U}\cup\mathcal{W}$ or in the conflicts of a cause of some $q \in \mathcal{W}$. If an assertion α appears in every potential solution, either $\mathsf{user}(\alpha) = \mathsf{false}$, or there is no validatable potential solution. We call such assertions *necessarily false*. If α appears in no potential solution, it is necessary to keep it in $\mathcal{A}$ to retrieve some wanted answers under IAR semantics, so either $\mathsf{user}(\alpha) \neq \mathsf{false}$, or it is not possible to satisfy all wanted answers. We call such assertions *necessarily nonfalse*.

When a potential solution does not exist, a *minimal correction subset of wanted answers* (MCSW) is an inclusion-minimal subset $\mathcal{W}' \subseteq \mathcal{W}$ such that removing $\mathcal{W}'$ from $\mathcal{W}$ yields a QRP with a potential solution. Because of the truthfulness condition, we know that the absence of a potential solution means that some wanted answers are supported only by causes containing erroneous assertions (otherwise the wanted and unwanted answers would be contradictory, which would violate the truthfulness condition). Moreover, since removing all such answers from $\mathcal{W}$ yields the existence of a potential solution, there exists a MCSW which contains only such answers, which we call an *erroneous MCSW*. This is why MCSWs can help identify the wanted answers that cannot be satisfied by a deletion-only repair plan.

Theorem 3. *For complexity w.r.t. $|\mathcal{A}|$, $|\mathcal{U}|$ and $|\mathcal{W}|$, deciding if a potential solution exists is* NP*-complete, deciding if an assertion is necessarily (non)false is co*NP*-complete, and deciding if $\mathcal{W}' \subseteq \mathcal{W}$ is a MCSW is* BH_2*-complete.*

The lower bounds are proven by reduction from propositional (un)satisfiability and related problems. For the upper bounds, we construct in polynomial time a propositional CNF $\varphi = \varphi_{\mathcal{U}} \wedge \varphi_{\mathcal{W}}$ with:

$$\varphi_{\mathcal{U}} = \bigwedge_{q \in \mathcal{U}} \bigwedge_{\mathcal{C} \in \mathsf{causes}(q,\mathcal{K})} \bigvee_{\alpha \in \mathcal{C}} x_\alpha$$

$$\varphi_{\mathcal{W}} = \bigwedge_{q \in \mathcal{W}} \bigvee_{\mathcal{C} \in \mathsf{causes}(q,\mathcal{K})} w_{\mathcal{C}} \wedge \bigwedge_{q \in \mathcal{W}} \bigwedge_{\mathcal{C} \in \mathsf{causes}(q,\mathcal{K})} \bigwedge_{\alpha \in \mathcal{C}} \neg w_{\mathcal{C}} \vee \neg x_\alpha \wedge \bigwedge_{q \in \mathcal{W}} \bigwedge_{\mathcal{C} \in \mathsf{causes}(q,\mathcal{K})} \bigwedge_{\beta \in \mathsf{confl}(\mathcal{C},\mathcal{K})} \neg w_{\mathcal{C}} \vee x_\beta$$

which has the following properties:

- there exists a potential solution iff φ is satisfiable (satisfying assignments correspond to potential solutions);
- α is necessarily false iff $\varphi \wedge \neg x_\alpha$ is unsatisfiable;
- α is necessarily nonfalse iff $\varphi \wedge x_\alpha$ is unsatisfiable;
- there exist disjoint subsets S, H of the clauses in φ such that the MCSWs correspond to the *minimal correction subsets* (MCSs) of S w.r.t. H, i.e. the subsets $M \subseteq S$ such that (i) $(S \backslash M) \cup H$ is satisfiable, and (ii) $(S \backslash M') \cup H$ is unsatisfiable for every $M' \subsetneq M$.

