Query Failure Explanation in Inconsistent Knowledge Bases Using Argumentation

Abdallah ARIOUA \textsuperscript{a} Nouredine TAMANI \textsuperscript{b} Madalina CROITORU \textsuperscript{b} and Patrice BUCHE \textsuperscript{a}

\textsuperscript{a}IATE, INRA, University of Montpellier 2
\textsuperscript{b}LIRMM, INRIA, University of Montpellier 2

Abstract.
We address the problem of explaining Boolean Conjunctive Query (BCQ) failure in the presence of inconsistency within the Ontology-Based Data Access (OBDA) setting, where inconsistency is handled by the intersection of closed repairs semantics (ICR) and the ontology is represented by Datalog+- rules. Our proposal relies on an interactive and argumentative approach where the processes of explanation takes the form of a dialogue between the \textit{User} and the \textit{Reasoner}. We exploit the equivalence between argumentation and ICR-semantics to prove that the \textit{Reasoner} can always provide an answer for user’s questions.

1. Introduction

In the popular \textbf{ONTOLOGY-BASED DATA ACCESS} setting the domain knowledge is represented by an ontology facilitating query answering over existing data [15]. In practical systems involving large amounts of data and multiple data sources, data inconsistency with respect to the ontology is unavoidable. Many inconsistency-tolerant semantics [3,2,12,13] have been proposed that rely on the notion of data repairs i.e. subsets of maximally consistent data with respect to the ontology. Query answering under these semantics may not be intuitively straightforward and can lead to loss of user’s trust, satisfaction and may affect the system’s usability [14]. As argued by Calvanese et al.[6] explanation facilities should not just account for user’s “Why Q ?” question (why a query holds under a given inconsistency-tolerant semantics) but also for question like “Why not Q ?” (why a query does not hold under a given inconsistency-tolerant semantics).

The research problem addressed by this paper is the boolean conjunctive query failure explanation in inconsistent knowledge bases, precisely: “\textit{Given an inconsistent KB and a boolean conjunctive query Q, why Q is not entailed from KB under the ICR-semantics?}”. We use argumentation as an approach for explanation. We consider the logical instantiation of Dung’s [10] abstract argumentation framework for OBDA in [8] and we exploit the equivalence result shown by the authors between the ICR-semantics and sceptical acceptance under preferred semantics to guarantee the existence of an explanation for any failed query. The explanation takes the form of a dialogue between the \textit{User} and the \textit{Reasoner} with the purpose of explaining the query failure. At each level of the dialogue, we use language-based introduced primitives such as \textit{clarification} and
**deepening** to further refine the answer. The added value of our contribution lies in its significance and originality. We are the first to propose query failure explanation in the context of OBDA for **inconsistent knowledge bases** by means of **argumentation**. Our approach differs from [4,6] in handling query failure since we consider an inconsistent setting within OBDA. In addition, the work presented in [11] is neither applied to an OBDA context nor to the Datalog+/- language.

2. Background and Overview

In this section, we introduce the motivation and the context of our work and a formal definition of the addressed problem. Consider a knowledge base about university staff and students which contains inconsistent knowledge. This inconsistency is handled by ICR-semantics. The **User** might be interested in knowing why the knowledge base **does not entail** the query \( Q \) :"Luca is a student”. Observe that the individual \( \delta \) (e.g. Luca in the example above) is a negative answer for a conjunctive query \( Q \) (e.g. get me all the students in the example above) if and only if the boolean conjunctive query \( Q(\delta) \) (e.g. student(Luca) in the example above) has failed. Hence, in this paper we concentrate only explaining the failure of a boolean conjunctive query. Let us formally introduce the problem of **Query Failure Explanation** in inconsistent knowledge bases.

**Definition 1 (Query Failure Explanation Problem \( \mathcal{P} \))** Let \( \mathcal{K} \) be an inconsistent knowledge base, \( Q \) a Boolean Conjunctive Query such that \( \mathcal{K} \not\models_{ICR} Q \). We then call \( \mathcal{P} = (\mathcal{K}, Q) \) a **Query Failure Explanation Problem (QFEP)**.

To address the **Query Failure Explanation Problem**, we use a logical instantiation of Dung’s [10] abstract argumentation framework for OBDA in [8] ensuring that the argumentation framework used respects the rationality postulates [7]. Let us first introduce the OBDA setting and inconsistency-tolerant semantics. We consider the **positive existential** conjunctive fragment of first-order logic, denoted by \( \text{FOL}(\land, \exists) \), which is composed of formulas built with the connectors \( (\land, \rightarrow) \) and the quantifiers \( (\exists, \forall) \). For more details about the language please check [8]. In this paper, for lack of space we simply give an example of a knowledge base \( \mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N}) \) is composed of finite set of facts \( \mathcal{F} \) and finite set of existential rules \( \mathcal{R} \) and a finite set of negative constrains \( \mathcal{N} \).

