Efficient Query Answering Over Expressive Inconsistent Description Logics*

Eleni Tsalapati[1] and **Giorgos Stoilos**[2] and **Giorgos Stamou**[1] and **George Koletsos**[1]

[1]School of Electrical and Computer Engineering,
National Technical University of Athens, Greece

[2]Department of Informatics,
Athens University of Economics and Business

Abstract

Inconsistent-tolerant semantics, like the IAR and ICAR semantics, have been proposed as means to compute meaningful query answers over inconsistent Description Logic (DL) ontologies. In the current paper we present a framework for scalable query answering under both the IAR and ICAR semantics, which is based on highly efficient data saturation systems. Our approach is sound and complete for ontologies expressed in the lightweight DL DL-Lite, but for more expressive DLs the problem is known to be intractable, hence our algorithm only computes upper approximations. Nevertheless, its structure motivates a new type of ICAR-like semantics which can be computed in polynomial time for a very large family of DLs. We have implemented our techniques and conducted an experimental evaluation obtaining encouraging results as *both* our IAR- and ICAR-answering approaches are far more efficient than existing available IAR-based answering systems.

1 Introduction

Conjunctive query answering over data described using ontological knowledge is a key reasoning service for many modern applications [Motik *et al.*, 2012; Chaussecourte *et al.*, 2013]. Although query answering is normally defined over consistent datasets, in real-world applications the data can very often be inconsistent with respect to the axioms specified in the ontology. In this case the straightforward approach would be to try and resolve the inconsistencies by "cleaning" the dataset from the conflicting elements. However, in many occasions this might not be practical due to, e.g., the large volume of the data and the non-deterministic nature of the cleaning procedures. A second approach that has been proposed in the literature, called *consistent query answering*, is to devise semantics which describe which answers are "meaningful" to be returned even in the presence of the inconsistencies [Arenas *et al.*, 1999; Bertossi, 2006]. For this reason such semantics are referred to as *inconsistent-tolerant*.

Consistent query answering has recently been studied in the field of Description Logics (DLs) [Lembo *et al.*, 2010; Rosati, 2011; Bienvenu and Rosati, 2013; Bienvenu *et al.*, 2014]. Two important approaches, mostly due to their nice computational properties over the lightweight DL DL-Lite, consist of the so-called *IAR* and *ICAR* semantics [Lembo *et al.*, 2010]. Despite important theoretical results and a few implemented systems [Masotti *et al.*, 2011; Rosati *et al.*, 2012; Bienvenu *et al.*, 2014], designing practically efficient consistent query answering systems that could scale up to billions of data is still largely open. Especially for the ICAR semantics, only a preliminary effort was reported by Masotti *et al.* [2011] but the evaluation used very small proprietary datasets (of the scale of a few thousands). Furthermore, over many *still* lightweight DLs, like $\mathcal{EL}$, ICAR-based query answering is intractable [Rosati, 2011] and, to the best of our knowledge, no approximate algorithms exist in the literature; Bienvenu and Rosati [2013] provided approximations of a different type of inconsistent-tolerant semantics called AR semantics.

In this work we present a framework for efficient query answering under both the IAR and ICAR semantics. First, we focus on the ICAR semantics and provide a query answering framework that is based on highly efficient mature data saturation (triple-store) systems. This is particularly interesting as these systems have shown to be able to handle billions of data. Moreover, their properties enable us to propose additional refinements and optimisations. Our algorithm is correct (sound and complete) for ontologies expressed in languages of the DL-Lite family. However, as mentioned, ICAR-based query answering over DLs like $\mathcal{EL}$ is intractable [Rosati, 2011], hence our algorithm only computes upper approximations of them. Nevertheless, its structure motivates a new type of ICAR-like semantics which we show that can be computed in polynomial time for a very large number of highly expressive DLs; none such semantics were previously known. Subsequently, we show that our framework can also be used for IAR-based query answering for ontologies expressed in the DLs DL-Lite and $\mathcal{EL}_{\bot nr}$ [Rosati, 2011]. Finally, we have conducted an experimental evaluation obtaining encouraging results as both our approaches (IAR and ICAR) are more efficient than existing IAR-answering systems [Rosati *et al.*, 2012; Bienvenu *et al.*, 2014].

*At the time of acceptance Giorgos Stoilos was a member of NTUA supported by a Marie Curie Career Reintegration Grant within European Union's 7th Framework Programme (FP7/2007-2013) under REA grant agreement 303914.

2 Preliminaries

We use standard notions of first-order constants, variables, atoms, satisfiability, entailment ($\models$), model, substitutions (σ), etc. We also assume familiarity with datalog. For a conjunction of atoms $\mathcal{B} = \alpha_1 \wedge \ldots \wedge \alpha_n$ we often abuse notation and write $\mathcal{B} \subseteq S$ to denote that $\{\alpha_1, \ldots, \alpha_n\} \subseteq S$.

2.1 Description Logics

Description Logics (DLs) [Baader *et al.*, 2003] constitute of a family of (mostly) decidable fragments of FOL. We use $\mathcal{L}$ to denote an arbitrary DL language and next we recapitulate the syntax of the DLs $\mathcal{EL}_\bot$ [Baader *et al.*, 2005], DL-Lite [Calvanese *et al.*, 2007], and $\mathcal{RL}$ [Motik *et al.*, 2009].

The set of $\mathcal{EL}_\bot$-concepts is inductively defined by the grammar: $C := \bot \mid A \mid C_1 \sqcap C_2 \mid \exists R.C$, where A (R) is called atomic concept (role) and $C_{(i)}$ are $\mathcal{EL}_\bot$-concepts. An $\mathcal{EL}_\bot$-TBox $\mathcal{T}$ is a finite set of inclusions of the form $C_1 \sqsubseteq C_2$ with C_1, C_2 $\mathcal{EL}_\bot$-concepts. Inclusions of the form $C_1 \sqcap C_2 \sqsubseteq \bot$ (also written as $C_1 \sqsubseteq \neg C_2$) are called *negative* and the rest *positive*. We will also refer to $\mathcal{EL}_{\bot nr}$, a syntactic restriction of $\mathcal{EL}_\bot$ defined by Rosati [2011].

DL-Lite restricts $\mathcal{EL}_\bot$ by allowing only for concepts of the form A or $\exists R.\top$; however, R in DL-Lite can also be the *inverse* of a role of the form S^- and we can also have inclusions of the form $S \sqsubseteq R$ or $S \sqsubseteq \neg R$ for S, R roles. Finally, roughly speaking, $\mathcal{RL}$ is defined as those DL axioms that can be translated (by simple syntactic transformations) into datalog. For example, $\exists R.A \sqsubseteq B$ is an $\mathcal{RL}$ axiom since it corresponds to the datalog rule $R(x, y) \wedge A(y) \rightarrow B(x)$, while $A \sqsubseteq \exists R.\top$ is not, as it corresponds to $A(x) \rightarrow R(x, f(x))$.

