On Instance-Level Update and Erasure in Description Logic Ontologies

Giuseppe De Giacomo, Maurizio Lenzerini,
Antonella Poggi, Riccardo Rosati

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Sapienza Università di Roma
Via Ariosto 25, 00185 Roma, Italy
lastname@dis.uniroma1.it

Abstract

A Description Logic ontology is constituted by two components, a TBox that expresses general knowledge about the concepts and their relationships, and an ABox that describes the properties of individuals that are instances of concepts. We address the problem of how to deal with changes to a Description Logic ontology, when these changes affect only the ABox, i.e., when the TBox is considered invariant. We consider two basic changes, namely instance-level update and instance-level erasure, roughly corresponding to the addition and the deletion of a set of facts involving individuals. We characterize the semantics of instance-level update and erasure on the basis of the approaches proposed by Winslett and by Katsuno and Mendelzon. Interestingly, Description Logics are typically not closed with respect to instance-level update and erasure, in the sense that the set of models corresponding to the application of any of these operations to a knowledge base in a Description Logic $\mathcal{L}$ may not be expressible by ABoxes in $\mathcal{L}$. In particular, we show that this is true for *DL-Lite*$_{\mathcal{F}}$, a tractable DL that is oriented towards data intensive applications. To deal with this problem, we first introduce *DL-Lite*$_{\mathcal{FS}}$, a DL that minimally extends *DL-Lite*$_{\mathcal{F}}$ and is closed with respect to instance-level update, and present a polynomial algorithm for computing instance-level update in this logic. Then, we provide a principled notion of best approximation with respect to a fixed language $\mathcal{L}$ of instance-level update and erasure, and exploit the algorithm for instance-level update for *DL-Lite*$_{\mathcal{FS}}$ to get polynomial algorithms for approximated instance-level update and erasure for *DL-Lite*$_{\mathcal{F}}$. These results confirm the nice computational properties of *DL-Lite*$_{\mathcal{F}}$ for data intensive applications, even where information about instances is not only read, but also written.

1 Introduction

Several areas of Computer Science and various application domains have witnessed a growing interest in ontologies in the last years. In particular, ontologies are considered

as one of the key concepts in the Semantic Web [7], where they can be used to describe the semantics of information at various sites, overcoming the problem of implicit and hidden knowledge, and thus enabling content exchange. Ontologies are also advocated as appropriate means to integrate data and services. Indeed, in the information integration scenario [24], ontologies can be profitably used to express the so-called global schema, which represents the reconciled and unified view of all the local resources (data or services) to be integrated [29].

It is widely accepted that Description Logics [3] (DL) may provide a solid foundation for both expressing ontologies with a logical formalism, and for reasoning about the knowledge represented in the ontologies. A knowledge base in DL is constituted by two components, called TBox and ABox. In a DL ontology, the former expresses the intensional level of the ontology, i.e., the general knowledge about the concepts and their relationships, whereas the latter is the extensional level of the ontology, that describes one state of affairs regarding the instances of concepts and relationships. One of the advantages of considering an ontology as a knowledge base expressed in DL is that we can re-phrase system services of ontology tools in terms of logical reasoning problems. In turn, this view allows us to exploit the whole body of research on algorithms for and complexity of reasoning in DLs, in the endeavor to build well-founded tools supporting inferences over ontologies [1, 4]. Indeed, current results on DLs may directly provide effective techniques to be incorporated in ontology management tools [22, 18, 33] to deal, for example with consistency and subsumption checking, query answering, or instance recognition. However, such results cannot actually be used to support other important tasks. One notable example of such tasks is ontology evolution.

By evolution we mean here both update and erasure, which are operations addressing the need of changing an ontology in order to reflect a change in the domain of interest the ontology is supposed to represent. Generally speaking, an update specifies a set of properties that must be valid in the state resulting from the change, whereas an erasure is intended to specify a set of properties that should not be valid in such state. One of the major challenges when dealing with an update is how to react to the case where the set of properties specified in an update is inconsistent with the current knowledge. Since a principled approach to this issue in the context of ontologies is missing, existing ontology management tools[1] adopt ad-hoc solutions to this problem, for example, just rejecting the update. Similarly, such tools do not provide formal mechanisms for dealing with erasure.

Although the problem of ontology evolution in its generality should consider the case of performing updates and erasures on the whole knowledge base [14, 27], i.e., either the TBox, or the ABox, or both, in this paper we restrict our attention to what we call *instance-level update and erasure*. In instance-level update (erasure) the ontology is specified by both a TBox and an ABox, but the update (erasure) affects only the ABox, in the sense that we enforce the condition that the ontology resulting from applying the evolution operations has the same TBox as the original ontology.

Although simplified with respect to the general case, we believe that this setting not only allows us to study the fundamental properties of the two operations, but is very relevant in practice. Indeed, in many applications, the intensional level of the

[1] `http://www.xml.com/2002/11/06/Ontology_Editor_Survey.html`.

ontology is quite stable, as its evolution represents a change or a refinement in the conceptualization of the domain of interest. On the other hand, the instance level contains information about specific instances, and hence, typically, changes much more frequently than the intensional level. In a sense, such kind of changes are similar to changes of the data in a database[2]. So, instance-level evolution is very close to the classical notion of update in databases, and their study can therefore shed light on the difference between database update and ontology evolution.

The aim of our work is a systematic investigation on instance-level update and erasure in DL ontologies. To the best of our knowledge, this is the first investigation of this type. The main contributions of the paper are as follows.

1. We formally define the notion of instance-level update and erasure of ontologies. Building on classical approaches on knowledge base update and erasure, we provide a general semantics for instance level evolution of DL ontologies. In particular, we follow the approach of [26], and we adapt Winslett's semantics [34, 35] for update, and Katsuno and Mendelzon's semantics [23] for erasure, to the case where the ontology is described by both a TBox and an ABox. As in the above mentioned approach, in our framework the result of an update and an erasure is given in terms of a set of models of the knowledge base used to express the ontology.

2. We study instance-level update in the case where the ontology is expressed in *DL-Lite*$_{\mathcal{F}}$ [8]. One of the main features of *DL-Lite*$_{\mathcal{F}}$ is that all reasoning tasks can be done in polynomial time with respect to the size of the ontology, and our goal was to verify whether the nice computational property of reasoning in this DL extends to ontology evolution. We point out a fundamental difficulty in this context: there are cases where the set of models characterizing an instance-level update of a *DL-Lite*$_{\mathcal{F}}$ ontology cannot be captured in *DL-Lite*$_{\mathcal{F}}$, i.e., this logic is *not* closed with respect to instance-level update. Observe that a similar phenomenon was observed in [26] for a more expressive DL. We then single out the minimal DL, called *DL-Lite*$_{\mathcal{FS}}$, that extends *DL-Lite*$_{\mathcal{F}}$ and is closed with respect to update, and present an efficient algorithm for computing the DL ontology resulting from an instance-level update in this logic.

3. One of the original motivations of our work was to add update facilities to QuOnto [1], that is a reasoning system for *DL-Lite*$_{\mathcal{F}}$ implementing efficient algorithms for both TBox reasoning and query answering. Like all other DL reasoners, QuOnto is based on a specific logic, i.e., the knowledge bases that it is able to manage must be expressed in *DL-Lite*$_{\mathcal{F}}$. The non-expressibility issue is obviously a problem for such systems. In order to cope with this problem, the brute-force approach would be the one that refuses an update whenever the resulting ontology cannot be expressed in the logic underlying the system. In this paper, we propose a different approach, based on the notion of approximation. In particular, our proposal is to address the problem by computing the *DL-Lite*$_{\mathcal{F}}$ knowledge base that approximates at best the set of models resulting from the

[2] In the database setting, changes at the intensional level would correspond to schema evolution or restructuring.

update. This is a general idea that might be pursued in every situation where an ontology management system based on a specific DL aims at supporting updates, but the DL is not closed with respect to such operation. We first introduce the notion of maximal approximation in DL, and then we present an efficient method for computing the maximal approximation of instance-level updates of *DL-Lite*$_{\mathcal{F}}$ ontologies.

4. Finally, we carry out a detailed study of instance-level erasure. To the best of our knowledge, this is the first work dealing with erasure in DLs. We show that, in general, erasures are not expressible in neither *DL-Lite*$_{\mathcal{F}}$ nor *DL-Lite*$_{\mathcal{FS}}$, and present a polynomial time algorithm for computing the maximal approximation of instance-level erasure in *DL-Lite*$_{\mathcal{F}}$.

The paper is organized as follows. In Section 2 we discuss related works. In Section 3 we provide a general overview of DL ontologies, and introduce *DL-Lite*$_{\mathcal{F}}$. In Section 4 we provide the formal definition of instance-level update, and then we present the non-expressibility result for *DL-Lite*$_{\mathcal{F}}$, as well as the update algorithm for *DL-Lite*$_{\mathcal{FS}}$. In Section 5 we introduce the notion of approximation that we use for coping with the non-expressiblity problem. In Section 6 we present the method for computing the maximal approximation of an update in *DL-Lite*$_{\mathcal{F}}$, and in Section 7 we present the results on instance-level erasure. Finally, Section 8 discusses interesting problems left open in our investigation.

This paper is an extended version (including also the proofs of theorems) of [11] and [12].

2 Related work

Several recent work addresses the issue of ontology evolution. In [19], the authors point out that one of the fundamental problems in dealing with ontology changes is how to guarantee consistency of the resulting ontology. They define the consistency of ontologies at three different levels, namely, structural, logical, and user-defined, and propose methods for resolving inconsistencies at these various levels, including resolution strategies when a change admits different consistent states.

A broad study of ontology evolution is presented in [14, 15, 16]. In particular, in [15], the authors propose a very general view of this problem, make connections between ontology evolution and several research disciplines (i.e., ontology versioning, alignment, mapping, integration), and present a comprehensive review of the recent literature on such disciplines. Interestingly, the authors of these papers often point out the importance of the AGM postulates for revision [2], and they study various aspects related to the application of the AGM theory to the problem of ontology evolution. Arguably, analogous postulates for update and erasure [23] should play an important role in ontology evolution.

As we said in the introduction, in this paper we study instance-level ontology evolution, and therefore, our work is closely related to [20, 17, 26]. In [20], the authors investigate the process of incrementally updating tableau completion graphs created

during consistency checking in expressive Description Logics, and present an algorithm for updating completion graphs under both the addition and removal of ABox assertions. Differently from our work, the paper follows a syntactic approach to updates.

On the contrary, both [17] and [26] adopt a semantic notion of update and erasure. In [17], erasure is studied for RDF, under the same semantics we use in the present paper, namely the Katsuno-Mendelson semantics [23]. In [5, 26] the authors propose a formal semantics for updates in DLs, and present interesting results on various aspects related to computing updates. In particular, [26] shows an example of non-expressibility of updates for the case of an expressive DL. However, since the problem of update is addressed in [5, 26] under the assumption that the knowledge base is specified only at the extensional level, i.e., with no TBox[3], the paper does not take into account the impact of the intensional level on ontology update.

As we said before, we follow the approaches to update and erasure developed in the Artificial Intelligence literature several years ago. Various approaches to update have been considered in literature; see, e.g., [13, 21] for surveys. Here, like in [5, 26], we essentially follow Winslett's approach [34, 35], originally developed for updates on databases in presence of incomplete information, and its conterpart, defined in [23], as notion of erasure.

The intuition behind such approach is the following. There is an actual state-of-affairs of the world of which, however, we have only an incomplete description. Such description identifies a (typically infinite) set of models, each corresponding to a state-of-affairs that we consider possible. Among them, there is one model corresponding to the actual state-of-affairs, but we do not know which. Now, we perform an update because the state-of-affairs is changed. However, since we do not really know which of our models corresponds to the actual state-of-affairs, we apply the change on every possible model, thus getting a new set of models representing the updated situation. Among them, we do have the model corresponding to performing the update on the actual state-of-affairs, but again we do not know which. As for how we perform the update on each model, the idea is that we apply exactly those changes that are absolutely necessary for accommodating what explicitly asserted in the specified update.

Observe that this intuition is essentially the one behind most of the research on reasoning about actions. For example this vision is completely shared by Reiter's variant of Situation Calculus [30]. See in particular [32], where possible worlds are explicitly considered, and actions act on such worlds exactly as said above.[4]

3 Description Logic ontologies

In this paper we focus on ontologies expressed as Description Logics knowledge bases. Description Logics (DLs) [3] are knowledge representation formalisms that are tailored for representing the domain of interest in terms of *constants* (individuals), *concepts* (or

[3]Or, with TBoxes assumed to be acyclic. Notice that such TBoxes can only be used to introduce abbreviations for complex combinations of primitive concepts and roles.

[4]Actually [32] studies also "knowledge producing actions" (i.e., sensing actions), which are more related to belief revision than update.

classes), which denote sets of objects, and *roles* (or relations), which denote denote binary relations between objects. DLs *knowledge bases* (KBs) are formed by two distinct parts: the so-called *TBox*, which contains intensional description of the domain of interest; and the so-called *ABox*, which contains extensional information.

When DLs are used to express ontologies [4], the TBox is used to express the *intensional level of the ontology*, i.e., the shared conceptualization of the domain of interest, while the ABox is used to represent the *instance level of the ontology*, i.e., the information on actual objects that are instances of the concepts and roles defined at the intensional level. From a formal point of view, a DL KB is based on an alphabet of atomic concepts, atomic roles, and constants, and is specified in terms of a pair $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where:

- $\mathcal{T}$, the *TBox*, is formed by a finite set of *universal assertions*. The precise form of such assertions depends on the specific DL. Generally speaking, the TBox is formed by inclusions that allow imposing constraints on the extensions of the concepts and roles used to describe the domain of interest.
- $\mathcal{A}$, the *ABox*, is formed by a finite set of *membership assertions* stating that a given object (or pair of objects) is an instance of a concept (or a role). Note that DLs adopt the open world assumption (and not the closed world assumption, typical of databases), i.e., it may happen that, for an object a and concept C, $\mathcal{K}$ does not imply neither that a is an instance of C, nor that a is not an instance of C; similarly for roles.