We present in Fig. 2 an algorithm OptDRP for computing optimal deletion-only repair plans. Within the algorithm, we denote by $R(\mathcal{K},\mathcal{U},\mathcal{W},\mathcal{A}')$ (resp. $N_{\mathsf{f}}(\mathcal{K},\mathcal{U},\mathcal{W},\mathcal{A}')$, $N_{\neg\mathsf{f}}(\mathcal{K},\mathcal{U},\mathcal{W},\mathcal{A}')$) the set of assertions from $\mathcal{A}' \subseteq \mathcal{A}$ that are relevant (resp. necessarily false, nonfalse) for the QRP $(\mathcal{K},\mathcal{U},\mathcal{W})$ when deletions are allowed only in $\mathcal{A}'$ (the set $\mathcal{A}'$ will be used to store assertions whose truth value is not yet determined). The general idea is that the algorithm incrementally builds a set of assertions that are false according to the user. It aids the user by suggesting assertions to remove, or wanted answers that might not be satisfiable when there is no potential solution, while taking into account the knowledge the user has already provided. If there exists a potential solution, the algorithm computes the necessarily (non)false assertions and asks the user either to validate them or to input false and nonfalse assertions to justify why they cannot be validated, and then to input further true or false assertions if the current set of false assertions does not address all defects. When a potential solution is found, the user has to verify that each wanted answer has a cause that does not contain any false assertion. If there does not exist a potential solution at some point, either initially or after some user inputs, the algorithm looks for an erroneous MCSW by computing all MCSWs, then showing for each of them the assertions involved in the causes of each query of the MCSW. If there is a query which has a cause without any false assertion, the MCSW under examination is not erroneous, nor are the other MCSWs that contain that query. Otherwise, the MCSW is erroneous

ALGORITHM OptDRP
Input: QRP $(\mathcal{K}=(\mathcal{T},\mathcal{A}),\mathcal{U},\mathcal{W})$ *Output:* repair plan
(Note: below $\mathcal{K}$ is a macro for $(\mathcal{T},\mathcal{A}\backslash\mathcal{E}_-)$, using the current $\mathcal{E}_-$.)

A. $\mathcal{K}_0 \leftarrow \mathcal{K}$, $\mathcal{A}' \leftarrow \mathcal{A}$, $\mathcal{E}_- \leftarrow \emptyset$

B. If a potential solution for $(\mathcal{K},\mathcal{U},\mathcal{W})$ exists in $\mathcal{A}'$:
1. $R \leftarrow R(\mathcal{K},\mathcal{U},\mathcal{W},\mathcal{A}')$, $N_{\mathsf{f}} \leftarrow N_{\mathsf{f}}(\mathcal{K},\mathcal{U},\mathcal{W},\mathcal{A}')$, $N_{\neg\mathsf{f}} \leftarrow N_{\neg\mathsf{f}}(\mathcal{K},\mathcal{U},\mathcal{W},\mathcal{A}')$
2. If the user validates $\mathsf{user}(\alpha) = \mathsf{false}$ for every $\alpha \in N_{\mathsf{f}}$ and $\mathsf{user}(\alpha) \neq \mathsf{false}$ for every $\alpha \in N_{\neg\mathsf{f}}$:
   a. $\mathcal{E}_- \leftarrow \mathcal{E}_- \cup N_{\mathsf{f}}$, $\mathcal{A}' \leftarrow \mathcal{A}'\backslash(N_{\mathsf{f}} \cup N_{\neg\mathsf{f}})$
   b. If $\mathcal{E}_-$ is a potential solution for $(\mathcal{K}_0,\mathcal{U},\mathcal{W})$:
      i. For each $q \in \mathcal{W}$: the user gives *all* false assertions $F \subseteq \bigcup_{\mathcal{C} \in \mathsf{causes}(q,\mathcal{K}),\mathsf{confl}(\mathcal{C},\mathcal{K})=\emptyset} \mathcal{C}$, $\mathcal{E}_- \leftarrow \mathcal{E}_- \cup F$
      ii. If $\mathcal{E}_-$ is still a potential solution: output $\mathcal{E}_-$
      iii. Else: $\mathcal{A}' \leftarrow \mathcal{A}'\backslash\mathcal{E}_-$, go to B
   c. Else: user selects *some* $F, T \subseteq R\backslash(N_{\mathsf{f}} \cup N_{\neg\mathsf{f}})$
      i. If $F = T = \emptyset$ (nothing left to input): return $\mathcal{E}_-$
      ii. Else: $\mathcal{E}_- \leftarrow \mathcal{E}_- \cup F \cup \mathsf{confl}(T,\mathcal{K})$, $\mathcal{A}' \leftarrow \mathcal{A}'\backslash(\mathcal{E}_- \cup T)$, go to B
3. Else: user gives $F \subseteq \{\alpha \in N_{\neg\mathsf{f}} \mid \mathsf{user}(\alpha) = \mathsf{false}\}$ and $NF \subseteq \{\alpha \in N_{\mathsf{f}} \mid \mathsf{user}(\alpha) \neq \mathsf{false}\}$ with $F \cup NF \neq \emptyset$, $\mathcal{E}_- \leftarrow \mathcal{E}_- \cup F$, $\mathcal{A}' \leftarrow \mathcal{A}'\backslash(\mathcal{E}_- \cup NF)$