An ABox $\mathcal{A}$ is a finite set of *assertions* of the form $A(a)$ or $R(a, b)$ where a, b are constants called *individuals*. $\mathcal{A}$ is *consistent* w.r.t. some TBox $\mathcal{T}$ if there exists a model for $\mathcal{T} \cup \mathcal{A}$; otherwise it is *inconsistent*. Finally, we will use $\mathcal{T}^+$ to denote all the positive inclusions of a TBox $\mathcal{T}$.

2.2 Queries and Query Answering

A *conjunctive query* (CQ) is an expression of the form $\exists \vec{y}.\phi(\vec{x}, \vec{y})$, where ϕ is a conjunction of function-free atoms containing only variables from $\vec{x}$ (called *answer* variables) or from $\vec{y}$ (called *existential* variables). A *union of conjunctive queries* (UCQ) is a set of CQs all having the same number of answer variables. A tuple of constants $\vec{a}$ is a *certain answer* of a query $\mathcal{Q}$ over $\mathcal{T} \cup \mathcal{A}$ if $\mathcal{T} \cup \mathcal{A} \models \exists \vec{y}.\phi(\vec{a}, \vec{y})$. We denote with $\mathsf{cert}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A})$ all the certain answers of $\mathcal{Q}$ over $\mathcal{T} \cup \mathcal{A}$.

Query answering can be realised via a technique known as query rewriting [Calvanese *et al.*, 2007] which we recall next.

Definition 1. Let $\mathcal{Q}$ be a CQ and let $\mathcal{T}$ be a TBox. A *datalog rewriting* (or simply *rewriting*) $\mathcal{R}$ for $\mathcal{Q}$, $\mathcal{T}$ is a pair $\langle \mathcal{R}_Q, \mathcal{R}_D \rangle$, where $\mathcal{R}_Q$ is a UCQ, $\mathcal{R}_D$ is a datalog program, and for each ABox $\mathcal{A}$ *consistent* w.r.t. $\mathcal{T}$ we have:

$$\mathsf{cert}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}) = \mathsf{cert}(\mathcal{R}_Q, \mathcal{R}_D \cup \mathcal{A})$$

If $\mathcal{R}_D = \emptyset$ then $\mathcal{R}$ is called *UCQ rewriting* and instead of $\langle \mathcal{R}_Q, \emptyset \rangle$ we simply use $\mathcal{R}_Q$ to identify the rewriting.

A DL $\mathcal{L}$ is called *datalog rewritable* if for every $\mathcal{L}$-TBox $\mathcal{T}$ a datalog program $\mathcal{R}_D$ exists such that for every query $\mathcal{Q}$ without existential variables $\mathcal{R} = \langle \{\mathcal{Q}\}, \mathcal{R}_D \rangle$ is a datalog rewriting for $\mathcal{Q}, \mathcal{T}$.

All DLs mentioned above are datalog rewritable.

2.3 $\mathcal{RL}$ query answering systems

Since $\mathcal{RL}$-axioms correspond to datalog rules, to perform query answering systems that support this DL follow a datalog-like saturation approach based on the $\mathcal{RL}$ fragment of some TBox $\mathcal{T}$, which in the following we denote by $\mathcal{T}|_{\mathsf{rl}}$.

Definition 2. An $\mathcal{RL}$ *ABox-saturation* system ans is a procedure that takes as input a TBox $\mathcal{T}$, an ABox $\mathcal{A}$, and a CQ $\mathcal{Q}$ and returns a set of tuples $\mathsf{ans}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}) = \mathsf{cert}(\mathcal{Q}, \mathcal{A}_s)$, where $\mathcal{A}_s \supseteq \mathcal{A}$, called *saturation*, contains all assertions α such that $\mathcal{T}|_{\mathsf{rl}} \cup \mathcal{A} \models \alpha$.

Most systems known to us, like GraphDB, Oracle's Semantic Graph, and RDFox are $\mathcal{RL}$ ABox-saturation systems.

2.4 Inconsistency-tolerant Query Answering

In order to return meaningful answers even from inconsistent datasets, consistent query answering has been introduced in the areas of databases and ontologies. Next, we recapitulate the so-called IAR and ICAR semantics [Lembo *et al.*, 2011].

Definition 3. For a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$ let $clc(\mathcal{T}, \mathcal{A}) = \{\alpha \mid \text{some consistent subset } S \subseteq \mathcal{A} \text{ exists s.t. } \mathcal{T} \cup S \models \alpha\}$. The *Intersection Closed ABox Repair* (ICAR) of $\mathcal{T} \cup \mathcal{A}$ is defined to be the intersection of all subsets $\mathcal{A}' \subseteq clc(\mathcal{T}, \mathcal{A})$ that are consistent w.r.t. $\mathcal{T}$ and no other $\mathcal{A}''$ consistent w.r.t. $\mathcal{T}$ exists s.t. $\mathcal{A}' \subset \mathcal{A}'' \subseteq clc(\mathcal{T}, \mathcal{A})$.

Let $\mathcal{Q}$ be a CQ and let $\mathcal{A}^{\mathsf{ic}}$ be the ICAR of $\mathcal{T} \cup \mathcal{A}$. A tuple $\vec{a}$ is an *ICAR-answer of $\mathcal{Q}$ w.r.t. $\mathcal{T} \cup \mathcal{A}$* if $\vec{a} \in \mathsf{cert}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}^{\mathsf{ic}})$. We denote by $\mathsf{cert}^{\mathsf{ic}}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A})$ all ICAR-answers of $\mathcal{Q}$ w.r.t. $\mathcal{T} \cup \mathcal{A}$. The IAR semantics and answers (denoted by $\mathsf{cert}^{\mathsf{ia}}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A})$) are defined similarly by replacing $clc(\mathcal{T}, \mathcal{A})$ with $\mathcal{A}$.

An algorithm to compute IAR-/ICAR-answers over DL-Lite-TBoxes was presented by Lembo *et al.* [2011]: Let $\mathcal{R}$ be a UCQ rewriting for some query $\mathcal{Q}$ and TBox $\mathcal{T}$. Evaluating $\mathcal{R}$ over $\mathcal{A}$ would return spurious IAR-/ICAR-answers; hence each $\mathcal{Q} \in \mathcal{R}$ needs to be extended with proper *negative* atoms which will guarantee that only answers that are supported by the intersection of all repairs are returned. Roughly speaking, for each atom $A(x)$ that appears in $\mathcal{Q}$ if $\mathcal{T} \models A \sqsubseteq \neg B$ then $\neg B(x)$ is added to $\mathcal{Q}$. We denote this procedure by $\mathsf{ref}(\mathcal{R}, \mathcal{T})$ and illustrate it with an example; for details please see [Lembo *et al.*, 2011, Section 7.2.2].