We give the semantics of a DL KB in terms of interpretations over a fixed countably infinite domain Δ of objects. We assume that, for each object in Δ, we have exactly one constant in our alphabet denoting such object. In this way we blur the distinction between constants and objects, so that we can use them interchangeably (with a little abuse of notation) without causing confusion.[5] An interpretation $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ consists of a first order structure over Δ, where $\cdot^{\mathcal{I}}$ is the interpretation function, i.e., a function mapping each concept to a subset of Δ and each role to a subset of $\Delta \times \Delta$. We say that $\mathcal{I}$ is a *model of a (TBox or ABox) assertion* α, or also that $\mathcal{I}$ *satisfies a (TBox or ABox) assertion* α, if α is true in $\mathcal{I}$[6]. We say that $\mathcal{I}$ is a *model of the KB* $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, or also that $\mathcal{I}$ satisfies $\mathcal{K}$, if $\mathcal{I}$ is a model of all the assertions in $\mathcal{T}$ and $\mathcal{A}$.

Given a set of (TBox or ABox) assertions $\mathcal{S}$, we denote by $Mod(\mathcal{S})$ the set of interpretations that are models of all assertions in $\mathcal{S}$. In particular, the *set of models of a KB* $\mathcal{K}$, denoted as $Mod(\mathcal{K})$, is the set of models of all assertions in $\mathcal{T}$ and $\mathcal{A}$, i.e. $Mod(\mathcal{K}) = Mod(\langle \mathcal{T}, \mathcal{A} \rangle) = Mod(\mathcal{T}) \cap Mod(\mathcal{A})$.

A KB $\mathcal{K}$ is *satisfiable* if $Mod(\mathcal{K}) \neq \emptyset$, i.e. it has at least one model. We say that a set $\mathcal{F}$ of assertions is *consistent* with $\mathcal{K}$ if $Mod(\mathcal{K} \cup \mathcal{F}) \neq \emptyset$, and we say that $\mathcal{K}$ *logically implies* an assertion α, written $\mathcal{K} \models \alpha$, if $Mod(\mathcal{K}) \subseteq Mod(\alpha)$. On the contrary, we say that a KB $\mathcal{K}$ *does not logically imply* an assertion α, written $\mathcal{K} \not\models \alpha$, if there exists at least one model of $\mathcal{K}$ that is not a model of α.

[5] We use such a shared domain of interpretation, i.e., we use the so-called standard names [25], to simplify comparison between models needed for updates. In fact, the use of standard names could be avoided, but this would make some of the definitions below clumsier.

[6] Obviously, the exact meaning of an assertion α being true in an interpretation $\mathcal{I}$ depends on the form of such assertion, and therefore on the particular DL that we are using for expressing the ontology.

The DL *DL-Lite*$_{\mathcal{F}}$ In this paper, we focus on one of the most interesting families of DLs that have come up in the last years: the *DL-Lite* family [10, 9, 8]. This is a family of DLs that despite their simplicity are able to capture the main notions (though not all, obviously) of both ontologies, and of conceptual modeling formalisms used in databases and software engineering, such as Entity-Relationship diagrams and UML class diagrams. At the same time they allow for querying the KB through arbitrary (non recursive) positive queries, or *unions of conjunctive queries*. These queries make use of explicit variables, and may express complex patterns on the instances of the KB, that go well beyond the class of queries that is typically considered in DLs. Formally, the fundamental characteristic of the DLs in *DL-Lite* family is that reasoning, including answering unions of conjunctive queries, is polynomially tractable in the size of the KB and in fact first-order reducible (and hence in LOGSPACE) with respect to data complexity [10], i.e., the complexity measured with respect to the number of individuals in the ABox. These features make DLs in the *DL-Lite* family particularly suitable as a sort of conceptual layer for data intensive applications.

Here, we concentrate on the DL called *DL-Lite*$_{\mathcal{F}}$, originally proposed in [8]. The *DL-Lite*$_{\mathcal{F}}$ concept expressions are:

$$\begin{array}{lcl} B & ::= & A \mid \exists R \\ C & ::= & B \mid \neg B \\ R & ::= & P \mid P^- \end{array}$$

where A denotes an *atomic concept*, P an *atomic role*, B a *basic concept*, and C a *general concept*. A basic concept can be either an atomic concept, a concept of the form $\exists P$, i.e. the standard DL construct of unqualified existential quantification on roles, or a concept of the form $\exists P^-$, which involves *inverse roles* (P^- denotes the inverse of the role P).

The TBox assertions allowed in *DL-Lite*$_{\mathcal{F}}$ are of the following forms:

$B_1 \sqsubseteq B_2$	inclusion assertion
$B_1 \sqsubseteq \neg B_2$	disjointness assertion
$(\mathsf{funct}\ R)$	functionality assertion

An *inclusion assertion* specifies that each instance of the basic concept B_1 is also an instance of the basic concept B_2, i.e., B_1 is subsumed by B_2. A *disjointness assertion* specifies that each instance of a basic concept B_1 is not an instance of the basic concept B_2, i.e., B_1 and B_2 are disjoint. Finally, a *functionality assertion* expresses the (global) functionality of an atomic role, or of the inverse of an atomic role. Note that negation is used in a restricted way, in particular for asserting disjointness of concepts. Moreover, disjunction is disallowed. Notably, if we remove any of these two limitations, reasoning becomes intractable, see [9].

The ABox assertions allowed in *DL-Lite*$_{\mathcal{F}}$ are of the following forms:

$$B(a), \quad R(a, b) \quad \text{membership assertions}$$

where a, b are constants, B is any basic concept, and R is either an atomic role or its inverse.

Concerning the semantics of concepts and roles, given an interpretation $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ the interpretation function $\cdot^{\mathcal{I}}$ interprets the constructs of *DL-Lite*$_{\mathcal{F}}$ as follows:

$$
\begin{array}{l}
A^{\mathcal{I}} \subseteq \Delta \\
P^{\mathcal{I}} \subseteq \Delta \times \Delta \\
(P^{-})^{\mathcal{I}} = \{(d', d) \mid (d, d') \in P^{\mathcal{I}}\} \\
(\exists R)^{\mathcal{I}} = \{d \mid \exists d'.(d, d') \in R^{\mathcal{I}}\} \\
(\neg B)^{\mathcal{I}} = \Delta \setminus B^{\mathcal{I}}
\end{array}
$$

Finally, we specify the conditions for an interpretation $\mathcal{I}$ to be a model of a TBox and an ABox assertion. In particular, $\mathcal{I}$ satisfies

- $B_1 \sqsubseteq B_2$, if $B_1^{\mathcal{I}} \subseteq B_2^{\mathcal{I}}$,
- $B_1 \sqsubseteq \neg B_2$, if $B_1^{\mathcal{I}} \subseteq \neg B_2^{\mathcal{I}}$,
- (funct R), if $(d, d') \in R^{\mathcal{I}}$ and $(d, d'') \in R^{\mathcal{I}}$ implies $d' = d''$,
- $B(a)$, if $a \in B^{\mathcal{I}}$,
- $R(a, b)$, if $(a, b) \in R^{\mathcal{I}}$.

4 Instance-level update

In instance-based update, we allow the user to state new membership assertions in order to revise the ABox, while still maintaining unchanged the intensional level of the ontology, i.e. the TBox.

As we said in the introduction, to assign formal semantics to update, we essentially follow Winslett's approach [34, 35] . Technically, the idea underlying Winslett's approach to knowledge base update, is the following. A knowledge base update is specified through a set of assertions, and produces a set of models that

- satisfy the assertions, and
- is obtained by updating each model of the initial knowledge base with minimal change. In particular, according to Winslett, the notion of minimal change builds upon both the symmetric difference, and an inclusion-based measure of closeness. The latter means that a set S' is *closer* to a set S than a set S'' if the elements on which S and S' differ are a proper subset of the elements on which S and S'' differ.

We start by adapting Winslett's notion of closeness to our context. To this aim, we need first to provide some definitions. Specifically, we need to define both the notion of difference and the notion of inclusion between interpretations.

Definition 4.1 (**Difference between interpretations**) Given two interpretations $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$ and $\mathcal{I}' = (\Delta, \cdot^{\mathcal{I}'})$ for KB $\mathcal{K}$, we define the *difference between* $\mathcal{I}$ *and* $\mathcal{I}'$, written $\mathcal{I} \ominus \mathcal{I}'$, as the interpretation $(\Delta, \cdot^{\mathcal{I} \ominus \mathcal{I}'})$ such that:

- $C^{\mathcal{I} \ominus \mathcal{I}'} = C^{\mathcal{I}} \ominus C^{\mathcal{I}'}$, for every atomic concept C in $\mathcal{K}$;
- $P^{\mathcal{I} \ominus \mathcal{I}'} = P^{\mathcal{I}} \ominus P^{\mathcal{P}'}$, for every atomic role P in $\mathcal{K}$;

where, for sets S and S', $S \ominus S'$ denotes the usual symmetric difference between S and S', i.e. $S \ominus S' = (S \cup S') \setminus (S \cap S')$.

Definition 4.2 (**Inclusion between interpretations**) Given two interpretations $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$ and $\mathcal{I}' = (\Delta, \cdot^{\mathcal{I}'})$ for a KB $\mathcal{K}$, we say that $\mathcal{I} \subseteq \mathcal{I}'$ iff $\mathcal{I}, \mathcal{I}'$ are such that:

- $C^{\mathcal{I}} \subseteq C^{\mathcal{I}'}$, for every concept C in $\mathcal{K}$;
- $R^{\mathcal{I}} \subseteq R^{\mathcal{I}''}$, for every role R in $\mathcal{K}$.

Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a DL KB, and suppose we want to perform an instance-level update of $\mathcal{K}$ with a finite set $\mathcal{F}$ of membership assertions. Since we want $\mathcal{T}$ to remain unchanged, the first observation is that $\mathcal{F}$ should be consistent with $\mathcal{T}$, while it may be inconsistent with $\mathcal{A}$. Following Winslett, we denote by $Update^{\mathcal{T}}(\mathcal{F}, \mathcal{A})$ the set of models obtained by updating each model $\mathcal{I}$ of $\mathcal{K}$ so as to apply *minimal changes* to the model, and to make all membership assertions in $\mathcal{F}$ true. Specifically, Winslett's model change operator is defined in our context as follows.

Definition 4.3 (**Model update**) Let $\mathcal{T}$ be a TBox, $\mathcal{I}$ a model of $\mathcal{T}$, and $\mathcal{F}$ a finite set of membership assertions such that $Mod(\mathcal{T} \cup \mathcal{F}) \neq \emptyset$. The *update of $\mathcal{I}$ with $\mathcal{F}$*, denoted $U^{\mathcal{T}}(\mathcal{I}, \mathcal{F})$, is defined as follows:

$$U^{\mathcal{T}}(\mathcal{I}, \mathcal{F}) = \{\mathcal{I}' \mid \mathcal{I}' \in Mod(\mathcal{T} \cup \mathcal{F}) \text{ and there exists no } \mathcal{I}'' \in Mod(\mathcal{T} \cup \mathcal{F}) \text{ such that } \mathcal{I} \ominus \mathcal{I}'' \subset \mathcal{I} \ominus \mathcal{I}'\}$$

Observe that $U^{\mathcal{T}}(\mathcal{I}, \mathcal{F})$ is the set of models of both $\mathcal{T}$ and $\mathcal{F}$ whose difference with respect to $\mathcal{I}$ is $\subseteq$-minimal, so as to capture the notion of *minimal change*.

With these notions in place we are now ready to provide the formal definition of instance-level update in DL ontologies.

Definition 4.4 (**Instance-level update**) Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a DL KB, and $\mathcal{F}$ a finite set of membership assertions such that $Mod(\mathcal{T} \cup \mathcal{F}) \neq \emptyset$. The *instance-level update of $\mathcal{K}$ with $\mathcal{F}$*, or simply the update of $\mathcal{K}$ with $\mathcal{F}$, denoted $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$, is defined as follows:

$$\mathcal{K} \circ_{\mathcal{T}} \mathcal{F} = \bigcup_{\mathcal{I} \in Mod(\mathcal{K})} U^{\mathcal{T}}(\mathcal{I}, \mathcal{F}).$$

Let us illustrate this notion by means of an example.

Example 4.5 Consider an ontology describing a basketball players' domain, where the intensional level of the ontology is expressed by means of the following *DL-Lite*$_{\mathcal{F}}$ TBox $\mathcal{T}$:

$$\exists WillPlay \sqsubseteq AvailablePlayer$$
$$AvailablePlayer \sqsubseteq Player$$
$$Injured \sqsubseteq \neg AvailablePlayer$$

The TBox $\mathcal{T}$ states that someone who will play in a game is an available player, an available player is a player, and someone who is injured cannot be an available player. Consider the instance level of the ontology expressed by means of the *DL-Lite*$_\mathcal{F}$ ABox $\mathcal{A}$:

$$WillPlay(john, allstargame)$$

Now suppose that $john$ gets injured: we update the ontology with the assertion $Injured(john)$, i.e. $\mathcal{F} = \{Injured(john)\}$. It is not difficult to see that, based on the semantics presented above, $\langle \mathcal{T}, \mathcal{A} \rangle \circ_\mathcal{T} \mathcal{F}$ is the set of models

$$\{\mathcal{I}' \mid \mathcal{I}' \in Mod(\mathcal{T}) \wedge john \in Injured^{\mathcal{I}'} \wedge john \in Player^{\mathcal{I}'}\}$$

Intuitively, in each model of the update, $john$ is a player who is actually injured. Note in particular that the fact that $john$ is injured implies that he is not an available player, and therefore he will not play the $allstargame$ anymore. However, since our notion of update maintains as much as possible the information that is logically implied by the initial KB and that is not inconsistent with the new membership assertions, $john$ is still a player. ■

One may wonder whether there exists a knowledge base $\langle \mathcal{T}, \mathcal{A}' \rangle$ in *DL-Lite*$_\mathcal{F}$ whose set of models is exactly the set $\langle \mathcal{T}, \mathcal{A} \rangle \circ_\mathcal{T} \mathcal{F}$ described in the example above. It is not hard to see that such a knowledge base exists and is such that $\mathcal{A}'$ is expressed as follows:

$$Injured(john)$$
$$Player(john)$$

Hence, the above example shows a case where the result of an instance-level update can be represented as an ABox in the same language as the original KB. However, as we said in the previous sections, this is not always the case. Thus, it makes sense to introduce the formal notion of expressibility of updates.