C. Search for a MCSW containing only answers that are supported only by erroneous causes:
1. $\mathcal{M} \leftarrow MCSWs(\mathcal{K},\mathcal{U},\mathcal{W},\mathcal{A}')$ ordered by size
2. While erroneous MCSW not found and $\mathcal{M} \neq \emptyset$:
   a. $M \leftarrow$ first$(\mathcal{M})$
   b. For every $q \in M$:
      i. the user selects $F, T \subseteq \bigcup_{\mathcal{C} \in \mathsf{causes}(q,\mathcal{K})} \mathcal{C}$
      ii. $\mathcal{E}_- \leftarrow \mathcal{E}_- \cup F \cup \mathsf{confl}(T,\mathcal{K})$, $\mathcal{A}' \leftarrow \mathcal{A}'\backslash(\mathcal{E}_- \cup T)$
      iii. If a cause for q contains no false assertion: $\mathcal{M} \leftarrow \mathcal{M}\backslash\{M' \in \mathcal{M} \mid q \in M'\}$, go to C.2
   c. MCSW found: $\mathcal{W} \leftarrow \mathcal{W}\backslash M$ and go to B.1
3. No MCSW found: do Step B of $\mathsf{OptRP}_{\mathcal{U}}$, output $\mathcal{E}_-$

Figure 2: Algorithm for optimal deletion-only repair plans

and its queries are removed from $\mathcal{W}$, and we return to the case where a potential solution exists.

Theorem 4. *The algorithm OptDRP always terminates, and it outputs an optimal deletion-only repair plan.*

Proof idea. Termination follows from the fact that every time we return to Step B, something has been added to $\mathcal{E}_-$ or deleted from $\mathcal{W}$, and nothing is ever removed from $\mathcal{E}_-$ or added to $\mathcal{W}$. Since we only add false assertions to $\mathcal{E}_-$, the output plan is validatable. If the algorithm ends at Step B.2.b.ii, then $\mathcal{E}_-$ satisfies every answer characterized in Prop. 3. Indeed, since $\mathcal{E}_-$ is a potential solution, it satisfies every unwanted answer. Moreover, the answers removed from $\mathcal{W}$ at Step C.2.c do not fulfill the conditions of Prop. 3 since all their causes contain some false assertion, and for every remaining $q \in \mathcal{W}$, we ensure that there is a conflict-free cause of q that contains no false assertions. If the algorithm ends at

Step B.2.c.i, the user was asked to indicate false or true assertions at Step B.2.c and was not able to input anything, so the user has deleted all false assertions he knows among the relevant assertions, and thus it is not possible to improve the current repair plan further. A similar argument applies if the algorithm ends at Step C.3. □

To avoid overwhelming the user with relevant assertions at Step B.2.c, it is desirable to reduce the number of assertions presented at a time. This leads us to propose two improvements to the basic algorithm. First, we can divide QRPs into *independent subproblems*. Two answers are considered dependent if their causes (and conflicts in the case of wanted answers) share some assertion. Independent sets of answers do not interact, so they can be handled separately. Second, at Step B.2.c, the assertions can be presented in small batches. Borrowing ideas from work on reducing user effort in interactive revision, we can use a notion of *impact* to determine the order of presentation of assertions. Indeed, deleting or keeping an assertion may force us to delete or keep other assertions to get a potential solution. Relevant assertions can be sorted using two scores that express the impact of being declared false or true. For the impact of an assertion α being false, we use the number of assertions that becomes necessarily (non)false if α is deleted. The impact of α being true also takes into account the fact that the conflicts of α can be marked as false: we consider the number of assertions that are in conflict with α or become necessarily (non)false when we disallow α's removal. We can rank assertions by the minimum of the two scores, using their sum to break ties.

6 Preliminary Experiments

We report on experiments made on core components of the above OptDRP algorithm. We focused on measuring the time to decide whether a potential solution exists (Step B), to compute necessarily (non)false and relevant assertions (Step B.1), to rank the relevant assertions w.r.t. their impact (Step B.2.c), and to find the MCSWs (Step C).

The components were developed in Java using the CQAPri system (www.lri.fr/∼bourgaux/CQAPri) to compute query answers under IAR and brave semantics, with their causes, and the KB's conflicts. We used SAT4J (www.sat4j.org) to solve the (UN)SAT reductions in Section 5.