Example 4. Consider the following TBox and ABox:

$$\begin{array}{rcl} \mathcal{T} & = & \{\exists P \sqsubseteq B,\ P \sqsubseteq \neg P', B \sqsubseteq \neg B'\}, \\ \mathcal{A} & = & \{P(a, b), P'(a, b), P(c, d), B(e), B'(e)\} \end{array}$$

as well as query $\mathcal{Q} = B(x)$. Clearly, $\mathcal{A}$ is inconsistent w.r.t. $\mathcal{T}$ (e.g., due to $\{B(e), B'(e), B \sqsubseteq \neg B'\}$), however, we have $\mathsf{cert}^{\mathsf{ia}}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}) = \{c\}$ and $\mathsf{cert}^{\mathsf{ic}}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}) = \{a, c\}$ ("a" is an ICAR-answer since $B(a) \in clc(\mathcal{T}, \mathcal{A})$). Now, $\mathcal{R}_Q = \{B(x), \exists y.P(x, y)\}$ is a UCQ rewriting for $\mathcal{Q}, \mathcal{T}$. Then, $\mathsf{ref}(\mathcal{R}_Q, \mathcal{T})$ returns the following set:

$$\{B(x) \wedge \neg B'(x), \exists y.(P(x, y) \wedge \neg P'(x, y)) \wedge \neg B'(x)\}$$

It is easy to see that if we evaluate $\mathsf{ref}(\mathcal{R}_Q, \mathcal{T})$ on $\mathcal{A}$ we obtain $\{c\}$, i.e., the IAR-answers; notice how the negative atom

$\neg B'(x)$ prevents "e" from being returned as an answer. Finally, to obtain the ICAR-answers, a rewriting algorithm is applied a *second* time over $\mathsf{ref}(\mathcal{R}_Q, \mathcal{T})$ returning the set:

$$\mathsf{ref}(\mathcal{R}_Q, \mathcal{T}) \cup \{\exists y.P(x,y) \wedge \neg B'(x)\}$$

Evaluating the above over $\mathcal{A}$ will indeed return $\{a, c\}$. ♢

3 Efficient ICAR-Answering Over DL-Lite

Interestingly, the saturation computed by ABox-saturation systems given $\mathcal{T}$ and $\mathcal{A}$ most likely consists of an approximation of $clc(\mathcal{T}, \mathcal{A})$. Hence, these systems could (potentially) be used for efficiently computing ICAR-answers.

Example 5. Consider $\mathcal{T}, \mathcal{A}$, and $\mathcal{Q}$ from Example 4 as well as some $\mathcal{RL}$ ABox-saturation system ans. Since $\mathcal{T}^+$ is an $\mathcal{RL}$-TBox, over $\mathcal{T}^+ \cup \mathcal{A}$ ans will compute the saturation $\mathcal{A}_s = \mathcal{A} \cup \{B(a), B(c)\}$. Moreover, $\mathsf{ref}(\{\mathcal{Q}\}, \mathcal{T}) = \{B(x) \wedge \neg B'(x)\}$ which evaluated over $\mathcal{A}_s$ returns the set $\{a, c\}$, i.e., precisely the ICAR-answers. Note that since we want to compute answers in the presence of inconsistencies rather than allow ans to report them, we discarded the negative inclusions and considered how ans behaves over $\mathcal{T}^+ \cup \mathcal{A}$.[1] ♢

However, in TBoxes that contain non-$\mathcal{RL}$-axioms such systems will miss certain (ICAR-)answers. To overcome this issue we consider the TBox completion framework introduced by Stoilos et al. [2011; 2014]. Intuitively, in many cases "materialising" certain entailed axioms of the TBox "helps" the $\mathcal{RL}$ system compute all certain answers even over non-$\mathcal{RL}$-TBoxes. We recall this notion next.

Definition 6 ([Stoilos *et al.*, 2011; Stoilos, 2014])**.** Let $\mathcal{T}$ be a TBox and let ans be an $\mathcal{RL}$ ABox-saturation system. A *completion* of $\mathcal{T}$ for ans is a set of axioms $\mathcal{C}$ such that $\mathcal{T} \models \mathcal{C}$ and for every ABox $\mathcal{A}$ consistent w.r.t. $\mathcal{T}$ and for every CQ $\mathcal{Q}$ without existential variables we have: $\mathsf{cert}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}) \subseteq \mathsf{ans}(\mathcal{Q}, \mathcal{T} \cup \mathcal{C} \cup \mathcal{A})$.

Moreover, let $\mathcal{Q}$ be some arbitrary CQ. If a rewriting $\langle \mathcal{R}_Q, \mathcal{R}_D \rangle$ for $\mathcal{Q}, \mathcal{T}$ exists then we have: $\mathsf{cert}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}) \subseteq \mathsf{ans}(\mathcal{R}_Q, \mathcal{T} \cup \mathcal{C} \cup \mathcal{A})$.

Example 7. Consider again $\mathcal{T}, \mathcal{A}$, and $\mathcal{Q}$ from Example 4 extended to $\mathcal{T}'$ and $\mathcal{A}'$ as follows:

$$\mathcal{T}' = \{C \sqsubseteq \exists P\} \cup \mathcal{T} \qquad \mathcal{A}' = \{C(f)\} \cup \mathcal{A}$$

Clearly, $\mathsf{cert}^{\mathsf{ic}}(\mathcal{Q}, \mathcal{T}' \cup \mathcal{A}') = \mathsf{cert}^{\mathsf{ic}}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}) \cup \{f\}$. Consider also an $\mathcal{RL}$ ABox-saturation system ans. Since $\mathcal{T}|_{\mathsf{rl}} = \{\exists P \sqsubseteq B\}$ there are clearly ABoxes for which ans will miss answers of $\mathcal{Q}$. For example, we have $f \in \mathsf{cert}(\mathcal{Q}, \mathcal{T}' \cup \{C(f)\})$ but $f \notin \mathsf{ans}(\mathcal{Q}, \mathcal{T}' \cup \{C(f)\})$ since $\mathcal{T}|_{\mathsf{rl}} \cup \{C(f)\} \not\models B(f)$. Hence, the ICAR-answer "f" cannot be computed by the procedure we illustrated in Example 5.

However, consider the completion $\mathcal{C} = \{C \sqsubseteq B\}$ of $\mathcal{T}$ for ans. Then, over $\mathcal{T}|_{\mathsf{rl}} \cup \mathcal{C} \cup \mathcal{A}$ system ans will compute the saturation $\mathcal{A}_s = \mathcal{A} \cup \{B(f)\}$ since $\mathcal{T}|_{\mathsf{rl}} \cup \mathcal{C} \cup \mathcal{A} \models B(f)$. Finally, evaluating $B(x) \wedge \neg B'(x)$ over $\mathcal{A}_s$ returns the desired ICAR-answers, i.e., the set $\{a, c, f\}$. ♢

[1] For this reason, in the following, when we write $\mathcal{T}|_{\mathsf{rl}}$ we mean $\mathcal{T}^+|_{\mathsf{rl}}$.