Definition 4.6 **(Expressibility)** Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a KB expressed in a DL $\mathcal{L}$, and $\mathcal{F}$ a finite set of membership assertions expressed in $\mathcal{L}$ such that $Mod(\mathcal{T} \cup \mathcal{F}) \neq \emptyset$. We say that the result of the instance-level update of $\mathcal{K}$ with $\mathcal{F}$ is expressible in $\mathcal{L}$ if there exists an ABox $\mathcal{A}'$ such that

$$\mathcal{K} \circ_\mathcal{T} \mathcal{F} = Mod(\langle \mathcal{T}, \mathcal{A}' \rangle).$$

4.1 Instance-level update in *DL-Lite*$_\mathcal{F}$

In [26], it is shown that, for expressive DLs, the result of an instance-level update *cannot* be expressed in the language of the original KB. Unfortunately, this is true for *DL-Lite*$_\mathcal{F}$ as well, as shown next.

Theorem 4.7 *There exist DL-Lite*$_\mathcal{F}$ *KBs* $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ *and sets of membership assertions* $\mathcal{F}$*, with* $Mod(\mathcal{T} \cup \mathcal{F}) \neq \emptyset$*, such that for no DL-Lite*$_\mathcal{F}$ *ABox* $\mathcal{A}'$ *we have* $Mod(\langle \mathcal{T}, \mathcal{A}' \rangle) = \mathcal{K} \circ_\mathcal{T} \mathcal{F}$*.*

Proof. It suffices to consider the following *DL-Lite*$_\mathcal{F}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$

$$\begin{array}{rcl} \mathcal{T} & = & \{\exists P^- \sqsubseteq A_1,\ A_2 \sqsubseteq \neg\exists P\} \\ \mathcal{A} & = & \{\exists P(a)\}. \end{array}$$

and its update with

$$\mathcal{F} = \{A_2(a)\}.$$

We show that no *DL-Lite*$_\mathcal{F}$ ABox $\mathcal{A}'$ exists such that the KB $\langle \mathcal{T}, \mathcal{A}' \rangle$ captures exactly the set of models $\mathcal{K} \circ_\mathcal{T} \mathcal{F}$. Clearly, given $\mathcal{I} \in Mod(\mathcal{K})$, an interpretation $\mathcal{I}' \in \mathcal{K} \circ_\mathcal{T} \mathcal{F}$ that satisfies both $A_2(a)$ and $\mathcal{T}$, and minimally differs from $\mathcal{I}$, is obtained by simply modifying the interpretation of P so that $a \in (\neg\exists P)^{\mathcal{I}'}$, that is for no $x \in \Delta$, we have that $(a, x) \in P^{\mathcal{I}'}$. In particular, this means that $A_1^{\mathcal{I}'} = \mathcal{A}_1^\mathcal{I}$. On the other hand, by $\exists P^- \sqsubseteq A_1$, for each x such that $(a, x) \in P^\mathcal{I}$, we have that $x \in A_1^\mathcal{I}$. Now, since $\exists P(a)$ holds in every model of $\mathcal{K}$, we have that every model $\mathcal{I}$ of $\mathcal{K}$ has at least one object $z \in \Delta$ such that $z \in A_1^\mathcal{I}$, and hence, every model $\mathcal{I}' \in \mathcal{K} \circ_\mathcal{T} \mathcal{F}$ is such that there exists at least one such object z that belongs to $A_1^{\mathcal{I}'}$. Unfortunately, since such object z may differ in different models, this fact *cannot* be captured with a *DL-Lite*$_\mathcal{F}$ membership assertion. □

One may wonder whether we can enhance the expressive power of *DL-Lite*$_\mathcal{F}$ in order to come up with a new DL where instance-level update is expressible. In the next subsection, we answer this question by presenting a minimal extension of *DL-Lite*$_\mathcal{F}$, called *DL-Lite*$_{\mathcal{FS}}$, where updates are indeed expressible.

4.2 Instance-level update in *DL-Lite*$_{\mathcal{FS}}$

DL-Lite$_{\mathcal{FS}}$ is a slight variant of *DL-Lite*$_\mathcal{F}$ that allows for more expressive membership assertions in the ABox. In particular, membership assertions in *DL-Lite*$_{\mathcal{FS}}$ may involve also *variables*, also called *soft constants*, i.e., a particular kind of terms that may denote different objects in different models. More precisely, *DL-Lite*$_{\mathcal{FS}}$ membership assertions are of the form:

$$C(a), \quad R(a, b), \quad C(z)$$

where a, b are constants, z is a variable, C is a general *DL-Lite*$_\mathcal{F}$ concept (i.e. a basic *DL-Lite*$_\mathcal{F}$ concept or its negation), and R is an atomic role or its inverse.

Intuitively, the assertions $C(a)$ and $R(a, b)$ state, respectively, that the object a, resp. the pair (a, b), is an instance of the concept C, resp. role R. We call these assertions *ground*, to emphasize the absence of variables. A (non-ground) assertion $C(z)$ instead states that the there exists an object, denoted by the variable z, that is an instance of the concept C. In other words, variables are used to express the existence of objects that are instance of concepts, without actually naming the objects.

With respect to the semantics, in order to interpret a set of extensional assertions possibly involving variables, we need to introduce the notion of *assignment*. Let $\mathcal{V}$ be the set of variables occurring in an ABox $\mathcal{A}$, we call *assignment for* $\mathcal{A}$ a function μ from $\mathcal{V}$ to Δ.

Let $\mathcal{I}$ be an interpretation, and μ an assignment for $\mathcal{A}$. We say that $\mathcal{I}$ is *model of* $\mathcal{A}$ *with* μ, or equivalently, $\mathcal{I}$ satisfies $\mathcal{A}$ with μ, if for each membership assertion in $\mathcal{A}$ of the form:

- $C(a)$, we have that $a \in C^{\mathcal{I}}$;
- $R(a, b)$, we have that $(a, b) \in R^{\mathcal{I}}$;
- $C(z)$, we have that $\mu(z) \in C^{\mathcal{I}}$.

We say that $\mathcal{I}$ is a *model of* $\mathcal{A}$ if there exists an assignment μ such that $\mathcal{I}$ is a model of $\mathcal{A}$ with μ.

Note that, as shown in [28], *DL-Lite*$_{\mathcal{FS}}$ retains all the nice computational properties of *DL-Lite*$_{\mathcal{F}}$. Moreover, we show here that this logic is closed under instance-level update. In particular, in what follows:

- we show that the result of an instance-level update is always expressible within *DL-Lite*$_{\mathcal{FS}}$: i.e., for any KB $\mathcal{K}$ in *DL-Lite*$_{\mathcal{FS}}$, and for any update, there always exists a *DL-Lite*$_{\mathcal{FS}}$ ABox that reflects the changes of the update to the original KB $\mathcal{K}$ (obviously the TBox remains unchanged as required);
- we show that the ABox resulting from an update can be effectively computed;
- finally, we show that the size of such an ABox is polynomially bounded by the size of the original KB, and moreover, that it can be computed in polynomial time.

In Figure 1, we provide an algorithm for instance-level update over *DL-Lite*$_{\mathcal{FS}}$ KBs. To simplify the presentation we make use of the following notation.

- We denote by R^- the inverse of R, i.e., if R is an atomic role, then R^- is its inverse, while if R is the inverse of an atomic role, then R is the atomic role itself.
- We write $\neg C$ to denote $\neg B$ if C is B, and B if C is $\neg B$.
- We use the notation $C_1 \sqsubseteq C_2$ to denote either assertions of the form $B_1 \sqsubseteq B_2$, $B_1 \sqsubseteq \neg B_2$, or $\neg B_1 \sqsubseteq \neg B_2$.
- We denote by $cl(\mathcal{T})$ the deductive closure of $\mathcal{T}$ i.e., $\{C_1 \sqsubseteq C_2 \mid \langle \mathcal{T}, \mathcal{A} \rangle \models C_1 \sqsubseteq C_2\}$. It can be shown that in *DL-Lite*$_{\mathcal{FS}}$, $cl(T)$ can be computed in polynomial time, based on the following inductive definition: *(i)* $\mathcal{T} \subseteq cl(\mathcal{T})$; *(ii)* if $C_1 \sqsubseteq C_2$ is in $cl(\mathcal{T})$ then $\neg C_2 \sqsubseteq \neg C_1$ is in $cl(\mathcal{T})$; *(iii)* if $C_1 \sqsubseteq C_2$ and $C_2 \sqsubseteq C_3$ are in $cl(\mathcal{T})$, then $C_1 \sqsubseteq C_3$ is in $cl(\mathcal{T})$.

The algorithm in Fig. 1 takes as input a satisfiable *DL-Lite*$_{\mathcal{FS}}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, and a finite set of ground (i.e., not involving variables) membership assertions $\mathcal{F}$, and returns an ABox $\mathcal{A}'$, if $\langle \mathcal{T}, \mathcal{F} \rangle$ is satisfiable, ERROR, otherwise. More precisely, the algorithm starts by checking the satisfiability of the knowledge base $\langle \mathcal{T}, \mathcal{F} \rangle$ (line 1). Then, if $\mathcal{K}$ is satisfiable, it proceeds by three main steps.

ALGORITHM $ComputeUpdate(\mathcal{T}, \mathcal{A}, \mathcal{F})$
INPUT: finite set of ground membership assertions $\mathcal{F}$,
satisfiable *DL-Lite*$_{\mathcal{FS}}$ KB $\langle \mathcal{T}, \mathcal{A} \rangle$
OUTPUT: an ABox $\mathcal{A}'$, or ERROR
[1] if $\langle \mathcal{T}, \mathcal{F} \rangle$ is not satisfiable **then** ERROR
[2] else
[3] { **for each** $F \in \mathcal{F}$ **do**
[4] **if** $F = R(a, b)$ **then** $\mathcal{F} := \mathcal{F} \cup \{\exists R(a), \exists R^-(b)\}$
[5] $\mathcal{F}' := \emptyset$
[6] **for each** $F \in \mathcal{F}$ **do**
[7] **if** $F = C(a)$ **then**
[8] { $\mathcal{F}'' := Saturate(\neg C(a), \mathcal{T})$
[9] **for each** $F'' \in \mathcal{F}''$ **do**
[10] { **if** $\langle \mathcal{T}, \mathcal{A} \rangle \models F''$ **then** $\mathcal{F}' := \mathcal{F}' \cup F''$
[11] **if** $F'' = \exists R(a)$ **then** $\mathcal{F}' := \mathcal{F}' \cup \{R(a, b) \mid R(a, b) \in \mathcal{A}\}$
[12] }
[13] }
[14] **else**
[15] **if** $F = R(a, b)$ **then**
[16] {**if** (funct R) in $\mathcal{T}$ **then**
[17] **for each** $b' \neq b$ s.t. $R(a, b') \in \mathcal{A}$ **do** $\mathcal{F}' := \mathcal{F}' \cup \{R(a, b')\}$
[18] **if** (funct R^-) in $\mathcal{T}$ **then**
[19] **for each** $a' \neq a$ s.t. $R(a', b) \in \mathcal{A}$ **do** $\mathcal{F}' := \mathcal{F}' \cup \{R(a', b)\}$
[20] }
[21] $\mathcal{A}' := \mathcal{A} \cup \mathcal{F}$;
[22] **for each** $F' \in \mathcal{F}'$ **do**
[23] **if** $F' = C(a)$ **then**
[24] $\{\mathcal{A}' := \mathcal{A}' \setminus \{C(a)\}$
[25] **for each** $C \sqsubseteq C_1$ in $cl(\mathcal{T})$ **do**
[26] **if** $(C_1(a) \notin \mathcal{F}')$ **then** $\mathcal{A}' := \mathcal{A}' \cup \{C_1(a)\}$
[27] **if** $F' = \exists R(a)$ **then**
[28] **for each** $\exists R^- \sqsubseteq C_2$ in $cl(\mathcal{T})$ **do**
[29] $\mathcal{A}' := \mathcal{A}' \cup \{C_2(z_{\exists R(a)})\}$, with $z_{\exists R(a)}$ new variable
[30] }
[31] **else**
[32] **if** $F' = R(a, b)$ **then**
[33] $\{\mathcal{A}' := \mathcal{A}' \setminus \{R(a, b), \exists R(a), \exists R^-(b)\}$
[34] **for each** $\exists R \sqsubseteq C_3$ in $cl(\mathcal{T})$ **do**
[35] **if** $C_3(a) \notin \mathcal{F}'$ **then** $\mathcal{A}' := \mathcal{A}' \cup \{C_3(a)\}$
[36] **for each** $\exists R^- \sqsubseteq C_4$ in $cl(\mathcal{T})$ **do**
[37] **if** $C_4(b) \notin \mathcal{F}'$ **then** $\mathcal{A}' := \mathcal{A}' \cup \{C_4(b)\}$
[38] }
[39] }

Figure 1: Algorithm $ComputeUpdate$

- First, it adds to $\mathcal{F}$ all membership assertions that are logically implied by $\mathcal{F}$ itself. Specifically, this amounts to possibly insert, for each $R(a, b)$ in $\mathcal{F}$, the membership assertions $\exists R(a)$ and $\exists R^-(b)$, trivially implied by $R(a, b)$ (lines 3–4).