We borrowed from the CQAPri benchmark [Bienvenu *et al.*, 2016a] available at the URL above its: (i) TBox which is the DL-Lite$_\mathcal{R}$ version of the Lehigh University Benchmark [Lutz *et al.*, 2013] augmented with constraints allowing for conflicts, (ii) c5 and c29 ABoxes with ∼10 million assertions and, respectively, a ratio of assertions involved in conflicts of 5%, that we found realistic, and of 29%, and (iii) queries q_1, q_2, q_3, q_4, shown below. We slightly modified the original queries by changing some constants or variables in order to obtain dependent answers (whose causes and conflicts of causes share some assertions). We built 13 QRPs per ABox, by adding more and more answers of these queries to $\mathcal{U}$ or $\mathcal{W}$; the size of $\mathcal{U} \cup \mathcal{W}$ varies from 8 to 121.

$$q_1 = \exists y\, \mathsf{Person}(x) \wedge \mathsf{takesCourse}(x, y) \wedge \mathsf{Person}(G131) \wedge \mathsf{GraduateCourse}(y) \wedge \mathsf{takesCourse}(G131, y)$$

$$q_2 = \exists x\, \mathsf{Employee}(x) \wedge \mathsf{memberOf}(x, D2) \wedge \mathsf{degreeFrom}(x, y)$$

$$q_3 = \exists y\, \mathsf{teacherOf}(x, y) \wedge \mathsf{degreeFrom}(x, U532)$$

$$q_4 = \exists z\, \mathsf{Employee}(x) \wedge \mathsf{degreeFrom}(x, U532) \wedge \mathsf{memberOf}(x, z) \wedge \mathsf{Employee}(y) \wedge \mathsf{degreeFrom}(y, U532) \wedge \mathsf{memberOf}(y, z)$$

In all of our experiments, deciding if a potential solution exists, as well as computing the set of relevant assertions, takes just a few milliseconds. The difficulty of computing the necessarily (non)false assertions correlates with the number of relevant assertions induced by the QRP. For the c5 QRPs involving 85 to 745 relevant assertions, this computation took between 30ms to 544ms, while it took 24ms to 1333ms for the c29 QRPs involving 143 to 1404 relevant assertions. While these times seem reasonable in practice, ranking the remaining relevant assertions based on their impact is time consuming as it requires a number of calls to the SAT solver quadratic in the number of assertions: it took less than 10s up to ∼150 assertions, less than 5mn up to ∼480 assertions, and up to 25mn for 825 assertions. For all of the QRPs we built, computing the MCSWs takes a few milliseconds; somewhat surprisingly, we always found at most one MCSW.

7 Discussion

The problem of modifying DL KBs to ensure (non)entailments of assertions and/or axioms has been investigated in many works, see e.g. [De Giacomo *et al.*, 2009; Calvanese *et al.*, 2010; Gutierrez *et al.*, 2011].

Our framework is inspired by that of [Jiménez-Ruiz *et al.*, 2011], in which a user specifies two sets of axioms that should be entailed or not by a KB. Repair plans are introduced as pairs of sets of axioms to remove and add to obtain an ontology satisfying these requirements. Deletion-only repair plans are studied in [Jiménez-Ruiz *et al.*, 2009] where heuristics based on the confidence and the size of the plan are used to help the user to choose a plan among all minimal plans.

When axiom (in)validation can be partially automatized, ranking axioms by their potential impact reduces the effort of manual revision [Meilicke *et al.*, 2008; Nikitina *et al.*, 2012]. In our setting, we believe that validating sets of necessarily (non)false assertions requires less effort than hunting for false assertions among all relevant assertions, leading us to propose a similar notion of impact to rank assertions to be examined.

Compared to prior work, distinguishing features of our framework are the specification of changes at the level of CQ answers, the use of inconsistency-tolerant semantics, and the introduction of optimality measures to handle situations in which not all objectives can be achieved.

In future work, two aspects of our approach deserve further attention. First, when insertions are needed, it would be helpful to provide users with suggestions of assertions to add. The framework of query abduction [Calvanese *et al.*, 2013], which was recently extended to inconsistent KBs [Du *et al.*, 2015], could provide a useful starting point. Second, our experiments revealed the difficulty of ranking relevant assertions, so we plan to develop optimized algorithms for computing impact and explore alternative definitions of impact.

Acknowledgements

This work was supported by contract ANR-12-JS02-007-01.

References

[Altwaijry *et al.*, 2013] Hotham Altwaijry, Dmitri V. Kalashnikov, and Sharad Mehrotra. Query-driven approach to entity resolution. *Proceedings of the VLDB Endowment (PVLDB)*, 6(14):1846–1857, 2013.

[Bergman *et al.*, 2015] Moria Bergman, Tova Milo, Slava Novgorodov, and Wang-Chiew Tan. QOCO: A query oriented data cleaning system with oracles. *PVLDB*, 8(12):1900–1911, 2015.