Consequently, completion and $\mathcal{RL}$ ABox-saturation systems can be used to efficiently compute the ICAR-answers over DL-Lite TBoxes. A minor technical issue is that we first need to remove all inconsistent singleton assertions.

Definition 8. Let $\mathcal{T}$ be a $\mathcal{L}$-TBox and $\mathcal{A}$ and ABox, then we denote by $cr(\mathcal{A}, \mathcal{T})$ the subset of $\mathcal{A}$ that is obtained by removing from $\mathcal{A}$ all assertions $\alpha \in \mathcal{A}$ such that the set $\mathcal{T} \cup \{\alpha\}$ is inconsistent.

Our previous claims are formalised next.

Theorem 9. *Let $\mathcal{Q}$ be a CQ, let $\mathcal{T}$ be a DL-Lite-TBox, let $\mathcal{A}$ be an ABox and let $\langle \mathcal{R}_Q, \mathcal{R}_D \rangle$ be a rewriting for $\mathcal{Q}, \mathcal{T}$. Let also* ans *be an $\mathcal{RL}$ ABox-saturation system and let $\mathcal{TC} = \mathcal{T}^+ \cup \mathcal{C}$ for $\mathcal{C}$ a completion of $\mathcal{T}$ for* ans*. Then,*

$$\mathsf{cert}^{\mathsf{ic}}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}) = \mathsf{ans}(\mathsf{ref}(\mathcal{R}_Q, \mathcal{T}), \mathcal{TC} \cup cr(\mathcal{A}, \mathcal{T}))$$

Note that for DL-Lite TBoxes, completions for $\mathcal{RL}$ systems and rewritings for any conjunctive query always exist. Hence, our result is of high practical relevance.

3.1 Additional Optimisations

Interestingly, as the following example shows, the use of ABox-saturation systems allows us to further simplify the structure of the query returned by the procedure ref.

Example 10. Consider the following TBox and ABox:

$$\begin{aligned} \mathcal{T} &= \{B \sqsubseteq A, B' \sqsubseteq A', A \sqsubseteq \neg A'\} \\ \mathcal{A} &= \{B(a), A'(a), B(b), B'(b)\} \end{aligned}$$

and consider also the query $\mathcal{Q} = B(x)$. Then, we have $\mathsf{ref}(\mathcal{Q}, \mathcal{T}) = \{\mathcal{Q}'\}$, where $\mathcal{Q}' = B(x) \wedge \neg A'(x) \wedge \neg B'(x)$, which evaluated over $\mathcal{A}$ returns $\emptyset$. Note that both of the negative atoms are needed in the query; $\neg A'(x)$ to exclude individual "a" and $\neg B'(x)$ to exclude "b".

Consider now an $\mathcal{RL}$ ABox-saturation system ans. For ans we have $\mathcal{C} = \emptyset$ and over $\mathcal{A} \cup \mathcal{T}^+$ it will compute $\mathcal{A}_s = \mathcal{A} \cup \{A(a), A(b), A'(b)\}$. As can be seen, due to $A'(b) \in \mathcal{A}_s$ and $\neg A'(x)$ in $\mathcal{Q}'$ we can drop atom $\neg B'(x)$. Indeed, evaluating $\mathcal{Q}'' = B(x) \wedge \neg A'(x)$ over $\mathcal{A}_s$ computes $\emptyset$ as required. ♢

It follows that some negative atoms added by ref can be discarded.

Definition 11. Let $\mathcal{T}$ be a $\mathcal{L}$-TBox and let $\mathcal{Q}$ be a CQ. An atom $\neg A(x)$ in $\mathcal{Q}$ is called *covered* if there exists some other atom $\neg B(x)$ in $\mathcal{Q}$ such that $\mathcal{T} \models A \sqsubseteq B$. An atom $\neg R(x, y)$ in $\mathcal{Q}$ is called *covered* if there exists some other atom $\neg A(x)$, $\neg A(y)$, $\neg S(x, y)$, or $\neg S(y, x)$ in $\mathcal{Q}$ such that $\mathcal{T} \models \exists R \sqsubseteq A$, $\exists R^- \sqsubseteq A$, $R \sqsubseteq S$, or $R \sqsubseteq S^-$, respectively. We denote with $\mathsf{min}(\mathcal{Q}, \mathcal{T})$ the query derived from $\mathcal{Q}$ by removing all the covered atoms. Moreover, for a UCQ $\mathcal{R}$ we define $\mathsf{ref}_{\mathsf{min}}(\mathcal{R}, \mathcal{T}) = \bigcup_{\mathcal{Q}' \in \mathsf{ref}(\mathcal{R}, \mathcal{T})} \{\mathsf{min}(\mathcal{Q}', \mathcal{T})\}$.

Theorem 12. *Let $\mathcal{Q}$ be a CQ, let $\mathcal{T}$ be a DL-Lite-TBox, let $\mathcal{A}$ be an ABox and let $\langle \mathcal{R}_Q, \mathcal{R}_D \rangle$ be a rewriting for $\mathcal{Q}, \mathcal{T}$. Let also* ans *be an $\mathcal{RL}$ ABox-saturation system and let $\mathcal{TC} = \mathcal{T}^+ \cup \mathcal{C}$ for $\mathcal{C}$ a completion of $\mathcal{T}$ for* ans*. Then,*

$$\mathsf{cert}^{\mathsf{ic}}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}) = \mathsf{ans}(\mathsf{ref}_{\mathsf{min}}(\mathcal{R}_Q, \mathcal{T}), \mathcal{TC} \cup cr(\mathcal{A}, \mathcal{T}))$$

As we will see in the evaluation section, the above minimisation improves performance significantly.

Algorithm 1 $\mathsf{ApproxAns}(\mathcal{T}, \mathcal{A})$

Input: TBox $\mathcal{T}$, ABox $\mathcal{A}$; **Output:** Saturation $\mathcal{A}_s$

1: $\mathcal{A}_s := cr(\mathcal{A}, \mathcal{T})$
2: $\mathcal{P} := \mathsf{toRules}(\mathcal{T}^+|_{\mathsf{rl}})$
3: **while** No new assertions are added to $\mathcal{A}_s$ **do**
4: $\quad \mathcal{A}_t := \emptyset$
5: $\quad$ **for all** $\mathcal{B} \to H \in \mathcal{P}$ s.t. $\mathcal{B}\sigma \subseteq \mathcal{A}_s$ for some σ **do**
6: $\quad\quad$ **if** $\mathcal{B}\sigma$ is consistent w.r.t. $\mathcal{T}$ **do**
7: $\quad\quad\quad \mathcal{A}_t := \mathcal{A}_t \cup \{H\sigma\}$
8: $\quad \mathcal{A}_s := \mathcal{A}_s \cup \mathcal{A}_t$
9: **return** $\mathcal{A}_s$

4 ICAR-Answering Over Expressive DLs

Even for DLs with polynomial data complexity, like $\mathcal{EL}_\bot$, query answering under the ICAR semantics has been shown to be coNP-hard [Rosati, 2011] and tractability *cannot* be recovered even after syntactic restrictions [Rosati, 2011].