- Second, it computes the set $\mathcal{F}'$ of membership assertions that contradict $\mathcal{F}$ and $\mathcal{T}$, and are logically implied by $\mathcal{K}$ (lines 5–20). To this aim, it considers one by one, each assertion F in $\mathcal{F}$, and proceeds differently depending on whether F involves a concept or a role. Specifically, in case F is a concept membership assertion, it computes the concept assertions that might contradict F, by violating some inclusion assertion of $\mathcal{T}$. This is achieved by calling the function $Saturate(\neg F_1, \mathcal{T})$ shown in Fig. 2, that returns the set of concept membership assertions $\mathcal{F}''$ that, according to $\mathcal{T}$, logically imply F (line 8). Then, for each assertion F'' in $\mathcal{F}''$, if F'' is actually logically implied by $\mathcal{K}$, it inserts F'' into $\mathcal{F}'$ (line 10). Furthermore, in case F'' has the form $\exists R(a)$, it inserts into $\mathcal{F}'$ also those membership assertions $R(a, b)$ that belong to $\mathcal{A}$, since they trivially imply F'' and hence also contradict F (line 11). On the other hand, in case F is a role membership assertion of the form $R(a, b)$, this step amounts to check whether F is functional, and to insert into $\mathcal{F}'$ those role membership assertions in $\mathcal{A}$ that violate the functionality (lines 15–20).

- Finally, it computes the result set $\mathcal{A}'$. More precisely, starting from all membership assertions in $\mathcal{A}$ and $\mathcal{F}$ (line 21), it deletes each assertion F' that belongs to $\mathcal{F}'$, and simultaneously, it inserts those membership assertions that are logically implied by F' and do not contradict $\mathcal{F}$ (lines 22–38). Note that, here again, it proceeds differently depending on whether F' involves a concept (lines 23–30), or a role (lines 32–38). Specifically, in the former case, besides deleting F' from $\mathcal{A}'$ and inserting all assertions, not in $\mathcal{F}'$, that are implied by F' according to some inclusion assertion in $\mathcal{T}$, it possibly inserts a membership assertion $C_2(z)$, for some new variable z denoting an (unnamed) object of the domain. This is motivated by the fact that, the assertion $\exists R(a)$ logically implies $\exists R(z)$, for some object denoted by z. Hence, some inclusion assertion of the form $\exists R^- \sqsubseteq C_2$, would imply $C_2(z)$ (lines 28–29). Dually, in case F' involves a role, i.e. it has the form $R(a, b)$, the algorithm deletes from $\mathcal{A}'$ also the assertions $\exists R(a)$ and $\exists R^-(b)$, that are trivially implied by F' (line 33), and possibly inserts into $\mathcal{A}'$ the assertions, not in $\mathcal{F}'$, that are implied by $\exists R(a)$ and $\exists R^-(b)$ according to some inclusion assertion in $\mathcal{T}$ (lines 34–37).

Next, we show soundness and completeness of the algorithm $ComputeUpdate$.

Theorem 4.8 *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a DL-Lite$_{\mathcal{FS}}$ KB, $\mathcal{F}$ a finite set of ground DL-Lite$_{\mathcal{FS}}$ membership assertions such that $Mod(\mathcal{T}) \cap Mod(\mathcal{F}) \neq \emptyset$, and $\mathcal{K}'$ the DL-Lite$_{\mathcal{FS}}$ KB such that $\mathcal{K}' = \langle \mathcal{T}, \mathcal{A}' \rangle$, where $\mathcal{A}' = ComputeUpdate(\mathcal{T}, \mathcal{A}, \mathcal{F})$. Then*

$$\mathcal{K} \circ_{\mathcal{T}} \mathcal{F} = Mod(\mathcal{K}').$$

Proof. We first prove that

$$Mod(\mathcal{K}') \subseteq \mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$$

ALGORITHM $Saturate(C(a), \mathcal{T})$
INPUT: membership assertion $C(a)$, TBox $\mathcal{T}$
OUTPUT: set $\mathcal{P}$ of ground membership assertions
$\mathcal{P} := \{C(a)\}$
repeat
  $\mathcal{P}' = \mathcal{P}$
  for each $C(a) \in \mathcal{P}'$
    if $C' \sqsubseteq C \in cl(\mathcal{T})$ **then** $\mathcal{P} := \mathcal{P} \cup \{C'(a)\}$
until $\mathcal{P} = \mathcal{P}'$

Figure 2: Algorithm $Saturate$

by showing that for each model $\mathcal{I}' \in Mod(\mathcal{K}')$ there exists a model $\mathcal{I} \in Mod(\mathcal{K})$ such that $\mathcal{I}' \in U^{\mathcal{T}}(\mathcal{I}, \mathcal{F})$.

In fact we build the model $\mathcal{I}$ starting from $\mathcal{I}'$, and reintroducing facts $\mathcal{F}'$ removed in computing $\mathcal{A}'$. First, we observe that, if $\mathcal{I}'$ is a model of $\mathcal{K}'$ then there exists an assignment of the variables in $\mathcal{A}'$, let μ be such assignment. Then, in order to build $\mathcal{I}$, we start by setting $\mathcal{I} = \mathcal{I}'$, and we proceed as follows:

- for each $A(a) \in \mathcal{F}'$, we include a in $A^{\mathcal{I}}$;
- for each $\neg A(a) \in \mathcal{F}'$, we remove a from $A^{\mathcal{I}}$;
- for each $R(a, b) \in \mathcal{F}'$, we include (a, b) in $R^{\mathcal{I}}$, and moreover
  - if (funct R) $\in \mathcal{T}$, we remove all $(a, x) \in R^{\mathcal{I}}$, with $b \neq x$, and
  - if (funct R^-) $\in \mathcal{T}$ we remove all $(x, b) \in R^{\mathcal{I}}$, with $a \neq x$;
- for each $\exists R(a) \in \mathcal{F}'$ we include $(a, \mu(z_{\exists R(a)}))$ in $R^{\mathcal{I}}$, and moreover
  - if (funct R) $\in \mathcal{T}$ we remove all $(a, x) \in R^{\mathcal{I}}$, with $\mu(z_{\exists R(a)}) \neq x$, and
  - if (funct R^-) $\in \mathcal{T}$ we remove all $(x, \mu(z_{\exists R(a)})) \in R^{\mathcal{I}}$, with $a \neq x$;
- for each $\neg\exists R(a) \in \mathcal{F}'$ we remove all (a, x) from $R^{\mathcal{I}}$.

It is easy to verify that $\mathcal{I}$ defined as above is a model of $\mathcal{K}$. Now, in order to complete the proof, we need to show that $\mathcal{I}' \in U^{\mathcal{T}}(\mathcal{I}, \mathcal{F})$, and since, by inspecting the algorithm, clearly $\mathcal{I}'$ is a model of both $\mathcal{T}$ and $\mathcal{F}$, this amounts in showing that exists no interpretation $\mathcal{I}''$ of $\mathcal{T}$ and $\mathcal{F}$ such that:

- $\mathcal{I}'' \in Mod(\mathcal{T} \cup \mathcal{F})$, and
- $\mathcal{I} \neq \mathcal{I}'' \subset \mathcal{I} \ominus \mathcal{I}'$.

We proceed by contradiction, case by case. Suppose that there exists an individual $x \in \Delta$ such that $x \in A^{\mathcal{I}}$ and $x \notin A^{\mathcal{I}'}$ but $x \in A^{\mathcal{I}''}$. Then x was introduced in $A^{\mathcal{I}}$ because $A(x) \in \mathcal{F}'$. But then $\neg A(x)$ was a logical consequence of $\mathcal{F}$ and $\mathcal{T}$, and hence, x cannot be in $A^{\mathcal{I}''}$ since $\mathcal{I}''$ is a model of both $\mathcal{T}$ and $\mathcal{F}$. Similarly, there cannot exist $x \in \Delta$ such that $x \notin A^{\mathcal{I}}$ and $x \in A^{\mathcal{I}'}$ but $x \notin A^{\mathcal{I}''}$. A similar line of reasoning can be applied also for the cases in which the difference between $\mathcal{I}'$ and $\mathcal{I}''$ involves atomic roles instead of atomic concepts.

For the other direction we show that

$$\mathcal{K} \circ_{\mathcal{T}} \mathcal{F} \subseteq Mod(\mathcal{K}').$$

We proceed by assuming by contradiction that there exists an interpretation $\bar{\mathcal{I}} \in \mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$ that is not a model of $\mathcal{K}'$. Then $\bar{\mathcal{I}}$ does not satisfy at least a membership assertion α in $\mathcal{A}' \setminus \mathcal{F}$. Note that, by construction, if α belongs to $\mathcal{A}' \setminus \mathcal{F}$, then α has been inserted into $\mathcal{A}'$ either at line 23, 26, 30 or 32 of the Algorithm $ComputeUpdate$ (cf. Fig 1). But these are logical consequences of $\mathcal{A}$ and $\mathcal{T}$ that do not contradict $\mathcal{F}$ so they must remain true in all models in $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$ including $\bar{\mathcal{I}}$. □

Next we turn to termination and computational complexity of $ComputeUpdate$.

Theorem 4.9 *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a DL-Lite$_{\mathcal{FS}}$ KB, and $\mathcal{F}$ a finite set of ground DL-Lite$_{\mathcal{FS}}$ membership assertions. Then $ComputeUpdate(\mathcal{T}, \mathcal{A}, \mathcal{F})$ terminates, returning ERROR if $Mod(\mathcal{T}) \cap Mod(\mathcal{F}) = \emptyset$, and an ABox $\mathcal{A}'$ such that $\langle \mathcal{T}, \mathcal{A}' \rangle$ is a DL-Lite$_{\mathcal{FS}}$ KB, otherwise. Moreover:*

- *the size of $\mathcal{A}'$ is polynomially bounded by the size of $\mathcal{T} \cup \mathcal{A} \cup \mathcal{F}$;*
- *$\mathcal{A}'$ is computed in polynomial time in the size of $\mathcal{T} \cup \mathcal{A} \cup \mathcal{F}$.*

Proof. Termination follows immediately by the fact that $cl(\mathcal{T})$ is finite and in fact polynomially bounded by $\mathcal{T}$, and hence also $Saturate(C(a), \mathcal{T})$ computes a finite set that is again polynomially bounded by $\mathcal{T}$. Now, observing that the number of steps to build the set of potential conflicts $\mathcal{F}'$ (lines 3–18) and the number of steps to build $\mathcal{A}'$ given $\mathcal{F}'$ (line 19–32), are both finite, and in fact polynomial in the size of $\mathcal{T}$, $\mathcal{A}$, $\mathcal{F}$, the result follows. □

We end this subsection on *DL-Lite$_{\mathcal{FS}}$* with an example of instance-level update in this logic.

Example 4.10 Let us consider again the KB mentioned in the proof of theorem 4.7:

$$\begin{array}{lcl} \mathcal{T} & = & \{\exists P^- \sqsubseteq A_1,\ A_2 \sqsubseteq \neg \exists P\} \\ \mathcal{A} & = & \{\exists P(a)\}. \end{array}$$

and consider again the update of $\mathcal{K}$ with

$$\mathcal{F} = \{A_2(a)\}.$$

By applying the algorithm $ComputeUpdate$ we get the ABox

$$\mathcal{A}' = \{A_1(z_{\exists P(a)}),\ A_2(a)\}$$

where $z_{\exists P(a)}$ is a variable. By Theorem 4.8, we get that $\langle \mathcal{T}, \mathcal{A} \rangle \circ_{\mathcal{T}} \mathcal{F} = Mod(\langle \mathcal{T}, \mathcal{A}' \rangle)$.

Observe that $\mathcal{A}'$ is expressed in *DL-Lite$_{\mathcal{FS}}$*, and not in *DL-Lite$_{\mathcal{F}}$*, since it makes use of the variable $z_{\exists P(a)}$. ■

The algorithm $ComputeUpdate$ and the results on *DL-Lite$_{\mathcal{FS}}$*, give us the basis for computing good approximations of updates and erasures in *DL-Lite$_{\mathcal{F}}$*, as shown in the next sections.

5 A notion of approximation in Description Logics

From the results of the previous section, as well as from other similar results in the literature [26], it follows that, in general, the result of an update is not expressible in the same language as the original KB. We will see in Section 7 that this holds for erasure as well. This fundamental problem leads us to study *approximation* (see e.g.,[31]) of update and erasure. The basic idea of approximation in our context is as follows. Suppose we are interested in KBs expressed in a DL $\mathcal{L}$, and consider a KB $\mathcal{K}$ expressed in such logic. Consider now an instance-level update (erasure) of $\mathcal{K}$, and suppose that no KB expressed in $\mathcal{L}$ exists that captures exactly the set of models $\mathcal{M}$ resulting from such an update (erasure). In this situation, we aim at computing the KB in $\mathcal{L}$ that approximates $\mathcal{M}$ "at best".

The goal of this section is to introduce a specific notion of approximation in Description Logics, that will be used in the next sections for devising techniques for approximating updates and erasures in *DL-Lite*$_{\mathcal{F}}$ ontologies.

Our notion of approximation is based on fixing a priori both the language $\mathcal{L}$ and the TBox $\mathcal{T}$.

Definition 5.1 (**Sound** $(\mathcal{L}, \mathcal{T})$**-Approximation**) Let $\mathcal{T}$ be a TBox in a DL $\mathcal{L}$, and $\mathcal{M}$ a set of models such that $\mathcal{M} \subseteq Mod(\mathcal{T})$. We say that a DL KB $\mathcal{K}$ is a *sound* $(\mathcal{L}, \mathcal{T})$*-approximation* of $\mathcal{M}$ in $\mathcal{L}$, if

1. $\mathcal{K}$ is in $\mathcal{L}$,
2. $\mathcal{K}$ is of the form $\langle \mathcal{T}, \mathcal{A} \rangle$, and
3. $\mathcal{M} \subseteq Mod(\mathcal{K})$.