[Bertossi, 2011] Leopoldo E. Bertossi. *Database Repairing and Consistent Query Answering*. Synthesis Lectures on Data Management. Morgan & Claypool Publishers, 2011.

[Bienvenu and Ortiz, 2015] Meghyn Bienvenu and Magdalena Ortiz. Ontology-mediated query answering with data-tractable description logics. In *Lecture Notes of the 11th International Reasoning Web Summer School*, volume 9203 of *LNCS*, pages 218–307. Springer, 2015.

[Bienvenu and Rosati, 2013] Meghyn Bienvenu and Riccardo Rosati. Tractable approximations of consistent query answering for robust ontology-based data access. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*, 2013.

[Bienvenu *et al.*, 2016a] Meghyn Bienvenu, Camille Bourgaux, and François Goasdoué. Explaining inconsistency-tolerant query answering over description logic knowledge bases. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI)*, 2016.

[Bienvenu *et al.*, 2016b] Meghyn Bienvenu, Camille Bourgaux, and François Goasdoué. Query-driven repairing of inconsistent DL-Lite knowledge bases (long version with appendix). Technical Report 1585, LRI, Orsay, France. Available at https://www.lri.fr/~bibli/Rapports-internes/2016/RR1585.pdf, 2016.

[Calvanese *et al.*, 2007] 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 (JAR)*, 39(3):385–429, 2007.

[Calvanese *et al.*, 2010] Diego Calvanese, Evgeny Kharlamov, Werner Nutt, and Dmitriy Zheleznyakov. Evolution of DL-Lite knowledge bases. In *Proceedings of the 9th International Semantic Web Conference (ISWC)*, 2010.

[Calvanese *et al.*, 2013] Diego Calvanese, Magdalena Ortiz, Mantas Simkus, and Giorgio Stefanoni. Reasoning about explanations for negative query answers in DL-Lite. *Journal of Artificial Intelligence Research (JAIR)*, 48:635–669, 2013.

[De Giacomo *et al.*, 2009] Giuseppe De Giacomo, Maurizio Lenzerini, Antonella Poggi, and Riccardo Rosati. On instance-level update and erasure in description logic ontologies. *Journal of Logic and Computation*, 19(5):745–770, 2009.

[Du *et al.*, 2015] Jianfeng Du, Kewen Wang, and Yi-Dong Shen. Towards tractable and practical ABox abduction over inconsistent description logic ontologies. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI)*, 2015.

[Gutierrez *et al.*, 2011] Claudio Gutierrez, Carlos A. Hurtado, and Alejandro A. Vaisman. RDFS update: From theory to practice. In *Proceedings of the 8th European Semantic Web Conference (ESWC)*, 2011.

[Jiménez-Ruiz *et al.*, 2009] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. Ontology integration using mappings: Towards getting the right logical consequences. In *Proceedings of the 6th European Semantic Web Conference (ESWC)*, 2009.

[Jiménez-Ruiz *et al.*, 2011] Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga Llavori. Supporting concurrent ontology development: Framework, algorithms and tool. *Data & Knowledge Engineering Journal (DKE)*, 70(1):146–164, 2011.

[Kontokostas *et al.*, 2014] Dimitris Kontokostas, Patrick Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri. Test-driven evaluation of linked data quality. In *Proceedings of the 23rd International Conference on World Wide Web (WWW)*, 2014.

[Lembo *et al.*, 2010] Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. Inconsistency-tolerant semantics for description logics. In *Proceedings of the 4th International Conference on Web Reasoning and Rule Systems (RR)*, 2010.

[Lembo *et al.*, 2011] Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. Query rewriting for inconsistent DL-Lite ontologies. In *Proceedings of the 5th International Conference on Web Reasoning and Rule Systems (RR)*, 2011.

[Lutz *et al.*, 2013] Carsten Lutz, Inanç Seylan, David Toman, and Frank Wolter. The combined approach to OBDA: Taming role hierarchies using filters. In *Proceedings of the 12th International Semantic Web Conference (ISWC)*, 2013.

[Meilicke *et al.*, 2008] Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin. Supporting manual mapping revision using logical reasoning. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI)*, 2008.

[Motik *et al.*, 2012] Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, and Carsten Lutz. OWL 2 Web Ontology Language profiles. W3C Recommendation, 11 December 2012. Available at http://www.w3.org/TR/owl2-profiles/.

[Nikitina *et al.*, 2012] Nadeschda Nikitina, Sebastian Rudolph, and Birte Glimm. Interactive ontology revision. *Journal of Web Semantics (JWS)*, 12:118–130, 2012.