Our approach can form the basis for approximate algorithms for computing ICAR-based answers over more expressive DLs. However, as the following example shows, directly applying it can provide with counterintuitive results.

Example 13. Consider the following $\mathcal{EL}_\bot$-TBox $\mathcal{T} = \{A \sqcap B \sqsubseteq C,\ A \sqcap B \sqsubseteq \bot\}$, ABox $\mathcal{A} = \{A(a), B(a)\}$, and query $\mathcal{Q} = C(x)$. Clearly, we have $clc(\mathcal{T}, \mathcal{A}) = \mathcal{A}$ and hence $\mathsf{cert}^{\mathsf{ic}}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}) = \emptyset$.

Now, since $\mathcal{T}|_{\mathsf{rl}} = \{A \sqcap B \sqsubseteq C\}$ any $\mathcal{RL}$ ABox-saturation system ans would compute for $\mathcal{T}^+ \cup \mathcal{A}$ the saturation $\mathcal{A}_s = \mathcal{A} \cup \{C(a)\}$. Finally, since $\mathsf{ref}(\mathcal{Q}, \mathcal{T}) = \{\mathcal{Q}\}$ then we will have $\mathsf{cert}(\mathsf{ref}(\mathcal{Q}, \mathcal{T}), \mathcal{A}_s) = \{a\}$. ♢

Consequently, in the case of more expressive languages, ABox-saturation systems compute far too many assertions and hence answers. To control them one has to either extend the ref procedure to include additional negative atoms (e.g., $\neg A(x) \wedge \neg B(x)$ in the previous example) or prevent some assertions from being derived in the first place (e.g., $C(a)$).

We follow the second approach and propose Algorithm 1 which saturates the input ABox over the $\mathcal{RL}$ fragment of the given TBox by also taking into account the negative inclusions. More precisely, the algorithm first translates the $\mathcal{RL}$ fragment of $\mathcal{T}^+$ into datalog rules $\mathcal{P}$ and then saturates $\mathcal{A}$ using $\mathcal{P}$. However, every time some rule of $\mathcal{P}$ "fires" (Line 5), before adding the conclusion $H\sigma$ to the saturation, in Line 6, it performs a consistency check in order to restrict the amount of assertions that are derived by the saturation procedure.

Clearly, over DL-Lite-TBoxes $\mathcal{T}$ Algorithm 1 computes $clc(\mathcal{T}, \mathcal{A})$ for any $\mathcal{A}$.

Proposition 14. *Let $\mathcal{T}$ be a DL-Lite-TBox, let $\mathcal{A}$ be an ABox, and let $\mathcal{Q}$ be a CQ. Let also* ans *be some $\mathcal{RL}$ ABox-saturation system and $\mathcal{C}$ a completion of $\mathcal{T}$ for* ans*. Then, $clc(\mathcal{T}, \mathcal{A}) = \mathsf{ApproxAns}(\mathcal{T} \cup \mathcal{C}, \mathcal{A})$.*

However, for more expressive DLs the algorithm approximates $clc(\mathcal{T}, \mathcal{A})$ from above.

Theorem 15. *Let $\mathcal{T}$ be a $\mathcal{L}$-TBox, let* ans *be an $\mathcal{RL}$ ABox-saturation system, and let $\mathcal{C}$ be a completion of $\mathcal{T}$ for* ans*. Then, we have $clc(\mathcal{T}, \mathcal{A}) \subseteq \mathsf{ApproxAns}(\mathcal{T} \cup \mathcal{C}, \mathcal{A})$*

To make Algorithm 1 exact, i.e., compute $clc(\mathcal{T}, \mathcal{A})$, for every assertion $H\sigma$ inferred at Line 5 one should check if some subset $\mathcal{A}'$ of the *original* ABox $\mathcal{A}$ consistent w.r.t. $\mathcal{T}$ exists such that $\mathcal{T} \cup \mathcal{A}' \models \mathcal{B}\sigma$ (and hence also $\mathcal{T} \cup \mathcal{A}' \models H\sigma$). In contrast, our algorithm provides a kind of "local" check to decide whether $H\sigma$ can be added to $\mathcal{A}_s$, i.e., it checks if $\mathcal{B}\sigma$ is consistent w.r.t. the *current* set of assertions (saturation).

Example 16. Consider the following TBox and ABox:

$$\begin{aligned} \mathcal{T} &= \{A \sqcap B \sqsubseteq C,\ D \sqsubseteq B,\ A \sqsubseteq \neg D\} \\ \mathcal{A} &= \{A(a), D(a)\} \end{aligned}$$

Then we have $clc(\mathcal{T}, \mathcal{A}) = \mathcal{A} \cup \{B(a)\}$ but, in contrast, Algorithm 1 would first compute and *add* $B(a)$ to the saturation $\mathcal{A}_s$ and also compute $C(a)$ due to the consistent subset $\{A(a), B(a)\} \subseteq \mathcal{A}_s$. ♢

As can be seen, if we explicitly add $B(a)$ to $\mathcal{A}$ then we would also have $C(a) \in clc(\mathcal{T}, \mathcal{A})$. In other words, materialising consistently entailed assertions can strictly increase the set computed by clc (and hence also the ICAR-answers), something that is not true under the standard first-order semantics where for every Σ and ϕ such that $\Sigma \models \phi$, Σ and $\Sigma \cup \{\phi\}$ are equivalent.

Based on the above, we feel that it is intuitive to introduce an extension of function clc which is mostly characterised by our approximate algorithm.

Definition 17. Let $\mathcal{T}$ be a $\mathcal{L}$-TBox and let $\mathcal{A}$ be an ABox. We define $clc^+(\mathcal{T}, \mathcal{A})$ to be the minimal set of assertions satisfying the following conditions:

- $cr(\mathcal{A}, T) \subseteq clc^+(\mathcal{T}, \mathcal{A})$
- If some $\mathcal{A}' \subseteq clc^+(\mathcal{T}, \mathcal{A})$ consistent w.r.t. $\mathcal{T}$ exists such that $\mathcal{T} \cup \mathcal{A}' \models \alpha$, then $clc^+(\mathcal{T}, \mathcal{A})$ contains α.

Differently than clc, in clc^+ any consistently entailed assertion can be used to support the entailment of other assertions. Hence, the following property follows easily.

Proposition 18. *For a $\mathcal{L}$-TBox $\mathcal{T}$ and ABox $\mathcal{A}$ we have $clc(\mathcal{T}, \mathcal{A}) \subseteq clc^+(\mathcal{T}, \mathcal{A})$.*

Moreover, in contrast to clc, for many DLs clc^+ can be computed in polynomial time.