In other words, a sound $(\mathcal{L}, \mathcal{T})$-approximation of a subset $\mathcal{M}$ of the models of a TBox $\mathcal{T}$ expressed in $\mathcal{L}$ is a KB that is still expressed in $\mathcal{L}$, that has the same TBox $\mathcal{T}$, and whose set of models includes all the models in $\mathcal{M}$. Obviously, there might be several $(\mathcal{L}, \mathcal{T})$-approximations of a set $\mathcal{M}$. Intuitively, some of them will be "better" than others, in the sense that they will be closer to $\mathcal{M}$. To capture this intuition, we aim at a method for comparing two $(\mathcal{L}, \mathcal{T})$-approximations $\mathcal{K}_1$ and $\mathcal{K}_2$ of $\mathcal{M}$. An appropriate criterion to be used for this purpose is set containment: $\mathcal{K}_1$ is a better approximation than $\mathcal{K}_2$ precisely if $Mod(\mathcal{K}_1) \subset Mod(\mathcal{K}_2)$. Based on this observation, we can define the notion of "best" $(\mathcal{L}, \mathcal{T})$-approximation.

Definition 5.2 (**Maximal** $(\mathcal{L}, \mathcal{T})$**-Approximation**) Let $\mathcal{T}$ be a TBox in a DL $\mathcal{L}$, and $\mathcal{M}$ a set of models such that $\mathcal{M} \subseteq Mod(\mathcal{T})$. We say that a DL KB $\mathcal{K}$ is a *maximal* $(\mathcal{L}, \mathcal{T})$*-approximation* of $\mathcal{M}$ if

1. $\mathcal{K}$ is a sound $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{M}$, and
2. there exists no KB $\mathcal{K}'$ that is a sound $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{M}$, and is such that $Mod(\mathcal{K}') \subset Mod(\mathcal{K})$.

In other words, the maximal $(\mathcal{L}, \mathcal{T})$-approximation of a subset $\mathcal{M}$ of the models of a TBox $\mathcal{T}$ expressed in $\mathcal{L}$ is a KB that is still expressed in $\mathcal{L}$, that has the same TBox $\mathcal{T}$, whose set of models includes all the models in $\mathcal{M}$, and whose semantics minimally differs from $\mathcal{M}$. When we will use this notion for update and erasure, the set of models $\mathcal{M}$ to be approximated will be exactly the models resulting from the update or the erasure of a KB with TBox $\mathcal{T}$.

Interestingly, when a maximal $(\mathcal{L}, \mathcal{T})$-approximation exists, it is unique up to logical equivalence, as the following theorem shows.

Theorem 5.3 *Let $\mathcal{T}$ be a TBox in a DL $\mathcal{L}$, and $\mathcal{M}$ a set of models such that $\mathcal{M} \subseteq Mod(\mathcal{T})$. If a KB $\mathcal{K}$ exists that is a maximal $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{M}$, then all maximal $(\mathcal{L}, \mathcal{T})$-approximations of $\mathcal{M}$ are equivalent to $\mathcal{K}$.*

Proof. Suppose that $\mathcal{K}' = \langle \mathcal{T}, \mathcal{A}' \rangle$ is a maximal $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{M}$ that is not equivalent to $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$. By definition of maximal $(\mathcal{L}, \mathcal{T})$-approximation, we have that both $\mathcal{K}$ and $\mathcal{K}'$ are sound $(\mathcal{L}, \mathcal{T})$-approximations of $\mathcal{M}$, which implies that both $\mathcal{M} \subseteq Mod(\mathcal{K})$, and $\mathcal{M} \subseteq Mod(\mathcal{K}')$, and therefore $\mathcal{M} \subseteq Mod(\mathcal{K}) \cap Mod(\mathcal{K}')$. Now, let $\mathcal{K}''$ be $\langle \mathcal{T}, \mathcal{A} \cup \mathcal{A}' \rangle$. It is easy to see that $Mod(\mathcal{K}'') = Mod(\mathcal{K}) \cap Mod(\mathcal{K}')$, and therefore $\mathcal{K}''$ is also a sound $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{M}$. But then, since $Mod(\mathcal{K}) \neq Mod(\mathcal{K}')$, we obtain that $Mod(\mathcal{K}'') \subset Mod(\mathcal{K})$, which contradicts the fact that $\mathcal{K}$ is a maximal $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{M}$. □

Based on the above property, we will talk about *the* maximal $(\mathcal{L}, \mathcal{T})$-approximation of a set of models.

One may wonder to what extent the maximal $(\mathcal{L}, \mathcal{T})$-approximation of a set $\mathcal{M}$ of models captures $\mathcal{M}$. The basic property of $(\mathcal{L}, \mathcal{T})$-approximation is that it preserves logical implication of $\mathcal{L}$ assertions. That is, as long as we are interested in in logical implication of $\mathcal{L}$ assertions, there is no difference between $\mathcal{M}$ and its maximal $(\mathcal{L}, \mathcal{T})$-approximation. Indeed, the next theorem states that, in terms of logical implication of $\mathcal{L}$-assertions, $\mathcal{M}$ and its maximal $(\mathcal{L}, \mathcal{T})$-approximation are exactly the same.

Theorem 5.4 *Let $\mathcal{T}$ be a TBox in a DL $\mathcal{L}$, and $\mathcal{M}$ a set of models such that $\mathcal{M} \subseteq Mod(\mathcal{T})$. If $\mathcal{K}$ is a maximal $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{M}$, then for every TBox and ABox assertion α in $\mathcal{L}$ it holds that $\mathcal{M} \models \alpha$ iff $\mathcal{K} \models \alpha$.*

Proof. For TBox assertions, the result is trivial. As for ABox assertions, consider a membership assertion α in $\mathcal{L}$. The if-direction is obvious: since $\mathcal{K}$ is an $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{M}$, $\mathcal{K} \models \alpha$ implies $\mathcal{M} \models \alpha$. As for the only-if direction, suppose that $\mathcal{M} \models \alpha$, but there is a model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I} \not\models \alpha$. This would imply that $\mathcal{K} \cup \alpha$ is a sound $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{M}$ that is not equivalent to $\mathcal{K}$, which contradicts the fact that $\mathcal{K}$ is a maximal $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{M}$. □

Observe however that formulas that go beyond $\mathcal{L}$ assertions are sufficient to separate a set of models $\mathcal{M}$ from its maximal $(\mathcal{L}, \mathcal{T})$-approximation. Consider the alphabet with just one concept A, the empty TBox on this alphabet, and the set $\mathcal{M}$ constituted by the two models $\{A(o_1)\}$, and $\{A(o_2)\}$. It is easy to see that the maximal $(\textit{DL-Lite}_{\mathcal{F}}, \mathcal{T})$-approximation of $\mathcal{M}$ is the KB $\mathcal{K}$ with the empty TBox and the empty

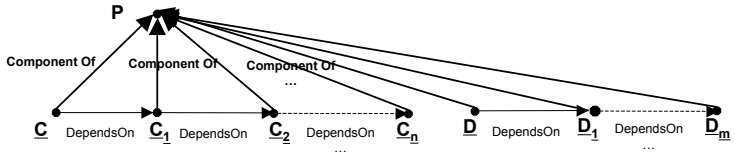

Figure 3: Example of model $\mathcal{I}$ of $\mathcal{K}$

ABox. Indeed, consider any nonempty ABox $\mathcal{A}'$, and let $\alpha(o_i)$ be an assertion in $\mathcal{A}'$. Obviously $\langle \mathcal{T}, \mathcal{A}' \rangle \models \alpha(o_i)$, while $\mathcal{M} \not\models \alpha(o_i)$. This shows that $\langle \mathcal{T}, \mathcal{A}' \rangle$ is not a sound approximation of $\mathcal{M}$. It is also easy to verify that, while $\mathcal{M} \models \exists x.A(x)$, we have that $\mathcal{K} \not\models \exists x.A(x)$.

Finally, we focus on maximal approximations of updates, and show that maximal $(\mathcal{L}, \mathcal{T})$-approximations of updates may not exist.

To this aim, we start by proving a preliminary lemma, in which we make use of the DL $\mathcal{ALCQIO}_{reg}$ (we refer to [3] for the definition of the language $\mathcal{ALCQIO}_{reg}$).

Lemma 5.5 *Let $\mathcal{F}$ be the set $\mathcal{F} = \{Working(c)\}$, and $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be the $\mathcal{ALCQIO}_{reg}$ KB defined as follows:*

$$\begin{array}{rcl} \mathcal{T} & = & \{\top \sqsubseteq \forall ComponentOf.\{p\}, \\ & & \top \sqsubseteq (\leq 1\, DependsOn^-), \\ & & \exists DependsOn^- \sqsubseteq \exists ComponentOf, \\ & & Working \sqsubseteq \forall DependsOn.Working\} \\ \mathcal{A} & = & \{ComponentOf(c,p), \\ & & \neg ComponentOf(p,p), \\ & & \forall ComponentOf^-.\neg Working(p)\} \end{array}$$

Then $\langle \mathcal{T}, \mathcal{A}' \rangle = \mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$, where $\mathcal{A}'$ is the following $\mathcal{ALCQIO}_{reg}$ ABox:

$$\begin{array}{l} \{Working(c), \\ \neg ComponentOf(p,p), \\ \forall ComponentOf^-.(\exists DependsOn^-.\top) \sqcup \{c\} \sqcup \forall (DependsOn)^*.\neg Working(p)\} \end{array}$$

Proof. Consider the initial KB $\mathcal{K}$. Intuitively, the TBox states that each model $\mathcal{I}$ of $\mathcal{K}$ is such that every element of the domain that is a component of an element, is a component of the element $p^{\mathcal{I}}$. Every element of the domain can have at most one element that depends on it. Furthermore, every component that a component depends on, is a component of something, and every element that is working depends only on working elements. On the other hand, the ABox describes the initial state of affairs and states that $\mathcal{I}$ is such that $c^{\mathcal{I}}$ is a component of $p^{\mathcal{I}}$, $p^{\mathcal{I}}$ is not a component of itself, and every element $c'^{\mathcal{I}}$ that is a component of $p^{\mathcal{I}}$ is not working. Note that it follows, in particular, that $c^{\mathcal{I}}$ is not working. Figure 3 shows a graphical representation of a possible model $\mathcal{I}$ of $\mathcal{K}$, where $c^{\mathcal{I}}, c_1^{\mathcal{I}}, \ldots, c_n^{\mathcal{I}}$ are not working.

Consider now the KB $\mathcal{K}' = \langle \mathcal{T}, \mathcal{A}' \rangle$. Intuitively, each model of $\mathcal{K}'$ is such that it satisfies the same intensional assertions satisfied by the models of $\mathcal{K}$. However, the set of models of $\mathcal{K}'$ differs from the set of models of $\mathcal{K}$, in that for each model $\mathcal{I}'$ of $\mathcal{K}'$

- $c^{\mathcal{I}'}$ is working,
- for every element $e^{\mathcal{I}'}$ different from $c^{\mathcal{I}'}$ that is component of $p^{\mathcal{I}'}$, either $e^{\mathcal{I}'}$ depends on some element, or $e^{\mathcal{I}'}$ is such that each element that depends on it, directly or indirectly, is not working.

We next show that $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F} \subseteq Mod(\mathcal{K}')$ is a model of $\mathcal{K}'$. Indeed, consider for example the model $\mathcal{I}$ shown in Figure 3. The set of models $U^{\mathcal{T}}(\mathcal{I}, \mathcal{F})$ comprises all models that are obtained modifying $\mathcal{I}$ either by interpreting as working both c and all the elements $c_1, \ldots, c_n$, or by interpreting as not depending one on the other, two elements c_i, c_{i+1}, and interpreting as working all elements $c_1, \ldots, c_i$, where $i \in \{1, n-1\}$. Clearly, this set of models is captured by $\mathcal{K}'$.

Now, let us show that for every model $\mathcal{I}'$ of $\mathcal{K}'$ there exists a model $\mathcal{I}$ of $\mathcal{K}$ such that $\mathcal{I}' \in U^{\mathcal{T}}(\mathcal{I}, \mathcal{F})$. Consider the model $\mathcal{I}$ obtained from $\mathcal{I}'$ by interpreting as not working c as well as all components that c depends on it, directly or indirectly. Clearly, $\mathcal{I}$ is a model of $\mathcal{K}$ and it is such that $\mathcal{I}' \in U^{\mathcal{T}}(\mathcal{I}, \mathcal{F})$. □

We are now ready to show that in general it is not possible to obtain a maximal approximation of updates.

Theorem 5.6 *There are DLs $\mathcal{L}$, KBs $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, and sets of membership assertions $\mathcal{F}$ such that maximal $(\mathcal{L}, \mathcal{T})$-approximations of $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$ do not exist.*

Proof. Let $\mathcal{L}$ be the language $\mathcal{ALCQIO}$ and let $\mathcal{T}$ and $\mathcal{F}$ be respectively the TBox and the set of assertions specified in Lemma 5.5. From the lemma, it follows that $\mathcal{K}' = \langle \mathcal{T}, \mathcal{A}' \rangle$ captures exactly the set of models $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$, where $\mathcal{A}'$ is the ABox specified in the lemma (note that $\mathcal{A}'$ is expressed in $\mathcal{ALCQIO}_{reg}$, since it uses the transitive closure of the role $DependsOn^-$).