**Example 1** Let us consider an example inspired from [5]. In an enterprise, employees work in departments and use offices which are located in departments, some employees direct departments, and supervise other employees. In addition, a supervised employee cannot be a manager. A director of a given department cannot be supervised by an employee of the same department, and two employees cannot direct the same department, and an employee cannot work in more than one department. The following sets of (existential) rules \( \mathcal{R} \) and negative constraints \( \mathcal{N} \) model the corresponding ontology:

\[
\mathcal{R} = \begin{cases} 
\forall x \forall y (\text{works\_in}(x,y) \rightarrow \text{emp}(x)) & (r_1) \\
\forall x \forall y (\text{directs}(x,y) \rightarrow \text{emp}(x)) & (r_2) \\
\forall x \forall y (\text{directs}(x,y) \land \text{works\_in}(x,y) \rightarrow \text{manager}(x)) & (r_3) \\
\forall x \forall y \forall z (\text{locate\_office}(y,z) \land \text{uses\_office}(x,y) \rightarrow \text{works\_in}(x,z)) & (r_4)
\end{cases}
\]
Example 2 The knowledge base in Example 1 is inconsistent because the set of facts \{f_1, f_4, f_6\} is inconsistent since it violates the negative constraint \(n_2\). To be able to reason in presence of inconsistency one has to construct first the repairs and intersect their closure. The following is of the repairs:

\[
\mathcal{A}_1 = \{\text{directs}(John, d_1), \text{supervises}(Tom, John), \text{works_in}(Linda, Statistic), \text{uses_office}(Linda, o_1), \text{directs}(Tom, d_1), \text{directs}(Tom, d_2), \text{works_in}(Carlo, Statistic), \text{works_in}(Jane, Statistic), \text{works_in}(Luca, Statistic), \text{emp}(John), \text{emp}(Tom), \text{emp}(Carlo), \text{emp}(Luca), \text{emp}(Jane), \text{emp}(Linda)\}.
\]

The intersection of all closed repairs is:

\[
\bigcap_{\alpha \in \text{Repair} (\mathcal{K})} \text{Cl}_{\mathcal{K}}(\alpha) = \{\text{directs}(Tom, d_1), \text{directs}(Tom, d_2), \text{works_in}(Carlo, Statistic), \text{works_in}(Luca, Statistic), \text{works_in}(Jane, Statistic), \text{emp}(Carlo), \text{emp}(Jane), \text{emp}(Luca), \text{emp}(Tom), \text{emp}(John), \text{emp}(Linda)\}.
\]

Observe that in the intersection of all closed repairs there is \text{works_in}(Luca, Statistic). That means that \text{works_in}(Luca, Statistic) is ICR-entailed from the knowledge base. Whereas, \text{works_in}(Linda, Statistic) is not ICR-entailed since the facts about Linda are conflicting (because she works also for the department of Accounting).
3. Argumentation Framework, Deepening and Clarification

In what follows we quickly recall the definition of argumentation framework in the context of rule-based languages. We use the definition of argument of [8] and extend it to the notions of deepened and clarified arguments. Given a knowledge base $\mathcal{K} = (\mathcal{F}, \mathcal{R}, \mathcal{N})$, the corresponding argumentation framework $\mathcal{A}$ is a pair $(\text{Arg}, \text{Att})$ where Arg is the set of arguments that can be constructed from $\mathcal{F}$ and Att is an asymmetric binary relation called attack defined over $\text{Arg} \times \text{Arg}$. Given an argument $a$ we denote by $\text{Supp}(a)$ the support of the argument and by $\text{Conc}(a)$ the conclusion.

**Example 3 (Argument)** The following argument indicates that John is an employee because he directs department $d_1$:

$$a = \{\text{directs(John, } d_1\text{)}, \text{directs(John, } d_1\text{), emp(John)}\}.$$ 

**Example 4 (Attack)** Consider the argument $a$ of Example 3, the following argument $b = \{\text{supervises(Tom, John), works_in(Tom, } d_1\text{)}, \text{supervises(Tom, John)\wedge works_in(Tom, } d_1\text{)})$ attacks $a$, because $\{\text{supervises(Tom, John)} \wedge \text{works_in(Tom, } d_1\text{)}, \text{directs(John, } d_1\text{)}\}$ is $\mathcal{R}$-inconsistent since it violates the constraint $n_2$.

