Beyond OWL 2 QL in OBDA: Rewritings and Approximations (Extended Version)

Elena Botoeva[1], **Diego Calvanese**[1], **Valerio Santarelli**[2], **Domenico F. Savo**[2],
Alessandro Solimando[3], and **Guohui Xiao**[1]

[1] KRDB Research Centre for Knowledge and Data, Free University of Bozen-Bolzano, Italy, *lastname*@inf.unibz.it
[2] Dip. di Ing. Informatica Automatica e Gestionale, Sapienza Università di Roma, Italy, *lastname*@dis.uniroma1.it
[3] DIBRIS, University of Genova, Italy, alessandro.solimando@unige.it

Abstract

Ontology-based data access (OBDA) is a novel paradigm facilitating access to relational data, realized by linking data sources to an ontology by means of declarative mappings. *DL-Lite*$_\mathcal{R}$, which is the logic underpinning the W3C ontology language OWL 2 QL and the current language of choice for OBDA, has been designed with the goal of delegating query answering to the underlying database engine, and thus is restricted in expressive power. E.g., it does not allow one to express disjunctive information, and any form of recursion on the data. The aim of this paper is to overcome these limitations of *DL-Lite*$_\mathcal{R}$, and extend OBDA to more expressive ontology languages, while still leveraging the underlying relational technology for query answering. We achieve this by relying on two well-known mechanisms, namely conservative rewriting and approximation, but significantly extend their practical impact by bringing into the picture the mapping, an essential component of OBDA. Specifically, we develop techniques to rewrite OBDA specifications with an expressive ontology to "equivalent" ones with a *DL-Lite*$_\mathcal{R}$ ontology, if possible, and to approximate them otherwise. We do so by exploiting the high expressive power of the mapping layer to capture part of the domain semantics of rich ontology languages. We have implemented our techniques in the prototype system ONTOPROX, making use of the state-of-the-art OBDA system ONTOP and the query answering system CLIPPER, and we have shown their feasibility and effectiveness with experiments on synthetic and real-world data.

1 Introduction

Ontology-Based Data Access (OBDA) is a popular paradigm that enables end users to access data sources through an ontology, abstracting away low-level details of the data sources themselves. The ontology provides a high-level description of the domain of interest, and is semantically linked to the data sources by means of a set of mapping assertions (Calvanese et al. 2009; Giese et al. 2015). Typically, the data sources are represented as relational data, the ontology is constituted by a set of logical axioms over concepts and roles, and each mapping assertion relates an SQL query over the database to a concept or role of the ontology.

As an example, consider a bank domain, where we can specify that a checking account in the name of a person is a simple account by means of the axiom (expressed in description logic notation) $\mathsf{CAcc} \sqcap \exists \mathsf{inNameOf}.\mathsf{Person} \sqsubseteq \mathsf{SAcc}$. We assume that the information about the accounts and their owners is stored in a database $\mathcal{D}$, and that the ontology terms CAcc, $\mathsf{inNameOf}$, and Person are connected to $\mathcal{D}$ respectively via the mapping assertions $sql_1(x) \rightsquigarrow \mathsf{CAcc}(x)$, $sql_2(x,y) \rightsquigarrow \mathsf{inNameOf}(x,y)$ and $sql_3(x) \rightsquigarrow \mathsf{Person}(x)$, where each sql_i is a (possibly very complex) SQL query over $\mathcal{D}$. Suppose now that the user intends to extract all simple accounts from $\mathcal{D}$. Formulating such a query directly over $\mathcal{D}$ would require to know precisely how $\mathcal{D}$ is structured, and thus could be complicated. Instead, exploiting OBDA, the user can simply query the ontology with $q(x) = \mathsf{SAcc}(x)$, and rely on the OBDA system to get the answers.