Theorem 19. *Let $\mathcal{L}$ be some DL such that deciding consistency of ABoxes can be done in polynomial time and $\mathcal{L}$ is also datalog rewritable. Then, for every $\mathcal{L}$-TBox $\mathcal{T}$ and ABox $\mathcal{A}$ $clc^+(\mathcal{T}, \mathcal{A})$ can be computed in polynomial time with respect to the size of the data.*

The following is an immediate consequence of Theorem 19 and various results in datalog rewritability of tractable Horn-DLs [Calvanese *et al.*, 2007; Pérez-Urbina *et al.*, 2010; Hustadt *et al.*, 2005] as well as highly expressive non-Horn DLs [Kaminski *et al.*, 2014].

Corollary 20. *Let $\mathcal{L}$ be any of the following DLs:*

- *DL-Lite, $\mathcal{ELHI}_\bot$, or Horn-$\mathcal{SHIQ}$; or*
- markable-$\mathcal{SHI}$ *or $\mathcal{ALCHI}$ that is simple w.r.t. disjunctive predicates [Kaminski* et al.*, 2014].*

Then, for every $\mathcal{L}$-TBox $\mathcal{T}$ and ABox $\mathcal{A}$, $clc^+(\mathcal{T}, \mathcal{A})$ can be computed in polynomial time with respect to the size of $\mathcal{A}$.

Algorithm 2 $\mathsf{ABoxIARRepair}(\mathcal{T}, \mathcal{A})$

Input: TBox $\mathcal{T}$, ABox $\mathcal{A}$; **Output:** IAR of $\mathcal{T} \cup \mathcal{A}$

1: $\mathcal{A} := cr(\mathcal{A}, \mathcal{T})$
2: $\Phi := \emptyset$
3: **for all** $C \sqsubseteq \neg D \in \mathcal{T}$ **do** add query $\pi(C) \wedge \pi(D)$ to Φ
4: **for all** $R \sqsubseteq \neg S \in \mathcal{T}$ **do** add query $\pi(S) \wedge \pi(R)$ to Φ
5: **for all** $\mathcal{Q}_\phi \in \Phi$ **do**
6: Compute a UCQ rewriting $\mathcal{R}_\phi$ for $\mathcal{Q}_\phi, \mathcal{T}$
7: **for all** $\mathcal{Q}'_\phi \in \mathcal{R}_\phi$ and all σ s.t. $\mathcal{Q}'_\phi\sigma \subseteq \mathcal{A}$ **do**
8: Remove $\mathcal{Q}'_\phi\sigma$ from $\mathcal{A}$
9: **return** $\mathcal{A}$

Concluding this section we note that the output of Algorithm 1 can (possibly) be used to compute some upper approximation of the ICAR-answers. More precisely, for some CQ $\mathcal{Q}$ we can compute a rewriting $\langle \mathcal{R}_Q, \mathcal{R}_D \rangle$ for $\mathcal{Q}, \mathcal{T}$ (if it exists) and then calculate $\mathsf{cert}(\mathsf{ref}(\mathcal{R}_Q, \mathcal{T}), \mathcal{A}_s)$. However, note that ref would also be an approximation as, to the best of our knowledge, it has only been defined for DL-Lite.

5 Efficient QA under the IAR Semantics

Concluding our technical contributions we show that our framework can also be used to compute IAR-answers. More precisely, by the results of Section 3 and the results by Rosati [2011] we have the following.

Proposition 21. *Let $\mathcal{T}$ be a $\mathcal{L}$-TBox, let $\mathcal{A}$ be an ABox, let $\mathcal{Q}$ be a CQ, and let $\mathcal{A}^{\mathsf{ia}}$ be the IAR of $\mathcal{T} \cup \mathcal{A}$. Let also* ans *be an $\mathcal{RL}$ ABox-saturation system, let $\mathcal{C}$ be a completion of $\mathcal{T}$ for* ans *and let $\langle \mathcal{R}_Q, \mathcal{R}_D \rangle$ be a rewriting for $\mathcal{Q}, \mathcal{T}$. Then, we have $\mathsf{cert}^{\mathsf{ia}}(\mathcal{Q}, \mathcal{T} \cup \mathcal{A}) = \mathsf{ans}(\mathcal{R}_Q, \mathcal{T} \cup \mathcal{C} \cup \mathcal{A}^{\mathsf{ia}})$.*

Since completions of DL-Lite and $\mathcal{EL}$ TBoxes for $\mathcal{RL}$ ABox-saturation systems always exist [Stoilos *et al.*, 2011] and Rosati [2011] showed that for TBoxes expressed in DL-Lite and $\mathcal{EL}_{\bot nr}$ the IAR $\mathcal{A}^{\mathsf{ia}}$ of some ABox can be computed in time polynomially w.r.t. the size of $\mathcal{A}$, it follows that our approach can compute the IAR-answers over these languages effectively by exploiting practically scalable $\mathcal{RL}$ systems. Unfortunately, even in these cases computing $\mathcal{A}^{\mathsf{ia}}$ can still be difficult and time-consuming if $\mathcal{A}$ is very large.

For DL-Lite TBoxes, an algorithm to compute $\mathcal{A}^{\mathsf{ia}}$ was proposed and implemented in [Rosati *et al.*, 2012]. This algorithm annotates all assertions in $\mathcal{A}$ and then updates their annotation if they violate some negative inclusions of $\mathcal{T}$. All assertions whose annotation changed are finally removed.

Inspired by the techniques in [Lembo *et al.*, 2015] we propose a different algorithm that is based on query rewriting and not on ABox annotation; this is depicted in Algorithm 2. The algorithm builds a special query for each negative inclusion in $\mathcal{T}$; π maps a concept A to $A(x)$, role R to $R(x, y)$ and a concept $\exists R$ to $\exists y.R(x, y)$. Then, a rewriting for $\mathcal{Q}_\phi, \mathcal{T}$ is computed and each member of the rewriting is evaluated over $\mathcal{A}$. If some of these queries match to assertions in $\mathcal{A}$, then these should be removed from $\mathcal{A}$ as they violate the negative inclusion.

Proposition 22. *For a given DL-Lite-TBox $\mathcal{T}$ and ABox $\mathcal{A}$, $\mathsf{ABoxIARRepair}(\mathcal{T}, \mathcal{A})$ returns an IAR of $\mathcal{T} \cup \mathcal{A}$.*

Algorithm 2 is amenable to an important optimisation. Instead of evaluating $\mathcal{Q}'_\phi$ over $\mathcal{A}$ (Lines 7–8) we can proceed as follows: at a pre-processing step we can evaluate all atomic queries $A(x)$ and $R(x, y)$ separately and cache their answers and the assertions of $\mathcal{A}$ that these atoms have a match. Then, to compute the answers of $\mathcal{Q}'_\phi$ over $\mathcal{A}$ we can use the pre-computed partial answers of each atom.