The results of [8] show the equivalence between sceptically acceptance under preferred semantics and ICR-entailment. Let us now propose functionalities that give the User the possibility to manipulate arguments to gain clarity for query answering and namely: deepening and clarification. Deepening aims at showing the reason why an argument attacks another. In our knowledge base the attack is justified by the violation of a constraint. Put differently, an argument attacks another argument if the conclusion of the former and the hypothesis of the latter are mutually exclusive. Thus deepening amounts to explain the attack between two arguments by showing the violated constraint.

**Definition 2 (Deepening $\mathcal{D}$)** Given two arguments $a, b \in \mathcal{A}$. The mapping deepening denoted by $\mathcal{D}$ is a total function from the set $\mathcal{F}$ to $2^\mathcal{F}$ defined as follows:

1. $n \in \mathcal{N}$ and,
2. $\exists f \in \text{Supp}(a)$ such that $\text{Cl}_{\mathcal{A}}((\text{Conc}(b), f)) \models H_n$.

Note that $H_n$ is the hypothesis of the constraint $n$.

**Example 5 (Deepening)** Consider the argument $a$ of Example 3, the argument $b = \{\text{supervises(Tom, John), works_in(Tom, } d_1\text{)}, \text{supervises(Tom, John)\wedge works_in(Tom, } d_1\text{)})$ attacks $a$, hence deepening is $\mathcal{D}(b, a) = \{\forall x \forall y \forall z (\text{supervises(x,y)\wedge work_in(x,z)\wedge directs(y,z)} \rightarrow \bot\}$. This explains why the argument $b$ attacks $a$.

The information carried by the argument would be more useful if the structure exhibits the line of reasoning leading to the conclusion, called clarifying the argument.

**Definition 3 (Clarifying $\mathcal{C}$)** Given an argument $a \in \mathcal{A}$. The mapping clarification denoted by $\mathcal{C}$ is a total function from the set $\mathcal{F}$ to $2^\mathcal{F}$ such that: $\mathcal{C}(a = (F_0, ..., F_n)) = \{r | r \in \mathcal{R}$ s.t $r$ is applicable to $F_i$ and the application of $r$ on $F_i$ yields $F_{i+1}$ for all $i \in \{0, ..., n\}\}$.

**Definition 4 (Clarified Argument)** Given an argument $a \in \mathcal{A}$. The corresponding clarified argument $\mathcal{C}_a$ is a 3-tuple $\langle \text{Supp}(a), \mathcal{C}(a), \text{Conc}(a) \rangle$ such that $\mathcal{C}(a) \subseteq \mathcal{R}$ are the rules used to derive the conclusion $\text{Conc}(a)$. 

Example 6 (Clarification count. Example 3) A clarified version of the argument $a$ is $C_a = \{\{\text{directs}(\text{John}, d_1)\}, \{\forall x \forall d \, \text{directs}(x, d) \rightarrow \text{emp}(x)\}, \text{emp}(\text{John})\}$ such that $\text{Supp}(C_a) = \{\text{directs}(\text{John}, d_1)\}$, $\text{Conc}(C_a) = \{\forall x \forall d \, \text{directs}(x, d) \rightarrow \text{emp}(x)\}$ and $\text{Conc}(C_a) = \text{emp}(\text{John})$.

4. Dialectical Explanation for Query Failure

In what follows, we describe a simple dialectical system of explanation based on the work of [9]. Our system is custom-tailored for the problem of Query Failure Explanation under ICR-semantics in inconsistent knowledge bases with rule-based language. Our dialectical explanation involves two parties: the User and the Reasoner. The User wants to understand why the query is not ICR-entailed and the Reasoner provides a respond aiming at showing the reason why the query is not ICR entailed. We model this explanation through a dialogue composed of moves (speech acts) put forward by both the User and the Reasoner. This dialogue is governed by rules (pre/post conditions rules, termination rules, success rules) that specify what type of moves should follow the other, the conditions under which the dialogue terminates, and when and under which conditions the explanation has been successfully achieved (success rules).

We denote by $\text{Arg}^+ (Q)$ the set of all arguments that support the query $Q$, namely $a \in \text{Arg}^+ (Q)$ iff $\text{Conc}(a) \models Q$. In what follows we define types of moves that can be used in the dialogue.