Making OBDA work efficiently over large amounts of data, requires that query answering over the ontology is *first-order (FO)-rewritable*[1] (Calvanese et al. 2007; Artale et al. 2009), which in turn limits the expressiveness of the ontology language, and the degree of detail with which the domain of interest can be captured. The current language of choice for OBDA is *DL-Lite*$_\mathcal{R}$, the logic underlying OWL 2 QL (Motik et al. 2009), which has been specifically designed to ensure FO-rewritability of query answering. Hence, it does not allow one to express disjunctive information, or any form of recursion on the data (e.g., as resulting from qualified existentials on the left-hand side of concept inclusions), since using such constructs in general causes the loss of FO-rewritability (Calvanese et al. 2013). For this reason, in many situations the expressive power of *DL-Lite*$_\mathcal{R}$ is too restricted to capture real-world scenarios; e.g., the axiom in our example is not expressible in *DL-Lite*$_\mathcal{R}$.

The aim of this work is to overcome these limitations of *DL-Lite*$_\mathcal{R}$ by allowing the use of additional constructs in the ontology. To be able to exploit the added value coming from OBDA in real-world settings, an important requirement is the efficiency of query answering, achieved through a rewriting-based approach. This is only possible for ontology languages that are FO-rewritable. Two general mechanisms that have been proposed to cope with computational complexity coming from high expressiveness of ontology languages, and that allow one to regain FO-rewritability, are conservative rewriting (Lutz, Piro, and

[1] Recall that FO queries constitute the core of SQL.

Wolter 2011) and approximation (Ren, Pan, and Zhao 2010; Console et al. 2014). Given an ontology in a powerful language, in the former approach it is rewritten, when possible, into an equivalent one in a restricted language, while in the latter it is approximated, thus losing part of its semantics.

In this work, we significantly extend the practical impact of both approaches by bringing into the picture *the mapping*, an essential component of OBDA that has been ignored so far. Indeed, it is a fairly expressive component of an OBDA system, since it allows one to make use of arbitrary SQL (hence FO) queries to relate the content of the data source to the elements of the ontology. Hence, a natural question is how one can use the mapping component to capture as much as possible additional domain semantics, resulting in better approximations or more cases where conservative rewritings are possible, while maintaining a *DL-Lite*$_\mathcal{R}$ ontology.

We illustrate how this can be done on our running example, where the non-*DL-Lite*$_\mathcal{R}$ axiom can be encoded by adding the assertion $sql_1(x) \bowtie sql_2(x,y) \bowtie sql_3(y) \rightsquigarrow \mathsf{SAcc}(x)$ to the mapping. This assertion connects $\mathcal{D}$ directly to the ontology term SAcc by making use of a join of the SQL queries in the original mapping. We observe that the resulting mapping, together with the ontology in which the non-*DL-Lite*$_\mathcal{R}$ axiom has been removed, constitutes a conservative rewriting of the original OBDA specification.

In this paper, we elaborate on this idea, by introducing a novel *framework for rewriting and approximation of OBDA specifications*. Specifically, we provide a notion of *rewriting* based on *query inseparability* of OBDA specifications (Bienvenu and Rosati 2015). To deal with those cases where it is not possible to rewrite the OBDA specification into a query inseparable one whose ontology is in *DL-Lite*$_\mathcal{R}$, we give a notion of *approximation* that is sound for query answering. We develop techniques for rewriting and approximation of OBDA specifications based on compiling the extra expressiveness into the mappings. We target rather expressive ontology languages, and for Horn-$\mathcal{ALCHIQ}$, a Horn fragment of OWL 2, we study decidability of existence of OBDA rewritings, and techniques to compute them when they exist, and to approximate them, otherwise.

We have implemented our techniques in a prototype system called ONTOPROX, which exploits functionalities provided by the ONTOP (Rodriguez-Muro, Kontchakov, and Zakharyaschev 2013) and CLIPPER systems (Eiter et al. 2012) to rewrite or approximate an OBDA specification expressed in Horn-$\mathcal{SHIQ}$ to one that can be directly processed by any OBDA system. We have evaluated ONTOPROX over synthetic and real OBDA instances against *(i)* the default ONTOP behavior, *(ii)* local semantic approximation (LSA), *(iii)* global semantic approximation (GSA), and *(iv)* CLIPPER over materialized ABoxes. We observe that using ONTOPROX, for a few queries we have been able to obtain more answers (in fact, complete answers, as confirmed by CLIPPER). However, for many queries ONTOPROX showed no difference with respect to the default ONTOP behavior. One reason for this is that in the considered real-world scenario, the mapping designers put significant effort to manually create complex mappings that overcome the limitations of *DL-Lite*$_\mathcal{R}$. Essentially they followed the principle of the technique presented here, and therefore produced an OBDA specification that was already "complete" by design.