6 Evaluation

We implemented our ICAR- (both the standard and the optimised one) and IAR-answering approaches into the prototype system SaQAI[2] (Saturation based Query Answering under Inconsistencies); in the following the various versions of SaQAI (standard/optimised ICAR and IAR) are called $\mathsf{SaQ}^{\mathsf{ic}}$, $\mathsf{SaQ}^{\mathsf{ic}}_{\mathsf{op}}$, and $\mathsf{SaQ}^{\mathsf{ia}}$, respectively. Unfortunately, implementing Algorithm 1 would either require modifying the internals of an ABox-saturation system or implementing one from scratch. Our system uses GraphDB [Kiryakov *et al.*, 2010] as an ABox-saturation system, Hydrowl [Stoilos, 2014] to compute completions, and Rapid [Trivela *et al.*, 2015] for rewriting.

For the evaluation we used the experimental setting proposed in [Bienvenu *et al.*, 2014] which consists of a DL-Lite version of the $\mathrm{LUBM}^{\exists}_{20}$ ontology [Lutz *et al.*, 2013] extended with additional negative inclusions, a set of test queries, and several inconsistent ABoxes. We use the same ABox and query names as in [Bienvenu *et al.*, 2014]. For example, $\mathcal{A}^m_n$ indicates an ABox containing data for n universities and inconsistencies added with probability m. Due to space limitations we will present results only for some of the test queries and for the larger datasets. To the best of our knowledge no available ICAR-answering system exists (we could not obtain the preliminary system reported in [Masotti *et al.*, 2011]), hence we compared against CQApri [Bienvenu *et al.*, 2014] and QuID [Rosati *et al.*, 2012], two IAR-answering systems.

Computing a completion of $\mathrm{LUBM}^{\exists}_{20}$ for GraphDB was done only once and required less than 5 seconds. Moreover, no unsatisfiable concepts were detected, hence $cr(\mathcal{A}^m_n, \mathcal{T})$ also completed in less than a second.

Table 1 presents the results for all considered approaches and tools. In the table we have grouped together number of answers, pre-processing times (note that different systems can perform different pre-processing tasks) and query evaluation. Column Sat denotes loading time of $\mathcal{T} \cup \mathcal{C} \cup \mathcal{A}^m_n$ into GraphDB and computation of the saturation $\mathcal{A}_s \supseteq \mathcal{A}^m_n$. First, we can note that, as expected, in many queries our ICAR-answering approach returned far more answers than CQApri ($\mathsf{SaQ}^{\mathsf{ia}}$ returned the same IAR-answers as CQApri, however, QuID returned different answers indicating some kind of bug but we did not investigate further). There are actually cases where CQApri returns 0 answers whereas our approach returns some answers to the input query. Second, we can observe that our optimisations in Section 3.1 are indeed quite relevant and improved performance of our ICAR-answering approach significantly. This is because, in most cases $\mathsf{ref}_{\mathsf{min}}$ (which $\mathsf{SaQ}^{\mathsf{ic}}_{\mathsf{op}}$ uses) computes queries containing much fewer negative atoms than ref; e.g., in query q_1 it computes 32

[2] http://image.ece.ntua.gr/~etsalap/SaQAI

Table 1: Results for $\mathsf{SaQ}^{\mathsf{ic}}$, $\mathsf{SaQ}^{\mathsf{ic}}_{\mathsf{op}}$, and CQApri (CQA).

		Number of Answers		**Loading, Pre-processing, and Evaluation (in seconds)**								
				ICAR Answering			*IAR Answering*					
							Repairing/Pre-proc.			Evaluation Time		
$\mathcal{A}$	$\mathcal{Q}$	$\mathsf{SaQ}^{\mathsf{ic}}$	CQA	Sat	$\mathsf{SaQ}^{\mathsf{ic}}$	$\mathsf{SaQ}^{\mathsf{ic}}_{\mathsf{op}}$	$\mathsf{SaQ}^{\mathsf{ia}}$	QuID	CQA	$\mathsf{SaQ}^{\mathsf{ia}}$	QuID	CQA
$\mathcal{A}_{10}^{15e\text{-}4}$	q_1	255,839	251,991	19.6	47.5	4.6	52.9	1,437.4	58.1	0.2	17.6	11.4
	q_2	88,994	88,816		22.3	1.6				<0.1	12.2	6.9
	q_4	966,856	769,786		1,060.3	99.3				1.0	14.1	17.9
	req_3	2,242	2,228		2.4	0.3				0.1	31.2	2.1
	lutz_1	189,519	189,519		227.6	21.1				0.9	190.9	61.3
	lutz_5	38,244	38,244		33.5	2.2				0.1	84.0	48.6
$\mathcal{A}_{10}^{5e\text{-}2}$	q_1	250,320	145,488	20.1	51.9	5.0	53.8	2,072.3	65.5	0.1	31.0	13.9
	q_2	80,803	73,453		20.2	1.5				<0.1	8.9	7.8
	q_4	540,237	1,001		683.1	65.3				0.1	1.0	26.4
	req_3	1,221	1,012		1.7	0.3				<0.1	9.6	4.1
	lutz_1	5,249	5,249		118.4	11.6				0.3	128.7	83.0
	lutz_5	25,168	25,163		29.3	2.1				0.1	77.0	49.6
$\mathcal{A}_{10}^{2e\text{-}1}$	q_1	233,153	35,086	21.2	62.5	10.3	55.2	3,205.0	84.6	<0.1	8.9	17.1
	q_2	61,171	35,762		16.9	1.3				<0.1	6.5	13.6
	q_4	88,323	0		161.2	15.5				<0.1	0.7	29.1
	req_3	197	98		1.5	0.2				<0.1	2.3	4.1
	lutz_1	0	0		70.6	6.9				0.1	78.7	132.3
	lutz_5	6,927	6,797		24.1	1.6				0.1	60.7	61.9
$\mathcal{A}_{20}^{15e\text{-}4}$	q_1	544,725	535,341	59.7	120.1	13.8	114.2	3,184.3	124.6	0.4	89.9	31.1
	q_2	189,527	189,209		54.9	4.6				0.1	46.7	17.2
	q_4	2,033,569	1,355,978		2,701.8	286.6				1.7	43.4	49.8
	req_3	4,851	4,823		4.8	0.8				0.4	94.9	4.5
	lutz_1	414,600	414,600		514.0	60.2				2.4	892.5	136.1
	lutz_5	81,996	81,996		84.0	6.6				0.5	267.2	118.4
$\mathcal{A}_{20}^{5e\text{-}2}$	q_1	532,587	309,625	61.2	146.8	15.0	113.9	5,246.0	136.8	0.2	72.4	44.0
	q_2	172,121	156,361		56.6	4.4				<0.1	34.9	25.5
	q_4	1,126,433	0		1,921	178.3				0.2	1.3	146.8
	req_3	2,569	2,125		4.8	0.8				0.2	44.4	9.1
	lutz_1	15,539	15,539		336.8	33.3				0.9	202.9	238.7
	lutz_5	53,430	53,420		82.9	6.0				0.3	149.7	123.5
$\mathcal{A}_{20}^{2e\text{-}1}$	q_1	497,164	73,611	66.3	149.0	16.7	122.1	t/o	205.7	0.2	-	42.9
	q_2	131,228	77,475		42.1	3.6				<0.1	-	23.9
	q_4	174,226	0		418.5	38.8				<0.1	-	oom
	req_3	407	184		3.2	0.6				<0.1	-	22.0
	lutz_1	0	0		175.4	19.7				0.3	-	266.2
	lutz_5	14,671	14,355		58.7	4.5				0.2	-	134.0

whereas ref computes 353 negative atoms. Third, although $\mathsf{SaQ}^{\mathsf{ic}}_{\mathsf{op}}$ computes more answers (the ICAR-answers) than the other systems, it is significantly faster in nearly all queries compared to QuID and CQApri with only noticeable exception query q_4. Nevertheless, besides the fact that our system computes more involved semantics, for the same query over ABox $\mathcal{A}_{20}^{2e\text{-}1}$ CQApri run out of memory and QuID failed to load the dataset within 90 minutes.