Let $\langle \mathcal{T}, \mathcal{A}'' \rangle$ be a maximal $(\mathcal{ALCQIO}, \mathcal{T})$-approximation of $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$. Then, $\langle \mathcal{T}, \mathcal{A}'' \rangle$ logically implies $Working(c)$, because otherwise, by adding to $\mathcal{A}''$ the assertion $Working(c)$, we would obtain an $(\mathcal{ALCQIO}, \mathcal{T})$-approximation of $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$ that is better than $\langle \mathcal{T}, \mathcal{A}'' \rangle$. Similarly, $\langle \mathcal{T}, \mathcal{A}'' \rangle$ logically implies $\neg ComponentOf(p, p)$. Now, based on form of the TBox $\mathcal{T}$, and on the fact that $\langle \mathcal{T}, \mathcal{A}'' \rangle$ is a sound $(\mathcal{ALCQIO}, \mathcal{T})$-approximation of $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$, expressed in the first-order language $\mathcal{ALCQIO}$ ($\mathcal{ALCQIO}$ is indeed a fragment of first-order logic), it can be shown that $\langle \mathcal{T}, \mathcal{A}'' \rangle$ does not logically imply

$$\forall ComponentOf^-.(\exists DependsOn^-.\top) \sqcup \{c\} \sqcup \forall (DependsOn)^*.\neg Working(p)$$

Equivalently, it is not true that for every natural number n, $\mathcal{A}''$ logically implies $\bigcup_{i=0}^{n} \mathcal{A}^i$, where

$$A^i = \forall ComponentOf^-.(\exists DependsOn^-.\top) \sqcup \{c\} \sqcup \forall (DependsOn)^i.\neg Working(p)$$

Hence, let m be the maximal natural number m such that $\mathcal{A}''$ logically implies $\bigcup_{i=0}^{m} \mathcal{A}^i$, and $\mathcal{A}''$ does not logically imply $\bigcup_{i=0}^{m+1} \mathcal{A}^i$. This contradicts the fact that

$\mathcal{A}''$ is a maximal $(\mathcal{ALCQIO}, \mathcal{T})$-approximation of $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$, since by adding to $\mathcal{A}''$ the assertion $\mathcal{A}^{m+1}$, we would obtain a sound $(\mathcal{ALCQIO}, \mathcal{T})$-approximation of $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$ that is better than $\langle \mathcal{T}, \mathcal{A}'' \rangle$. Indeed, it is easy to see that $\langle \mathcal{T}, \mathcal{A}'' \cup \mathcal{A}^{m+1} \rangle$ is a sound $(\mathcal{ALCQIO}, \mathcal{T})$-approximation of $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$ such that $Mod(\langle \mathcal{T}, \mathcal{A}'' \cup \mathcal{A}^{m+1} \rangle) \subseteq Mod(\langle \mathcal{T}, \mathcal{A}'' \rangle)$. Moreover, let $\mathcal{I}^m$ be a model of $\mathcal{A}''$ such that there exist elements $d_0, d_1, \ldots, d_m, d_{m+1}$, that are not reachable from $c^{\mathcal{I}^m}$ by means of the role $DependsOn$, and are such that (*i*) $d_{m+1}^{\mathcal{I}^m}$ is working, and (*ii*) for every $i \in \{0, m\}$, $d_i^{\mathcal{I}^m}$ is not working, and $d_i^{\mathcal{I}^m}$ depends on $d_{i+1}^{\mathcal{I}^m}$. It is easy to see that $\mathcal{I}^m$ is a model of $\mathcal{A}''$ but is not a model of $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$, since $d_{m+1}^{\mathcal{I}^m}$ is working. On the other hand, $\mathcal{I}^m$ is not a model of $\langle \mathcal{T}, \mathcal{A}'' \cup \mathcal{A}^{m+1} \rangle$, which proves that $Mod(\langle \mathcal{T}, \mathcal{A}'' \cup \mathcal{A}^{m+1} \rangle) \subset Mod(\langle \mathcal{T}, \mathcal{A}'' \rangle)$. □

6 Approximated instance-level update in *DL-Lite*$_{\mathcal{F}}$

With the notion of maximal $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{M}$ in place, in this section we come back to the issue of approximating instance-level updates in *DL-Lite*$_{\mathcal{F}}$.

First, we define the notion of $(\mathcal{L}, \mathcal{T})$-update, which immediately follows from the definition of maximal $(\mathcal{L}, \mathcal{T})$-approximation given in the previous section.

Definition 6.1 **($(\mathcal{L}, \mathcal{T})$-Update)** Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, $\mathcal{K}^a = \langle \mathcal{T}, \mathcal{A}^a \rangle$ be two KBs in a DL $\mathcal{L}$, and $\mathcal{F}$ a finite set of membership assertions expressed in $\mathcal{L}$ such that $Mod(\mathcal{T}) \cap Mod(\mathcal{F}) \neq \emptyset$. We say that $\mathcal{K}^a$ is a $(\mathcal{L}, \mathcal{T})$-*update of* $\mathcal{K}$ *with* $\mathcal{F}$ if $\mathcal{K}^a$ is a maximal $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F}$.

From Theorem 5.3 we know that if an $(\mathcal{L}, \mathcal{T})$-update of $\mathcal{K}$ with $\mathcal{F}$ exists, it is unique up to logical equivalence. Moreover, by Theorem 5.4 we know that $(\mathcal{L}, \mathcal{T})$-update captures exactly the logical implication of the membership assertions of the "exact" update. Also, Theorem 5.6 shows that in general there are cases for which $(\mathcal{L}, \mathcal{T})$-updates do not exist.

We now focus our attention to computing the maximal approximation of updates in *DL-Lite*$_{\mathcal{F}}$. The simplest idea for computing the best approximation of an update to a *DL-Lite*$_{\mathcal{F}}$ would be to call $ComputeUpdate(\mathcal{T}, \mathcal{A}, \mathcal{F})$, and then ignoring all those assertions of the resulting *DL-Lite*$_{\mathcal{FS}}$ KB that are not *DL-Lite*$_{\mathcal{F}}$ assertions. Actually, this idea works, and is exactly the method used in the algorithm $ComputeUpdate^{app}$ presented in Figure 4. The algorithm takes as input a TBox $\mathcal{T}$, an ABox $\mathcal{A}$ and a set of membership assertions $\mathcal{F}$, where $\mathcal{T}$, $\mathcal{A}$ and $\mathcal{F}$ are all expressed in *DL-Lite*$_{\mathcal{F}}$, and $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ is satisfiable.

The correctness of the algorithm $ComputeUpdate^{app}$ is based on the following property.

Theorem 6.2 *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a satisfiable DL-Lite$_{\mathcal{FS}}$ KB, and α a DL-Lite$_{\mathcal{F}}$ assertion. If $\mathcal{K} \models \alpha$, then there exists a DL-Lite$_{\mathcal{F}}$ membership assertion α' in $\mathcal{A}$ such that $\langle \mathcal{T}, \{\alpha'\} \rangle \models \alpha$.*

Proof. To prove this property, we start by recalling the definition of *chase* of a *DL-Lite*$_{\mathcal{F}}$ KB [10]. Given a satisfiable *DL-Lite*$_{\mathcal{F}}$ KB $\mathcal{K}$, the chase of $\mathcal{K}$, denoted by

ALGORITHM $ComputeUpdate^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$
INPUT: finite set of *DL-Lite*$_\mathcal{F}$ membership assertions $\mathcal{F}$,
satisfiable *DL-Lite*$_\mathcal{F}$ KB $\langle \mathcal{T}, \mathcal{A} \rangle$
OUTPUT: an ABox $\mathcal{A}^a$, or ERROR
[1] if $\langle \mathcal{T}, \mathcal{F} \rangle$ is not satisfiable **then return** ERROR
[2] else begin
[3] $\mathcal{A}^a = ComputeUpdate(\mathcal{T}, \mathcal{A}, \mathcal{F})$;
[4] delete from $\mathcal{A}^a$ all the assertions that are not *DL-Lite*$_\mathcal{F}$ membership assertions;
[5] **return** $\mathcal{A}^a$
[6] end

Figure 4: Algorithm $ComputeUpdate^{app}$

$chase(\mathcal{K})$, is a (possibly infinite) ABox obtained by closing the initial ABox $\mathcal{A}$ with respect to the following *inclusion chase rules* (where A, A_1, A_2 denote concept symbols, and R, R_1, R_2 role symbols):

- if $A_1 \sqsubseteq A_2 \in \mathcal{T}$ and there is an assertion of the form $A_1(a)$ in $chase(\mathcal{K})$ and $A_2(a) \notin chase(\mathcal{K})$, then add the assertion $A_2(a)$;
- if $\exists R \sqsubseteq A \in \mathcal{T}$ (respectively, $\exists R^- \sqsubseteq A \in \mathcal{T}$) and there is an assertion of the form $R(a, b)$ (respectively, $R(b, a)$) in $chase(\mathcal{K})$ and $A(a) \notin chase(\mathcal{K})$, then add the assertion $A(a)$;
- if $A \sqsubseteq \exists R \in \mathcal{T}$ (respectively, $A \sqsubseteq \exists R^- \in \mathcal{T}$) and there is an assertion of the form $A(a)$ in $chase(\mathcal{K})$ and there is no assertion of the form $R(a, x)$ in $chase(\mathcal{K})$ (where x is any constant symbol), then add the assertion $R(a, n)$ (respectively, $R(n, a)$) where n is a new constant symbol (i.e., a symbol not occurring already in $chase(\mathcal{K})$);
- if $\exists R_1 \sqsubseteq \exists R_2 \in \mathcal{T}$ (respectively, $\exists R_1^- \sqsubseteq \exists R_2 \in \mathcal{T}$) and there is an assertion of the form $R_1(a, b)$ (respectively, $R(b, a)$) in $chase(\mathcal{K})$ and there is no assertion of the form $R_2(a, x)$ in $chase(\mathcal{K})$ (where x is any constant symbol), then add the assertion $R_2(a, n)$ where n is a new constant symbol (i.e., a symbol not occurring already in $chase(\mathcal{K})$);
- if $\exists R_1 \sqsubseteq \exists R_2^- \in \mathcal{T}$ (respectively, $\exists R_1^- \sqsubseteq \exists R_2^- \in \mathcal{T}$) and there is an assertion of the form $R_1(a, b)$ (respectively, $R(b, a)$) in $chase(\mathcal{K})$ and there is no assertion of the form $R_2(x, a)$ in $chase(\mathcal{K})$ (where x is any constant symbol), then add the assertion $R_2(n, a)$ where n is a new constant symbol (i.e., a symbol not occurring already in $chase(\mathcal{K})$).

In [10] it has been shown that $chase(\mathcal{K})$ identifies a *canonical model* for conjunctive queries over $\mathcal{K}$: namely, conjunctive queries over $\mathcal{K}$ can be decided by simply evaluating them over $chase(\mathcal{K})$. As a corollary of this property, we get the following lemma.

Lemma 6.3 *For every satisfiable DL-Lite*$_\mathcal{F}$ *KB* $\mathcal{K}$ *and for every DL-Lite*$_\mathcal{F}$ *membership assertion* α, $\mathcal{K} \models \alpha$ *iff* $\alpha \in chase(\mathcal{K})$.

Coming back to the proof of the theorem, suppose now that $\mathcal{K} \models \alpha$: then, from Lemma 6.3, $\alpha \in chase(\mathcal{K})$. There are two possible cases:

1. $\alpha \in \mathcal{A}$: in this case, the thesis holds for $\alpha' = \alpha$;
2. $\alpha \notin \mathcal{A}$: by the inductive definition of $chase(\mathcal{K})$, it immediately follows that there is a sequence of assertions $\alpha_1, \ldots, \alpha_n$ such that $\alpha_1 \in \mathcal{A}$ and each α_{i+1} is obtained by applying an inclusion chase rule to α_i and to some inclusion assertion in $\mathcal{T}$. Consequently, $\alpha \in chase(\langle \mathcal{T}, \{\alpha_1\}\rangle)$ and thus from Lemma 6.3 it follows that $\langle \mathcal{T}, \{\alpha_1\}\rangle \models \alpha$. Therefore, the thesis holds for $\alpha' = \alpha_1$.

□

We are now ready to prove the correctness of the algorithm $ComputeUpdate^{app}$.

Theorem 6.4 *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$ be a satisfiable DL-Lite$_\mathcal{F}$ KB, $\mathcal{F}$ a finite set of DL-Lite$_\mathcal{F}$ membership assertions such that $Mod(\mathcal{T} \cup \mathcal{F}) \neq \emptyset$, and let $\mathcal{K}^a = \langle \mathcal{T}, \mathcal{A}^a\rangle$, where $\mathcal{A}^a$ is the ABox returned by $ComputeUpdate^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$. Then, $\mathcal{K}^a$ is a (DL-Lite$_\mathcal{F}$, $\mathcal{T}$)-update of $\mathcal{K}$ with $\mathcal{F}$.*

Proof. Clearly the algorithm $ComputeUpdate^{app}$ terminates, since so does $ComputeUpdate$. Now, let $\mathcal{A}^a = ComputeUpdate^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$. We first show that $\langle \mathcal{T}, \mathcal{A}^a\rangle$ is a sound (DL-Lite$_\mathcal{F}$, $\mathcal{T}$)-approximation of $\mathcal{K} \circ_\mathcal{T} \mathcal{F}$, then we show that it is a maximal one. Let $\mathcal{A}^p = ComputeUpdate(\mathcal{T}, \mathcal{A}, \mathcal{F})$. By construction, $\mathcal{A}^a \subseteq \mathcal{A}^p$ and therefore, since DL-Lite$_\mathcal{F}$ is monotone, $Mod(\mathcal{A}^p) \subseteq Mod(\mathcal{A}^a)$. Hence $Mod(\langle \mathcal{T}, \mathcal{A}^p\rangle) \subseteq Mod(\langle \mathcal{T}, \mathcal{A}^a\rangle)$. Moreover, by Theorem 4.8, $Mod(\langle \mathcal{T}, \mathcal{A}^p\rangle) = \mathcal{K} \circ_\mathcal{T} \mathcal{F}$. It follows that $\langle \mathcal{T}, \mathcal{A}^a\rangle$ is a sound (DL-Lite$_\mathcal{F}$, $\mathcal{T}$)-approximation of $\mathcal{K} \circ_\mathcal{T} \mathcal{F}$. Now, let us show that $\mathcal{K}^a = \langle \mathcal{T}, \mathcal{A}^a\rangle$ is the maximal (DL-Lite$_\mathcal{F}$, $\mathcal{T}$)-approximation of $\mathcal{K} \circ_\mathcal{T} \mathcal{F}$. By contradiction, let $\mathcal{K}' = \langle \mathcal{T}, \mathcal{A}'\rangle$ be a sound (DL-Lite$_\mathcal{F}$, $\mathcal{T}$)-approximation of $\mathcal{K} \circ_\mathcal{T} \mathcal{F}$ such that $Mod(\mathcal{K}') \subset Mod(\mathcal{K}^a)$. Since $Mod(\mathcal{K}' \cup \mathcal{K}^a) = Mod(\mathcal{K}') \cap Mod(\mathcal{K}^a)$, we have that $Mod(\mathcal{K}' \cup \mathcal{K}^a) = Mod(\mathcal{K}')$, which implies that that $\mathcal{K}^a \subset \mathcal{K}'$, and thus that there exists a DL-Lite$_\mathcal{F}$ membership assertion α such that $\alpha \in \mathcal{A}'$, $\mathcal{K} \circ_\mathcal{T} \mathcal{F} \models \alpha$ and $\mathcal{K}^a \not\models \alpha$. Let $\mathcal{A}^p = ComputeUpdate(\mathcal{T}, \mathcal{A}, \mathcal{F})$ and $\mathcal{K}^p = \langle \mathcal{T}, \mathcal{A}^p\rangle$. By Theorem 4.8, $Mod(\mathcal{K}^p) = \mathcal{K} \circ_\mathcal{T} \mathcal{F}$. Then we have that $\mathcal{K}^p \models \alpha$, where α is a membership assertion in DL-Lite$_\mathcal{F}$. By Theorem 6.2, there must exist a DL-Lite$_\mathcal{F}$ membership assertion α' in $\mathcal{A}^p$ such that $\langle \mathcal{T}, \alpha'\rangle \models \alpha$. But then, by construction, α' belongs to $\mathcal{A}^a$, contradicting $\mathcal{K}^a \not\models \alpha$. □

Since $ComputeUpdate(\mathcal{T}, \mathcal{A}, \mathcal{F})$ runs in polynomial time, it follows immediately that the algorithm $ComputeUpdate^{app}$ also terminates and runs in time polynomial with respect to the size of its input.