The observations above immediately suggest a significant practical value of our approach, which can be used to facilitate the design of new OBDA specifications for existing expressive ontologies: instead of a manual compilation, which is cumbersome, error-prone, and difficult to maintain, mapping designers can write straightforward mappings, and the resulting OBDA specification can then be automatically transformed into a *DL-Lite*$_\mathcal{R}$ OBDA specification with rich mappings.

The paper is structured as follows. In Section 2, we provide some preliminary notions, and in Section 3, we present our framework of OBDA rewriting and approximation. In Section 4, we illustrate a technique for computing the OBDA-rewriting of a given Horn-$\mathcal{ALCHIQ}$ specification. In Section 5, we address the problem of OBDA-rewritability, and show how to obtain an approximation when a rewriting does not exist. In Section 6, we discuss our prototype ONTOPROX and experiments. Finally, in Section 7, we conclude the paper. The omitted proofs can be found in the appendix.

2 Preliminaries

We give some basic notions about ontologies and OBDA.

2.1 Ontologies

We assume to have the following pairwise disjoint countably infinite alphabets: $\mathsf{N_C}$ of *concept names*, $\mathsf{N_R}$ of *role names*, and $\mathsf{N_I}$ of constants (also called *individuals*). We consider ontologies expressed in Description Logics (DLs). Here we present the logics Horn-$\mathcal{ALCHIQ}$, the Horn fragment of $\mathcal{SHIQ}$ without role transitivity, and *DL-Lite*$_\mathcal{R}$, for which we develop some of the technical results in the paper. However, the general approximation framework is applicable to any fragment of OWL 2.

A Horn-$\mathcal{ALCHIQ}$ TBox in normal form is a finite set of axioms: *concept inclusions (CIs)* $\bigsqcap_i A_i \sqsubseteq C$, *role inclusions (RIs)* $R_1 \sqsubseteq R_2$ and *role disjointness* axioms $R_1 \sqcap R_2 \sqsubseteq \bot$, where A, A_i denote concept names, R, R_1, R_2 denote role names P or their inverses P^-, and C denotes a concept of the form $\bot$, A, $\exists R.A$, $\forall R.A$, or $\leq 1\, R.A$ (Kazakov 2009). For an inverse role $R = P^-$, we use R^- to denote P. $\bot$ denotes the empty concept/role. A *DL-Lite*$_\mathcal{R}$ TBox is a finite set of axioms of the form $B_1 \sqsubseteq B_2$, $B_1 \sqcap B_2 \sqsubseteq \bot$, $R_1 \sqsubseteq R_2$, and $R_1 \sqcap R_2 \sqsubseteq \bot$, where B_i denotes a concept of the form A or $\exists R.\top$. In what follows, for simplicity we write $\exists R$ instead of $\exists R.\top$, and we use N to denote either a concept or a role name. We also assume that all TBoxes are in normal form.

An *ABox* is a finite set of *membership assertions* of the form $A(c)$ or $P(c, c')$, where $c, c' \in \mathsf{N_I}$. For a DL $\mathcal{L}$, an *$\mathcal{L}$-ontology* is a pair $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ is an $\mathcal{L}$-TBox and $\mathcal{A}$ is an ABox. A *signature* Σ is a finite set of concept and role names. An ontology $\mathcal{O}$ is said to be defined over (or simply, *over*) Σ if all the concept and role names occurring in it belong to Σ (and likewise for TBoxes, ABoxes, concept inclusions, etc.). When $\mathcal{T}$ is over Σ, we denote by $\mathsf{sig}(\mathcal{T})$ the subset of Σ actually occurring in $\mathcal{T}$. Moreover we denote with $\mathsf{Ind}(\mathcal{A})$, the set of individuals appearing in $\mathcal{A}$.