Finally, our method for computing the IAR-answers is also significantly more efficient than QuID and CQApri. First, our ABox repairing method is several orders of magnitude faster than QuID which is based on the ABox annotation algorithm in [Rosati *et al.*, 2012] (recall that QuID actually did not manage to load the largest ABox $\mathcal{A}_{20}^{2e\text{-}1}$) and is also much faster than the pre-processing step of CQApri. Second, after the repairing step and due to Proposition 21 our system computes the IAR-answers by standard query evaluation using GraphDB (since the ABox is repaired the ref procedure is not required) and this takes in all cases just a few milliseconds in contrast to QuID and CQApri that still require several seconds to evaluate the queries they have computed and construct the IAR-answers. All in all both our approaches are much more efficient and robust.

7 Conclusions

We proposed a framework for efficient and scalable IAR- and ICAR-answering over inconsistent Description Logic knowledge bases which is based on mature data saturation technologies. The approach is exact for both semantics over DL-Lite and for the IAR semantics over $\mathcal{EL}_{\bot nr}$ ontologies. For more expressive DLs we proposed an algorithm that computes an approximation of ICAR-answers but which characterises a new family of semantics that can be computed in polynomial time for a very large family of DLs. Our experiments provided with encouraging results as both of our approaches are more efficient and robust compared to the state-of-the-art.

References

[Arenas *et al.*, 1999] Marcelo Arenas, Leopoldo E. Bertossi, and Jan Chomicki. Consistent query answers in inconsistent databases. In *Proceedings of the 18th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*, pages 68–79, 1999.

[Baader *et al.*, 2003] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[Baader *et al.*, 2005] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05)*, 2005.

[Bertossi, 2006] Leopoldo E. Bertossi. Consistent query answering in databases. *SIGMOD Record*, 35(2):68–76, 2006.

[Bienvenu and Rosati, 2013] Meghyn Bienvenu and Riccardo Rosati. New inconsistency-tolerant semantics for robust ontology-based data access. In *Proceedings of the 26th International Workshop on Description Logics*, 2013.

[Bienvenu *et al.*, 2014] Meghyn Bienvenu, Camille Bourgaux, and François Goasdoué. Querying inconsistent description logic knowledge bases under preferred repair semantics. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 996–1002, 2014.

[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*, 39(3):385–429, 2007.

[Chaussecourte *et al.*, 2013] Pierre Chaussecourte, Birte Glimm, Ian Horrocks, Boris Motik, and Laurent Pierre. The energy management adviser at EDF. In *Proceedings of the 12th International Semantic Web Conference (ISWC)*, 2013.

[Hustadt *et al.*, 2005] Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Data Complexity of Reasoning in Very Expressive Description Logics. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI 2005)*, pages 466–471, 2005.

[Kaminski *et al.*, 2014] Mark Kaminski, Yavor Nenov, and Bernardo Cuenca Grau. Computing datalog rewritings for disjunctive datalog programs and description logic ontologies. In *Proceedings of the 8th International Conference on Web Reasoning and Rule Systems (RR)*, 2014.

[Kiryakov *et al.*, 2010] Atanas Kiryakov, Barry Bishop, Damyan Ognyanoff, Ivan Peikov, Zdravko Tashev, and Ruslan Velkov. The features of BigOWLIM that enabled the BBC's World Cup website. In *Workshop on Semantic Data Management (SemData)*, 2010.

[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)*, pages 103–117, 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 Web Reasoning and Rule Systems (RR)*, pages 155–169, 2011.

[Lembo *et al.*, 2015] Domenico Lembo, Maurizio Lenzerini, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. Inconsistency-tolerant query answering in ontology-based data access. *Journal of Web Semantics*, 33:3–29, 2015.

[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)*, pages 314–330, 2013.

[Masotti *et al.*, 2011] Giulia Masotti, Riccardo Rosati, and Marco Ruzzi. Practical abox cleaning in dl-lite (progress report). In *Proceedings of the 24th International Workshop on Description Logics (DL)*, 2011.

[Motik *et al.*, 2009] Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, and Carsten Lutz (Editors). OWL 2 Web Ontology Language Profiles, 2009.

[Motik *et al.*, 2012] Boris Motik, Ian Horrocks, and Su Myeon Kim. Delta-reasoner: A semantic web reasoner for an intelligent mobile platform. In *Proceedings of the 21st International Conference Companion on World Wide Web (WWW 12)*. ACM, 2012.

[Pérez-Urbina *et al.*, 2010] Héctor Pérez-Urbina, Boris Motik, and Ian Horrocks. Tractable query answering and rewriting under description logic constraints. *Journal of Applied Logic*, 8(2):186–209, 2010.

[Rosati *et al.*, 2012] Riccardo Rosati, Marco Ruzzi, Mirko Graziosi, and Giulia Masotti. Evaluation of techniques for inconsistency handling in OWL 2 QL ontologies. In *Proceedings of the 11th International Semantic Web Conference (ISWC)*, pages 337–349, 2012.

[Rosati, 2011] Riccardo Rosati. On the complexity of dealing with inconsistency in description logic ontologies. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1057–1062, 2011.

[Stoilos *et al.*, 2011] Giorgos Stoilos, Bernardo Cuenca Grau, Boris Motik, and Ian Horrocks. Repairing ontologies for incomplete reasoners. In *Proceedings of the 10th International Semantic Web Conference (ISWC-11), Bonn, Germany*, pages 681–696, 2011.

[Stoilos, 2014] Giorgos Stoilos. Ontology-based data access using rewriting, OWL 2 RL systems and repairing. In *Proceedings of the 11th European Semantic Web Conference (ESWC)*, pages 317–332, 2014.

[Trivela *et al.*, 2015] Despoina Trivela, Giorgos Stoilos, Alexandros Chortaras, and Giorgos Stamou. Optimising resolution-based rewriting algorithms for owl ontologies. *Journal of Web Semantics*, 33:30–49, 2015.