Theorem 6.5 *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$ be a satisfiable DL-Lite$_\mathcal{F}$ KB, and $\mathcal{F}$ a finite set of DL-Lite$_\mathcal{F}$ membership assertions. Then $ComputeUpdate^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$ terminates, returning ERROR if $Mod(\mathcal{T}) \cap Mod(\mathcal{F}) = \emptyset$, and an ABox $\mathcal{A}^a$ such that $\langle \mathcal{T}, \mathcal{A}^a\rangle$ is a DL-Lite$_\mathcal{F}$ KB, otherwise. Moreover:*

- *the size of $\mathcal{A}^a$ is polynomially bounded by the size of $\mathcal{T} \cup \mathcal{A} \cup \mathcal{F}$;*
- *$\mathcal{A}^a$ is computed in polynomial time in the size of $\mathcal{T} \cup \mathcal{A} \cup \mathcal{F}$.*

Example 6.6 Consider the *DL-Lite*$_{\mathcal{F}}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ mentioned in Example 4.10, and let us compute the (*DL-Lite*$_{\mathcal{F}}$, $\mathcal{T}$)-update of $\mathcal{K}$ with $\mathcal{F} = \{A_1(a)\}$. First, we apply the update algorithm $ComputeUpdate$ of Section 4. This returns a *DL-Lite*$_{\mathcal{FS}}$ ABox $\mathcal{A}' = \{A_1(z_{\exists P(a)}),\ A_2(a)\}$. Then, we delete from $\mathcal{A}'$ all assertions that are not *DL-Lite*$_{\mathcal{F}}$ membership assertions, and obtain the *DL-Lite*$_{\mathcal{F}}$ ABox $\mathcal{A}^a = \{A_1(a)\}$. ∎

Finally, we show that $ComputeUpdate^{app}$ captures, in a sound and complete way, logical implication of *DL-Lite*$_{\mathcal{F}}$ assertions after update.

Theorem 6.7 *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a satisfiable DL-Lite$_{\mathcal{F}}$ KB, $\mathcal{F}$ a finite set of DL-Lite$_{\mathcal{F}}$ membership assertions such that $Mod(\mathcal{T} \cup \mathcal{F}) \neq \emptyset$, and $\mathcal{K}^a = \langle \mathcal{T}, \mathcal{A}^a \rangle$, where $\mathcal{A}^a$ is the ABox returned by $ComputeUpdate^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$. Then, for every membership assertion α in DL-Lite$_{\mathcal{F}}$, we have that $\mathcal{K} \circ_{\mathcal{T}} \mathcal{F} \models \alpha$ iff $\mathcal{K}^a \models \alpha$.*

Proof. The proof is an immediate consequence of from Theorem 6.4 and Theorem 5.4. □

7 Instance-level erasure

In this section we consider the operation of instance-level *erasure* [23]. This is the operation consisting of retracting (or deleting) membership assertions from a DL KB, while keeping the TBox unchanged. So, erasure is in fact complementary to the update operation studied in the previous sections. We show that, in a way similar to the update operation, the result of an erasure operation against an $\mathcal{L}$ KB is in general not expressible in the DL $\mathcal{L}$. Thus, we introduce the notion of $(\mathcal{L}, \mathcal{T})$-erasure, i.e., a maximal approximation in $\mathcal{L}$ of the result of an erasure against an $\mathcal{L}$ KB $\langle \mathcal{T}, \mathcal{A} \rangle$. Finally, we study the problem of computing $(\mathcal{L}, \mathcal{T})$-erasures in *DL-Lite*$_{\mathcal{F}}$.

We start by formally defining instance-level erasure over DL KBs.

Definition 7.1 (**Instance-level erasure**) Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a KB expressed in a DL $\mathcal{L}$, and $\mathcal{F}$ a finite set of membership assertions expressed in $\mathcal{L}$ such that $Mod(\mathcal{T} \cup \neg\mathcal{F}) \neq \emptyset$, where $\neg\mathcal{F}$ denotes the set of membership assertions $\{\neg F_i \mid F_i \in \mathcal{F}\}$. The *instance-level erasure of $\mathcal{F}$ from $\mathcal{K}$*, or simply the *erasure of $\mathcal{F}$ from $\mathcal{K}$*, denoted $\mathcal{K} \bullet_{\mathcal{T}} \mathcal{F}$, is defined as follows:

$$\mathcal{K} \bullet_{\mathcal{T}} \mathcal{F} = Mod(\mathcal{K}) \cup \Big(\bigcup_{\mathcal{I} \in Mod(\mathcal{K})} U^{\mathcal{T}}(\mathcal{I}, \neg\mathcal{F}) \Big).$$

Intuitively, the result of erasing a finite set of formulas $\mathcal{F}$ from a KB $\mathcal{K}$ should be any KB that does not logically imply any of the formulas in $\mathcal{F}$, and whose set of models minimally differs from the set of models of $\mathcal{K}$.

The following simple example illustrates the erasure operation.

Example 7.2 Consider the KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ with TBox $\mathcal{T} = \{A \sqsubseteq B, A \sqsubseteq C\}$ and the ABox $\mathcal{A} = \{A(d)\}$. Now, given the set of membership assertions $\mathcal{F} = \{C(d)\}$, consider $\mathcal{K} \bullet_{\mathcal{T}} \mathcal{F}$, i.e., the erasure of $\mathcal{F}$ from $\mathcal{K}$. By definition, $\mathcal{K} \bullet_{\mathcal{T}} \mathcal{F} = Mod(\mathcal{K}) \cup$

ALGORITHM $ComputeErasure^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$
INPUT: finite set of *DL-Lite*$_\mathcal{F}$ membership assertions $\mathcal{F}$,
satisfiable *DL-Lite*$_\mathcal{F}$ KB $\langle \mathcal{T}, \mathcal{A} \rangle$
OUTPUT: an ABox $\mathcal{A}^a$, or ERROR
[1] if $\langle \mathcal{T}, \neg\mathcal{F} \rangle$ is not satisfiable **then return** ERROR
[2] else begin
[3] $\mathcal{A}^a := ComputeUpdate(\mathcal{T}, \mathcal{A}, \neg\mathcal{F})$;
[4] delete from $\mathcal{A}^a$ all the assertions that are not *DL-Lite*$_\mathcal{F}$ membership assertions;
[5] **return** $\mathcal{A}^a$
[6] end

Figure 5: Algorithm $ComputeErasure^{app}$

$(\mathcal{K} \circ_\mathcal{T} \{\neg C(d)\})$. Thus, each model $\mathcal{I}'$ in $\mathcal{K} \bullet_\mathcal{T} \mathcal{F}$ is obtained from a model $\mathcal{I}$ of $\mathcal{K}$ by either not modifying anything, or by modifying the interpretation of d so that d does not belong to A. Hence, for each $\mathcal{I}'$ we must have $d \in B^{\mathcal{I}'}$, and either $d \in A^{\mathcal{I}'}, d \in C^{\mathcal{I}'}$ or $d \notin A^{\mathcal{I}'}, d \notin C^{\mathcal{I}'}$. ∎

In the same way as in the case of update, we now introduce the notion of maximal approximation of instance-level erasure in a DL $\mathcal{L}$.

Definition 7.3 **($(\mathcal{L}, \mathcal{T})$-Erasure)** Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, $\mathcal{K}^a = \langle \mathcal{T}, \mathcal{A}^a \rangle$ be two KBs in a DL $\mathcal{L}$ and $\mathcal{F}$ a finite set of membership assertions expressed in $\mathcal{L}$ such that $Mod(\mathcal{T}) \cap Mod(\neg\mathcal{F}) \neq \emptyset$. We say that $\mathcal{K}^a$ is a $(\mathcal{L}, \mathcal{T})$-*erasure of* $\mathcal{K}$ *with* $\mathcal{F}$ if $\mathcal{K}^a$ is a maximal $(\mathcal{L}, \mathcal{T})$-approximation of $\mathcal{K} \bullet_\mathcal{T} \mathcal{F}$.

We now study instance-level erasure in *DL-Lite*$_\mathcal{F}$. We start by showing that, in *DL-Lite*$_\mathcal{F}$, the result of an erasure cannot be always expressed in terms of a *DL-Lite*$_\mathcal{F}$ KB.

Theorem 7.4 *The result of an erasure to a DL-Lite*$_\mathcal{F}$ *KB may not be expressible in DL-Lite*$_\mathcal{F}$ *itself.*

Proof. Consider again the *DL-Lite*$_\mathcal{F}$ KB $\mathcal{K}$ shown in Example 7.2, i.e., $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ with $\mathcal{T} = \{A \sqsubseteq B, A \sqsubseteq C\}$ and $\mathcal{A} = \{A(d)\}$. Let $\mathcal{F} = \{C(d)\}$. By definition, $\mathcal{K} \bullet_\mathcal{T} \mathcal{F} = Mod(\mathcal{K}) \cup (\mathcal{K} \circ_\mathcal{T} \{\neg C(d)\})$. Thus, each model $\mathcal{I}$ in $\mathcal{K} \bullet_\mathcal{T} \mathcal{F}$ is obtained from a model $\mathcal{I}$ of $\mathcal{K}$ by either not modifying anything, or by modifying the interpretation of d so that d does not belong to $A^\mathcal{I}$. Hence, for each $\mathcal{I}$, either $d \in A^\mathcal{I}, d \in C^\mathcal{I}$ or $d \notin A^\mathcal{I}, d \notin C^\mathcal{I}$. It is immediate to verify that there is no way to express this set of models through a set of *DL-Lite*$_\mathcal{F}$ membership assertions. □

Therefore, like in the case of update, in *DL-Lite*$_\mathcal{F}$ it is interesting to look at maximal approximations of instance-level erasure. For this purpose, we define the algorithm $ComputeErasure^{app}$ shown in Figure 5.

The following theorem shows that the maximal approximation of instance-level erasure in a *DL-Lite*$_\mathcal{F}$ KB always exists, and is computed by the algorithm $ComputeErasure^{app}$.

Theorem 7.5 *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a satisfiable DL-Lite$_\mathcal{F}$ KB, $\mathcal{F}$ a finite set of DL-Lite$_\mathcal{F}$ membership assertions such that $Mod(\mathcal{T} \cup \neg\mathcal{F}) \neq \emptyset$, and $\mathcal{K}^a = \langle \mathcal{T}, \mathcal{A}^a \rangle$, where $\mathcal{A}^a$ is the ABox returned by $ComputeErasure^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$. Then, $\mathcal{K}^a$ is a (DL-Lite$_\mathcal{F}$, $\mathcal{T}$)-erasure of $\mathcal{K}$ with $\mathcal{F}$.*

Proof. First, from definition of erasure, $\mathcal{K}\bullet_\mathcal{T}\mathcal{F} = Mod(\mathcal{K}) \cup (\mathcal{K} \circ_\mathcal{T} \neg\mathcal{F})$. Then, by definition of the algorithm $ComputeErasure^{app}$, it follows that for every membership assertion $\alpha \in \mathcal{A}^a - \mathcal{A}$, we have $\mathcal{K} \models \alpha$, which immediately implies that (i) $Mod(\mathcal{K}) \subseteq Mod(\mathcal{K}^a)$. Moreover, let $\mathcal{K}^p = \langle \mathcal{T}, \mathcal{A}^p \rangle$ where $\mathcal{A}^p$ is the ABox returned by $ComputeUpdate^{app}(\mathcal{T}, \mathcal{A}, \neg\mathcal{F})$: by definition, $\mathcal{A}^a \subseteq \mathcal{A}^p$, consequently every model of $\mathcal{K}^p$ is also a model of $\mathcal{K}^a$, and since by Theorem 4.8 $Mod(\mathcal{K}^p) = \mathcal{K} \circ_\mathcal{T} \neg\mathcal{F}$, it follows that (ii) $\mathcal{K} \circ_\mathcal{T} \neg\mathcal{F} \subseteq Mod(\mathcal{K}^a)$. Hence, from (i) and (ii) it follows that $\mathcal{K}^a$ is a (DL-Lite$_\mathcal{F}$, $\mathcal{T}$)-approximation of $\mathcal{K}\bullet_\mathcal{T}\mathcal{F}$.