The semantics, models, and the notions of satisfaction and consistency of ontologies are defined in the standard way. We only point out that we adopt the *Unique Name Assumption* (UNA), and for simplicity we also assume to have *standard names*, i.e., for every interpretation $\mathcal{I}$ and every constant $c \in \mathsf{N_I}$ interpreted by $\mathcal{I}$, we have that $c^{\mathcal{I}} = c$.

2.2 OBDA and Mappings

Let $\mathcal{S}$ be a relational schema over a countably infinite set $\mathsf{N_S}$ of database predicates. For simplicity, we assume to deal with plain relational schemas without constraints, and with database instances that directly store abstract objects (as opposed to values). In other words, a database instance $\mathcal{D}$ of $\mathcal{S}$ is a set of ground atoms over the predicates in $\mathsf{N_S}$ and the constants in $\mathsf{N_I}$.[2] Queries over $\mathcal{S}$ are expressed in SQL. We use $\varphi(\vec{x})$ to denote that query φ has $\vec{x} = x_1, \ldots, x_n$ as *free* (i.e., answer) variables, where n is the arity of φ. Given a database instance $\mathcal{D}$ of $\mathcal{S}$ and a query φ over $\mathcal{S}$, $ans(\varphi, \mathcal{D})$ denotes the set of tuples of constants in $\mathsf{N_I}$ computed by evaluating φ over $\mathcal{D}$.

In OBDA, one provides access to an (external) database through an ontology TBox, which is connected to the database by means of a mapping. Given a source schema $\mathcal{S}$ and a TBox $\mathcal{T}$, a (GAV) *mapping assertion* between $\mathcal{S}$ and $\mathcal{T}$ has the form $\varphi(x) \rightsquigarrow A(x)$ or $\varphi'(x, x') \rightsquigarrow P(x, x')$, where A and P are respectively concept and role names, and $\varphi(x)$, $\varphi'(x, x')$ are arbitrary (SQL) queries expressed over $\mathcal{S}$. Intuitively, given a database instance $\mathcal{D}$ of $\mathcal{S}$ and a mapping assertion $m = \varphi(x) \rightsquigarrow A(x)$, the instances of the concept A generated by m from $\mathcal{D}$ is the set $ans(\varphi, \mathcal{D})$; similarly for a mapping assertion $\varphi(x, x') \rightsquigarrow P(x, x')$.

An *OBDA specification* is a triple $\mathcal{P} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$, where $\mathcal{T}$ is a DL TBox, $\mathcal{S}$ is a relational schema, and $\mathcal{M}$ is a finite set of mapping assertions. Without loss of generality, we assume that all concept and role names appearing in $\mathcal{M}$ are contained in $\mathsf{sig}(\mathcal{T})$. An *OBDA instance* is a pair $\langle \mathcal{P}, \mathcal{D} \rangle$, where $\mathcal{P}$ is an OBDA specification, and $\mathcal{D}$ is a database instance of $\mathcal{S}$. The semantics of the OBDA instance $\langle \mathcal{P}, \mathcal{D} \rangle$ is specified in terms of interpretations of the concepts and roles in $\mathcal{T}$. We define it by relying on the following (*virtual*[3]) ABox

$$\mathcal{A}_{\mathcal{M},\mathcal{D}} = \{N(\vec{o}) \mid \vec{o} \in ans(\varphi, \mathcal{D}) \text{ and } \varphi(\vec{x}) \rightsquigarrow N(\vec{x}) \text{ in} \mathcal{M}\}$$

generated by $\mathcal{M}$ from $\mathcal{D}$, where N is a concept or role name in $\mathcal{T}$. Then, a model of $\langle \mathcal{P}, \mathcal{D} \rangle$ is simply a model of the ontology $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},\mathcal{D}} \rangle$.