Definition 5 (Moves) A move is a 3-tuple $m = \langle ID, I, \omega \rangle$ such that:

1. $m$ is an explanation request, denoted by $m^{\text{Req}}$ iff $ID = \text{User}$, $I$ is a query $Q$ and $\omega$ is an argument supporting $Q$.
2. $m$ is an explanation response, denoted by $m^{\text{Resp}}$ iff $ID = \text{Reasoner}$, $I$ is an argument supporting $Q$ and $\omega$ is an argument such that $\omega$ attacks $I$.
3. $m$ is a follow-up question, denoted by $m^{\text{FQ}}$ iff $ID = \text{User}$, $I$ is an argument and $\omega$ is either $\text{Conc}(I)$ or an argument $\omega_1$ that supports $Q$ s.t. $(\omega, \omega_1) \in \text{ktc}$.
4. $m$ is a follow-up answer, denoted by $m^{\text{FA}}$ iff $ID = \text{Reasoner}$, $I$ is an argument and $\omega$ is either a deepening $\mathcal{D}$ or a clarified argument $\mathcal{C}(I)$.

The explanation request $m^{\text{Req}} = \langle \text{User}, Q, \omega \rangle$ is an explanation request made by the User asking "why the query $Q$ is not ICR-entailed while there is an argument $\omega$ asserts the entailment of $Q$". An explanation response $m^{\text{Resp}} = \langle \text{Reasoner}, \omega, \omega_1 \rangle$ made by the Reasoner is an explanation for the previous inquiry by showing that the argument $\omega$ (that supports $Q$) is the subject of an attack made by $\omega_1$. The User also can ask a follow-up question if the Reasoner provides an explanation. The follow-up question $m^{\text{FQ}} = \langle \text{User}, \omega_1, \omega \rangle$ is a compound move, it can represent a need for deepening (the User wants to know why the argument $\omega_1$ is attacking the argument $\omega$) or the need for clarification (how the argument $\omega_1$ comes to a certain conclusion). To distinguish them, the former has $\omega = \text{Conc}(\omega_1)$ and the latter has $\omega$ as an argument. A follow-up answer $m^{\text{FA}} = \langle \text{Reasoner}, \omega_1, \omega_\prime \rangle$ is also a compound move. Actually, it depends on the follow-up question. It shows the argument $\omega_1$ that needs to be deepened (resp. clarified) and its deepening (resp. clarification) by the deepening mapping $\mathcal{D}(\omega_1, \omega)$ (resp. clarification mapping $\mathcal{C}(\omega)$) in Definition 4 (resp. Definition 6). An example is provided afterward.
In what follows we specify the structure of dialectical explanation and the rules that have to be respected throughout the dialogue.

**Definition 6 (Dialectical Explanation)** Given a QFEP $\mathcal{P}$. A dialectical explanation $D_{\text{exp}}$ for $\mathcal{P}$ is a non-empty sequence of moves $\langle m_1, m_2, \ldots, m_n \rangle$ where $s \in \{\text{ERQ}, \text{FQ}, \text{ERP}, \text{FA}\}$ and $i \in \{1, \ldots, n\}$ such that:

1. The first move is always an explanation request $m_1^{\text{ERQ}}$, we call it an opening.
2. $s \in \{\text{ERQ}, \text{FQ}\}$ iff $i$ is odd, $s \in \{\text{ERP}, \text{FA}\}$ iff $i$ is even.
3. For every explanation request $m_i^{\text{ERQ}} = \langle \text{User}, I_i, \omega_i \rangle$, $I_i$ is the query $Q$ to be explained and $\omega_i$ is an argument supporting $Q$ and for all $m_j^{\text{ERQ}}$ s.t $j < i$, $\omega_i \neq \omega_j$.
4. For every explanation response $m_i^{\text{ERP}} = \langle \text{Reasoner}, I_i, \omega_i \rangle$ s.t $i \geq 1$, $I_i = \omega_{i-1}$ and $\omega_i = \omega_i' s.t (\omega_i', I_i) \in \text{Att}$.
5. For every follow-up question $m_i^{\text{FQ}} = \langle \text{User}, I_i, \omega_i \rangle$, $i \geq 1$, $I_i = I_{i-1}$ and $\omega_i$ is either $I_{i-1}$ or $\text{Conc}(\omega_{i-1})$.
6. For every follow-up answer $m_i^{\text{FA}} = \langle \text{Reasoner}, I_i, \omega_i \rangle$, $i \geq 1$, $I_i = I_{i-1}$ and $\omega_i = D(I_i, \omega_{i-1})$ or $\omega = C(I_i)$.

We denote by $\text{Arg}_{\text{user}}(D_{\text{exp}})$ the set of all arguments put by the User in the dialogue.