Now, suppose $\mathcal{K}^a$ is not the maximal (DL-Lite$_\mathcal{F}$, $\mathcal{T}$)-approximation of $\mathcal{K}\bullet_\mathcal{T}\mathcal{F}$. Then, there exists a DL-Lite$_\mathcal{F}$ KB $\mathcal{K}' = \langle \mathcal{T}, \mathcal{A}' \rangle$ such that $\mathcal{K}\bullet_\mathcal{T}\mathcal{F} \subseteq Mod(\mathcal{K}') \subset Mod(\mathcal{K}^a)$. Since $Mod(\mathcal{K}') \subset Mod(\mathcal{K}^a)$, there exists at least a (membership) assertion $\alpha \in \mathcal{A}' - \mathcal{A}^a$ such that $\mathcal{K}^a \not\models \alpha$, and since $\mathcal{K}\bullet_\mathcal{T}\mathcal{F} \subseteq Mod(\mathcal{K}')$ and $\mathcal{K}\bullet_\mathcal{T}\mathcal{F} = Mod(\mathcal{K}) \cup (\mathcal{K} \circ_\mathcal{T} \neg\mathcal{F})$, it follows that $\mathcal{K}^p \models \alpha$. Now, by Theorem 6.2 it follows that there exists a membership assertion $\alpha' \in \mathcal{A}^p$ such that $\langle \mathcal{T}, \{\alpha'\} \rangle \models \alpha$. Hence, by definition of $ComputeErasure^{app}$, it follows that $\alpha' \in \mathcal{A}^a$, consequently $\mathcal{K}^a \models \alpha$. Contradiction. Therefore, $\mathcal{K}^a$ is the maximal (DL-Lite$_\mathcal{F}$, $\mathcal{T}$)-approximation of $\mathcal{K}\bullet_\mathcal{T}\mathcal{F}$. □

Observe that, as mentioned in the previous section, the algorithm $ComputeUpdate(\mathcal{T}, \mathcal{A}, \mathcal{F})$ runs in polynomial time, and therefore also the algorithm $ComputeErasure^{app}$ runs in time polynomial with respect to the size of its input.

Theorem 7.6 *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a satisfiable DL-Lite$_\mathcal{F}$ KB, and $\mathcal{F}$ a finite set of DL-Lite$_\mathcal{F}$ membership assertions. Then $ComputeErasure^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$ terminates, returning ERROR if $Mod(\mathcal{T}) \cap Mod(\neg\mathcal{F}) = \emptyset$, and an ABox $\mathcal{A}'$ such that $\langle \mathcal{T}, \mathcal{A}' \rangle$ is a DL-Lite$_\mathcal{F}$ KB, otherwise. Moreover:*

- *the size of $\mathcal{A}'$ is polynomially bounded by the size of $\mathcal{T} \cup \mathcal{A} \cup \mathcal{F}$;*
- *$\mathcal{A}'$ is computed in polynomial time in the size of $\mathcal{T} \cup \mathcal{A} \cup \mathcal{F}$.*

The following example illustrates the algorithm $ComputeErasure^{app}$.

Example 7.7 Consider the DL-Lite$_\mathcal{F}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ introduced in the example of the proof of Theorem 7.4. Now suppose to compute the (DL-Lite$_\mathcal{F}$, $\mathcal{T}$)-erasure of $\mathcal{K}$ with $\mathcal{F} = \{C(a)\}$. First, we apply the update algorithm $ComputeUpdate$ and compute the ABox $\mathcal{A}^p = ComputeUpdate(\mathcal{T}, \mathcal{A}, \{\neg C(a)\})$. This returns a DL-Lite$_{\mathcal{FS}}$ ABox that is obtained from $\mathcal{A}$ by removing the assertion $A(a)$, and introducing the assertions $\neg C(a)$ and $B(a)$. Second, we perform the projection of $\mathcal{A}^p$ in DL-Lite$_\mathcal{F}$, and obtain the DL-Lite$_\mathcal{F}$ ABox $\mathcal{A}^a = \{B(a)\}$. ■

Finally, we prove the following important property: computing a maximal approximation of an erasure in *DL-Lite*$_\mathcal{F}$ is indeed sufficient to decide, in a sound and complete way, instance checking over the exact result of the erasure. In other words, we prove that the algorithm $ComputeErasure^{app}$ captures, in a sound and complete way, logical implication of *DL-Lite*$_\mathcal{F}$ assertions after erasure.

Theorem 7.8 *Let $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a satisfiable DL-Lite$_\mathcal{F}$ KB, $\mathcal{F}$ a finite set of DL-Lite$_\mathcal{F}$ membership assertions such that $Mod(\mathcal{T} \cup \neg\mathcal{F}) \neq \emptyset$, and $\mathcal{K}^a = \langle \mathcal{T}, \mathcal{A}^a \rangle$, where $\mathcal{A}^a$ is the ABox returned by $ComputeErasure^{app}(\mathcal{T}, \mathcal{A}, \mathcal{F})$. Then, for every membership assertion α in DL-Lite$_\mathcal{F}$, we have that $\mathcal{K} \bullet_\mathcal{T} \mathcal{F} \models \alpha$ iff $\mathcal{K}^a \models \alpha$.*

Proof. The proof is an immediate consequence of Theorem 7.5 and Theorem 5.4. □

8 Conclusion

We have investigated the notion of instance-level update and erasure of a DL KB. Specifically, we have focused on *DL-Lite*$_\mathcal{F}$, a tractable DL tailored for data intensive applications. Since in general the result of instance-level update and erasure cannot be expressed as a KB in the same language as the original KB, we have provided a principled notion of maximal approximation, and have presented polynomial algorithms for computing such maximal approximations in the case of *DL-Lite*$_\mathcal{F}$. These results confirm the nice computational properties of *DL-Lite*$_\mathcal{F}$ for data intensive applications, even when information about instances is not only read but also written.

There are several interesting directions for continuing our research. First, we have implemented in the QuOnto reasoning system [1] the algorithms presented in this paper. Related to this point, we are currently studying optimization techniques to deal with ontologies that include very large ABoxes, as those produced by materializing data in ontology-based information integration applications. Notably, for such kinds of applications it would also be interesting to avoid actual materialization of data, and "push" updates and erasures into the data sources. This task is very challenging, since it corresponds to an advanced form of the notorious view update problem in databases [6].

Second, the kind of approximation considered in this paper preserves logical implication of ABox and TBox assertions in the DL considered. Obviously, it would be interesting both to consider different notions of approximation and to study completeness of the approximation with respect to more expressive classes of formulas. In particular, we are currently studying the properties of approximation in *DL-Lite*$_\mathcal{F}$ KBs for several classes of unions of conjunctive queries.

Third, in this paper we adopted a classical model-based approach to update and erasure, stemming from the existing literature on updating knowledge bases. Other approaches to update and erasure have been studied and their application to ontology might be of interest, as well as approaches based on belief revision and contraction. We believe that, in principle, several approaches to ontology evolution could coexist on the same ontology management system, in order to model different types of services involving some sort of ontology evolution.

Finally, updates bring in the general issue of dealing with inconsistency in ontologies. The semantics that we have considered in this paper address the issue of solving inconsistency between the current instance level of the ontology and what has been asserted (retracted) by the update (erasure), while it does not deal with inconsistencies between the update and the intensional level. It would be interesting to study possible semantics that are tolerant with respect to the latter form of inconsistency.

Acknowledgments

This research has been partially supported by FET project TONES (Thinking ONtologiES), funded by the EU under contract number FP6-7603, by project HYPER, funded by IBM through a Shared University Research (SUR) Award grant, and by MIUR FIRB 2005 project "Tecnologie Orientate alla Conoscenza per Aggregazioni di Imprese in Internet" (TOCAI.IT).

References

[1] Andrea Acciarri, Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Mattia Palmieri, and Riccardo Rosati. QUONTO: QUerying ONTOlogies. In *Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005)*, pages 1670–1671, 2005.

[2] C. E. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change: Partial meet contraction and revision functions. *J. of Symbolic Logic*, 50:510–530, 1985.

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

[4] Franz Baader, Ian Horrocks, and Ulrike Sattler. Description logics as ontology languages for the semantic web. In *Mechanizing Mathematical Reasoning: Essays in Honor of Jrg Siekmann on the Occasion of his 60th Birthday*, volume 2605 of *Lecture Notes in Artificial Intelligence*, pages 228–248. Springer, 2005.

[5] Franz Baader, Carsten Lutz, Maja Milicic, Ulrike Sattler, and Frank Wolter. Integrating description logics and action formalisms: First results. In *Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005)*, pages 572–577, 2005.

[6] François Bancilhon and Nicolas Spyratos. Update semantics of relational views. *ACM Trans. on Database Systems*, 6(4):557–575, 1981.

[7] Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, May 2001.

[8] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. *DL-Lite*: Tractable description logics for ontologies. In

Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005), pages 602–607, 2005.

[9] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Data complexity of query answering in description logics. In *Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 260–270, 2006.

[10] 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. *J. of Automated Reasoning*, 39(3):385–429, 2007.

[11] Giuseppe De Giacomo, Maurizio Lenzerini, Antonella Poggi, and Riccardo Rosati. On the update of description logic ontologies at the instance level. In *Proc. of the 21st Nat. Conf. on Artificial Intelligence (AAAI 2006)*, 2006.

[12] Giuseppe De Giacomo, Maurizio Lenzerini, Antonella Poggi, and Riccardo Rosati. On the approximation of instance level update and erasure in description logics. In *Proc. of the 22nd Nat. Conf. on Artificial Intelligence (AAAI 2007)*, 2007.

[13] Thomas Eiter and Georg Gottlob. On the complexity of propositional knowledge base revision, updates and counterfactuals. *Artificial Intelligence*, 57:227–270, 1992.

[14] Giorgos Flouris, Dimitris Plexousakis, and Grigoris Antoniou. On applying the AGM theory to DLs and OWL. In *Proc. of the 4th Int. Semantic Web Conf. (ISWC 2005)*, 2005.

[15] Giorgos Flouris, Dimitris Plexousakis, and Grigoris Antoniou. A classification of ontology change. In *Proc. of the 3rd Italian Semantic Web Workshop: Semantic Web Applications and Perspectives (SWAP 2006)*, 2006.

[16] Giorgos Flouris, Dimitris Plexousakis, and Grigoris Antoniou. Evolving ontology evolution. In *Proc. of the 32nd Int. Conf. on Current Trends in Theory and Practice of Computer Science (SOFSEM 2006)*, 2006.

[17] Claudio Gutiérrez, Carlos Hurtado, and Alejandro Vaisman. The meaning of erasing in RDF under the Katsuno-Mendelzon approach. In *Proc. of the 9th Int. Workshop on the Web and Databases (WebDB 2006)*, 2006.

[18] Volker Haarslev and Ralf Möller. RACER system description. In *Proc. of the Int. Joint Conf. on Automated Reasoning (IJCAR 2001)*, volume 2083 of *Lecture Notes in Artificial Intelligence*, pages 701–705. Springer, 2001.

[19] Peter Haase and Ljiljana Stojanovic. Consistent evolution of owl ontologies. In *Proc. of the 2nd European Semantic Web Conf. (ESWC 2005)*, pages 182–197, 2005.

[20] Christian Halaschek-Wiener, Bijan Parsia, Evren Sirin, and Aditya Kalyanpur. Description logic reasoning for dynamic aboxes. In *Proc. of the 2006 Description Logic Workshop (DL 2006)*, volume 189 of *CEUR Electronic Workshop Proceedings*, `http://ceur-ws.org/Vol-189/`, 2006.

[21] Andreas Herzig and Omar Rifi. Propositional belief update and minimal change. *Artificial Intelligence*, 115(1):107–138, 1999.

[22] Ian Horrocks. The FaCT system. In Harrie de Swart, editor, *Proc. of the 7th Int. Conf. on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX'98)*, volume 1397 of *Lecture Notes in Artificial Intelligence*, pages 307–312. Springer, 1998.

[23] Hirofumi Katsuno and Alberto Mendelzon. On the difference between updating a knowledge base and revising it. In *Proc. of the 2nd Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR'91)*, pages 387–394, 1991.

[24] Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proc. of the 21st ACM SIGACT SIGMOD SIGART Symp. on Principles of Database Systems (PODS 2002)*, pages 233–246, 2002.

[25] Hector J. Levesque and Gerhard Lakemeyer. *The Logic of Knowledge Bases*. The MIT Press, 2001.

[26] H. Liu, C. Lutz, M. Milicic, and F. Wolter. Updating description logic ABoxes. In *Proc. of the 10th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR 2006)*, pages 46–56, 2006.

[27] Thomas Meyer, Kevin Lee, and Richard Booth. Knowledge integration for description logics. In *Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005)*, pages 645–650, 2005.

[28] Antonella Poggi. *Structured and Semi-Structured Data Integration*. PhD thesis, Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", 2006.

[29] Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. Linking data to ontologies. *J. on Data Semantics*, X:133–173, 2008.

[30] Raymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. The MIT Press, 2001.

[31] Marco Schaerf and Marco Cadoli. Tractable reasoning via approximation. *Artificial Intelligence*, 74(2):249–310, 1995.

[32] Richard B. Scherl and Hector J. Levesque. Knowledge, action, and the frame problem. *Artificial Intelligence*, 144(1–2):1–39, 2003.

[33] Evren Sirin and Bijan Parsia. Pellet: An OWL DL reasoner. In *Proc. of the 2004 Description Logic Workshop (DL 2004)*, volume 104 of *CEUR Electronic Workshop Proceedings*, `http://ceur-ws.org/Vol-104/`, 2004.

[34] Marianne Winslett. Reasoning about action using a possible models approach. In *Proc. of the 15th Nat. Conf. on Artificial Intelligence (AAAI'98)*, 1988.

[35] Marianne Winslett. *Updating Logical Databases*. Cambridge University Press, 1990.