Following Di Pinto et al. (2013), we split each mapping assertion $m = \varphi(\vec{x}) \rightsquigarrow N(\vec{x})$ in $\mathcal{M}$ into two parts by introducing an intermediate view name V_m for the SQL query $\varphi(\vec{x})$. We obtain a *low-level* mapping assertion of the form $\varphi(\vec{x}) \rightsquigarrow V_m(\vec{x})$, and a *high-level* mapping assertion of the form $V_m(\vec{x}) \rightsquigarrow N(\vec{x})$. In our technical development, we deal only with the high-level mappings. Hence, we abstract away the low-level mapping part, and in the following we directly consider the intermediate views as our data sources.

[2]All our results easily extend to the case where objects are constructed from retrieved database values (Calvanese et al. 2009).

[3]We call such an ABox 'virtual', because we are not interested in actually materializing its facts.

2.3 Query Answering

We consider conjunctive queries, which are the basic and most important querying mechanism in relational database systems and ontologies. A *conjunctive query* (CQ) $q(\vec{x})$ over a signature Σ is a formula $\exists \vec{y}.\, \varphi(\vec{x}, \vec{y})$, where φ is a conjunction of atoms $N(\vec{z})$, such that N is a concept or role name in Σ, and $\vec{z}$ are variables from $\vec{x}$ and $\vec{y}$. The set of *certain answers* to a CQ $q(\vec{x})$ over an ontology $\langle \mathcal{T}, \mathcal{A} \rangle$, denoted $cert(q, \langle \mathcal{T}, \mathcal{A} \rangle)$, is the set of tuples $\vec{c}$ of elements from $\mathsf{Ind}(\mathcal{A})$ of the same length as $\vec{x}$, such that $q(\vec{c})$ (considered as a FO sentence) holds in every model of $\langle \mathcal{T}, \mathcal{A} \rangle$. We mention two more query classes. An *atomic query* (AQ) is a CQ consisting of exactly one atom whose variables are all free. A *CQ with inequalities (CQ$^{\neq}$)* is a CQ that may contain inequality atoms between the variables of the predicate atoms.

Given a CQ q, an OBDA specification $\mathcal{P} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ and a database instance $\mathcal{D}$ of $\mathcal{S}$, the answer to q over the OBDA instance $\langle \mathcal{P}, \mathcal{D} \rangle$, denoted $cert(q, \mathcal{P}, \mathcal{D})$, is defined as $cert(q, \langle \mathcal{T}, \mathcal{A}_{\mathcal{M},\mathcal{D}} \rangle)$. Observe that, when $\mathcal{D}$ is inconsistent with $\mathcal{P}$ (i.e., $\langle \mathcal{P}, \mathcal{D} \rangle$ does not have a model), then $cert(q, \mathcal{P}, \mathcal{D})$ is the set of all possible tuples of constants in $\mathcal{A}_{\mathcal{M},\mathcal{D}}$ (of the same arity as q).

3 An OBDA Rewriting Framework

We extend the notion of query inseparability of ontologies (Botoeva et al. 2014) to OBDA specifications. We adopt the proposal by Bienvenu and Rosati (2015), but we do not enforce preservation of inconsistency.

Definition 1. *Let Σ be a signature. Two OBDA specifications $\mathcal{P}_1 = \langle \mathcal{T}_1, \mathcal{M}_1, \mathcal{S} \rangle$ and $\mathcal{P}_2 = \langle \mathcal{T}_2, \mathcal{M}_2, \mathcal{S} \rangle$ are* Σ-CQ inseparable *if $cert(q, \mathcal{P}_1, \mathcal{D}) = cert(q, \mathcal{P}_2, \mathcal{D})$, for every CQ q over Σ and every database instance $\mathcal{D}$ of $\mathcal{S}$.*

In OBDA, one must deal with the trade-off between the computational complexity of query answering and the expressiveness of the ontology language. Suppose that for an OBDA specification $\mathcal{P} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$, $\mathcal{T}$ is expressed in an ontology language $\mathcal{L}$ that does not allow for efficient query answering. A possible solution is to exploit the expressive power of the mapping layer to compute a new OBDA specification $\mathcal{P}' = \langle \mathcal{T}', \mathcal{M}', \mathcal{S} \rangle$ in which $\mathcal{T}'$ is expressed in a language $\mathcal{L}_t$ more suitable for query answering than $\mathcal{L}$. The aim is to encode in $\mathcal{M}'$ not only $\mathcal{M}$ but also part of the semantics of $\mathcal{T}$, so that $\mathcal{P}'$ is query-inseparable from $\mathcal{P}$. This leads to the notion of rewriting of OBDA specifications.