Every dialogue has to respect certain rules (protocol). These rules organize the way the Reasoner and the User should put the moves. For each move we specify the conditions that have to be met for the move to be valid (preconditions). We also specify the conditions that identify the next moves (postconditions).

**Definition 7 (Pre/Post Condition Rules)** Given a QFEP $\mathcal{P}$ and a dialectical explanation $D_{\text{exp}}$ for $\mathcal{P}$. Then, $D_{\text{exp}}$ has to respect the following rules:

**Explanation request:**
- **Preconditions:** The beginning of the dialogue or the last move of the Reasoner was either an explanation response or a follow-up answer.
- **Postconditions:** The next move must be an explanation answer.

**Explanation response:**
- **Preconditions:** The last move by the User was an explanation request.
- **Postconditions:** The next move must be either another explanation request (it may implicitly mean that the User had not understood the previous explanation) or a follow-up question.

**Follow-up question:**
- **Preconditions:** The last move by the Reasoner was an explanation response or this follow-up question is not the second in a row.
- **Postconditions:** The next move must be a follow-up answer.

**Follow-up answer:**
- **Preconditions:** The last move by the User was a follow-up question.
- **Postconditions:** The next move must be an explanation request (it may implicitly mean that the User had not understood the previous explanation).
Beside the previous rules, there are termination rules that indicate the end of a dialectical explanation.

**Definition 8 (Termination Rules)** Given a QFEP $P$ and a dialectical explanation $D_{exp}$ for $P$. Then, $D_{exp}$ terminates when the User puts an empty explanation request $m_{i}^{ERQ} = \langle \text{User}, 0, 0 \rangle$ or when $\text{Arg}_{U}(D_{exp}) = \text{Arg}^{\sim}(Q)$.

The rules in Definition 7 & 8 state that the Reasoner is always committed to respond with an explanation response, the User then may indicate the end of the dialogue by an empty explanation request (Definition 8) declaring his/her understanding, otherwise starts another explanation request (this indicates that he/she has not understood the last explanation) or asks a follow-up question, the User cannot ask more than two successive follow-up questions. If the User asks a follow-up question then the Reasoner is committed to a follow-up answer. When the User asks for another explanation he/she cannot use an argument that has already been used. If the User ran out of arguments and he/she has not yet understood, the dialogue ends (Definition 8) and the explanation is judged unsuccessful. It is important to notice that when the Reasoner wants to answer the User there may be more than one argument to choose. There are many “selection strategies” that can be used in such case (for instance, the shortest argument, the least attacked argument...etc), but their study is beyond the scope of the paper.

In what follows we elaborate more on the success and the failure of an explanation.

**Definition 9 (Success Rules)** Given a QFEP $P$ and a dialectical explanation $D_{exp}$ for $P$. Then, $D_{exp}$ is successful when it terminates with an empty explanation request $m_{i}^{ERQ} = \langle \text{User}, 0, 0 \rangle$, otherwise it is unsuccessful.

A dialectical explanation is judged to be successful if the User terminates the dialogue voluntarily by putting an empty explanation request. If the User has used all arguments supporting $Q$ then he/she is forced to stop without indicating his/her understanding, in this case we consider the explanation unsuccessful. By virtue of the equivalence between ICR-semantics and argumentation presented in Section 3, the existence of response is always guaranteed. This property is depicted in the following proposition.

**Proposition 1 (Existence of response)** Given a QFEP $P$ and a dialectical explanation $D_{exp}$ for $P$. Then, For every $m_{i} \in D_{exp}$ s.t $s \in \{ERQ, FQ\}$ and $1 \leq i \leq |D_{exp}|$, the next move $m_{i+1}$ s.t $s \in \{ERP, FA\}$ always exists.

5. **Conclusion**

In this paper, we have presented a dialectical approach for explaining boolean conjunctive queries failure, designated by Query Failure Explanation Problem (QFEP), in an inconsistent ontological knowledge base where inconsistency is handled by inconsistency-tolerant semantics (ICR) and issued from the set of facts. The introduced approach relies on both (i) the relation between ontological knowledge base and logical argumentation framework and (ii) the notions of argument deepening and clarifications. So, through a dialogue, the proposed approach explains to the User how and why his/her query is not entailed under ICR semantics.
We currently investigate the explanation problem not only for Query Failure but also for Query Answering. We have proposed a Query Explanation framework under the CoGui editor[1] and plan to test the two approaches within the DUR-DUR ANR project which investigates the use of argumentation in agri-food chains.

Acknowledgement Financial support from the French National Research Agency (ANR) for the project DUR-DUR and ASPIQ (ANR-12-BS02-0003) is gratefully acknowledged.

References