Definition 2. *Let $\mathcal{L}_t$ be an ontology language. The OBDA specification $\mathcal{P}' = \langle \mathcal{T}', \mathcal{M}', \mathcal{S} \rangle$ is a* CQ-rewriting *in $\mathcal{L}_t$ of the OBDA specification $\mathcal{P} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ if (i) $\mathsf{sig}(\mathcal{T}) \subseteq \mathsf{sig}(\mathcal{T}')$, (ii) $\mathcal{T}'$ is an $\mathcal{L}_t$-TBox, and (iii) $\mathcal{P}$ and $\mathcal{P}'$ are Σ-CQ inseparable, for $\Sigma = \mathsf{sig}(\mathcal{T})$. If such $\mathcal{P}'$ exists, we say that $\mathcal{P}$ is* CQ-rewritable into $\mathcal{L}_t$.

We observe that the new OBDA specification can be defined over a signature that is an extension of that of the original TBox. This is specified by condition *(i)*. In condition *(ii)*, we impose that the new ontology is specified in the target language $\mathcal{L}_t$. Finally, condition *(iii)* imposes that the OBDA specifications cannot be distinguished by CQs over the original TBox. Note that the definition allows for changing the ontology and the mappings, but not the source schema, accounting for the fact that the data sources might not be under the control of the designer of the OBDA specification.

As expected, it is not always possible to obtain a CQ-rewriting of $\mathcal{P}$ in an ontology language $\mathcal{L}_t$ that allows for efficient query answering. Indeed, the combined expressiveness of $\mathcal{L}_t$ with the new mappings might not be sufficient to simulate query answering over $\mathcal{P}$ without loss. In these cases, we can resort to approximating query answers over $\mathcal{P}$ in a *sound* way, which means that the answers to queries posed over the new specification are contained in those produced by querying $\mathcal{P}$. Hence, we say that the OBDA specification $\mathcal{P}' = \langle \mathcal{T}', \mathcal{M}', \mathcal{S}\rangle$ is a *sound CQ-approximation* in $\mathcal{L}_t$ of the OBDA specification $\mathcal{P} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ if $\mathcal{P}'$ satisfies *(i)*, *(ii)*, and $cert(q, \mathcal{P}', \mathcal{D}) \subseteq cert(q, \mathcal{P}, \mathcal{D})$, for each CQ q over $\mathsf{sig}(\mathcal{T})$ and for each instance $\mathcal{D}$ of $\mathcal{S}$.

Next, we study CQ-rewritability of OBDA specifications into *DL-Lite*$_\mathcal{R}$, developing suitable techniques.

4 Rewriting OBDA Specifications

In this section, we develop our OBDA rewriting technique, which relies on Datalog rewritings of the TBox (and mappings). Recall that a *Datalog program* (with inequalities) is a finite set of definite *Horn* clauses *without* functions symbols, i.e., rules of the form $head \leftarrow \varphi$, where φ is a finite non-empty list of predicate atoms and guarded inequalities called the body of the rule, and $head$ is an atom, called the head of the rule, all of whose variables occur in the body. The predicates that occur in rule heads are called *intensional* (IDB), the other predicates are called *extensional* (EDB).

4.1 ET-mappings

Now, we extend the notion of T-mappings introduced by Rodriguez-Muro, Kontchakov, and Zakharyaschev (2013), and define the notion of an ET-mapping that results from compiling into the mapping the expressiveness of ontology languages that are Datalog rewritable, as introduced below.

We first introduce notation we need. Let Π be a Datalog program and N an IDB predicate. For a database $\mathcal{D}$ over the EDB predicates of Π, let $N^i_\Pi(\mathcal{D})$ denote the set of facts about N that can be deduced from $\mathcal{D}$ by at most $i \geq 1$ applications of the rules in Π, and let $N^\infty_\Pi(\mathcal{D}) = \bigcup_{i\geq 1} N^i_\Pi(\mathcal{D})$. It is known that the predicate $N^\infty_\Pi(\cdot)$ defined by N in Π can be characterized by a possibly infinite union of CQ$^{\neq}$s (Cosmadakis et al. 1988), i.e., there exist CQ$^{\neq}$s $\varphi^N_0, \varphi^N_1, \ldots$ such that $N^\infty_\Pi(\mathcal{D}) = \bigcup_{i\geq 0}\{N(\vec{a}) \mid \vec{a} \in ans(\varphi^N_i, \mathcal{D})\}$, for every $\mathcal{D}$. The φ^N_i's are called the *expansions* of N and can be described in terms of expansion trees(see Appendix A.1). We denote by $\Phi_\Pi(N)$ the set of expansion trees for N in Π, and abusing notation also the (possibly infinite) union of CQ$^{\neq}$s corresponding to it. Note that $\Phi_\Pi(N)$ might be infinite due to the presence of IDB predicates that are *recursive*, i.e., either directly or indirectly refer to themselves.

We call a TBox $\mathcal{T}$ *Datalog rewritable* if it admits a translation $\Pi_\mathcal{T}$ to Datalog that preserves consistency and answers to AQs (see, e.g., the translations by Hustadt, Motik, and Sattler (2005), Eiter et al. (2012), and Trivela et al. (2015) for Horn-$\mathcal{SHIQ}$, and by Cuenca Grau et al. (2013) for $\mathcal{SHI}$). We assume that $\Pi_\mathcal{T}$ makes use of a special nullary predicate $\perp$ that encodes inconsistency, i.e., for an ABox $\mathcal{A}$, $\langle \mathcal{T}, \mathcal{A}\rangle$ is consistent iff $\perp^\infty_{\Pi_\mathcal{T}}(\mathcal{A})$ is empty.[4] We also assume that $\Pi_\mathcal{T}$ includes the following auxiliary rules, which ensure that $\Pi_\mathcal{T}$ derives all possible facts constructed over $\mathsf{sig}(\mathcal{T})$ and $\mathsf{Ind}(\mathcal{A})$ whenever $\langle \mathcal{T}, \mathcal{A}\rangle$ is inconsistent:

$$\top_\Delta(x) \leftarrow A(x);\ \top_\Delta(x) \leftarrow P(x,y);\ \top_\Delta(y) \leftarrow P(x,y);$$
$$A(x) \leftarrow \perp, \top_\Delta(x);\quad P(x,y) \leftarrow \perp, \top_\Delta(x), \top_\Delta(y);$$

where A and P respectively range over concept and role names in $\mathsf{sig}(\mathcal{T})$, and $\top_\Delta$ is a fresh unary predicate denoting the set of all the individuals appearing in $\mathcal{A}$.

In the following, we denote with $\Pi_\mathcal{M}$ the (high-level) mapping $\mathcal{M}$ viewed as a Datalog program, and with $\Pi_{\mathcal{T},\mathcal{M}}$ the Datalog program $\Pi_\mathcal{T} \cup \Pi_\mathcal{M}$ *associated to* a Datalog rewritable TBox $\mathcal{T}$ and a mapping $\mathcal{M}$. From the properties of the translation $\Pi_\mathcal{T}$ (and the simple structure of $\Pi_\mathcal{M}$), we obtain that $\Pi_{\mathcal{T},\mathcal{M}}$ satisfies the following:

Lemma 3. *Let $\langle \mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ be an OBDA specification where $\mathcal{T}$ is Datalog rewritable. Then, for every database instance $\mathcal{D}$ of $\mathcal{S}$, concept or role name N of $\mathcal{T}$, and $\vec{a}$ in $\mathsf{Ind}(\mathcal{A}_{\mathcal{M},\mathcal{D}})$, we have that $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},\mathcal{D}}\rangle \models N(\vec{a})$ iff $N(\vec{a}) \in N^\infty_{\Pi_{\mathcal{T},\mathcal{M}}}(\mathcal{D})$.*

For a predicate N, we say that an expansion $\varphi^N \in \Phi_{\Pi_{\mathcal{T},\mathcal{M}}}(N)$ is *DB-defined* if φ^N is defined over database predicates. Now we are ready to define ET-mappings.

Definition 4. *Let $\langle \mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ be an OBDA specification where $\mathcal{T}$ is Datalog rewritable. The* ET-mapping *for $\mathcal{M}$ and $\mathcal{T}$, denoted $\mathsf{etm}_\mathcal{T}(\mathcal{M})$, is defined as the set of assertions of the form $\varphi^N(\vec{x}) \rightsquigarrow N(\vec{x})$ such that N is a concept or role name in $\mathcal{T}$, and $\varphi^N \in \Phi_{\Pi_{\mathcal{T},\mathcal{M}}}(N)$ is DB-defined.*

It is easy to show that, for $\mathcal{M}' = \mathsf{etm}_\mathcal{T}(\mathcal{M})$ and each database instance $\mathcal{D}$, the virtual ABox $\mathcal{A}_{\mathcal{M}',\mathcal{D}}$ (which can be defined for ET-mappings as for ordinary mappings) contains all facts entailed by $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},\mathcal{D}}\rangle$. In this sense, the ET-mapping $\mathsf{etm}_\mathcal{T}(\mathcal{M})$ plays for a Datalog rewritable TBox $\mathcal{T}$ the same role as T-mappings play for (the simpler) *DL-Lite*$_\mathcal{R}$ TBoxes. Note that, in general, an ET-mapping is not a mapping, as it may contain infinitely many assertions. However, $\mathcal{A}_{\mathcal{M}',\mathcal{D}}$ is still finite, given that it is constructed over the finite number of constants appearing in $\mathcal{D}$.

4.2 Rewriting Horn-$\mathcal{ALCHIQ}$ OBDA Specifications to *DL-Lite*$_\mathcal{R}$

Let $\langle \mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ be an OBDA specification, where $\mathcal{T}$ is a Horn-$\mathcal{ALCHIQ}$ TBox over a signature Σ. Figure 1 describes the algorithm $\mathsf{RewObda}(\mathcal{T}, \mathcal{M})$, which constructs a *DL-Lite*$_\mathcal{R}$ TBox $\mathcal{T}_r$ and an ET-mapping $\mathcal{M}_c$ such that $\langle \mathcal{T}_r, \mathcal{M}_c, \mathcal{S}\rangle$ is Σ-CQ inseparable from $\langle \mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$.

In Step 2, the algorithm applies to $\mathcal{T}_1$ the normalization procedure $\mathsf{norm}_\exists$, which gets rid of concepts of the form $\exists R.(\bigsqcap A'_j)$ in the right-hand side of CIs. This is achieved by the following well-known substitution (Artale et al. 2009): every CI $\bigsqcap^m_{i=1} A_i \sqsubseteq \exists R.(\bigsqcap^n_{j=1} A'_j)$ in $\mathcal{T}_1$ is replaced with $\bigsqcap^m_{i=1} A_i \sqsubseteq \exists P_{new}$, $P_{new} \sqsubseteq R$, and $\top \sqsubseteq \forall P_{new}.A'_j$, for $1 \leq j \leq n$, where P_{new} is a fresh role name. Notice that the latter

[4] Here we simply consider $\mathcal{A}$ as a database.

Input: Horn-$\mathcal{ALCHIQ}$ TBox $\mathcal{T}$ and mapping $\mathcal{M}$.
Output: *DL-Lite*$_\mathcal{R}$ TBox $\mathcal{T}_r$ and ET-mapping $\mathcal{M}_c$.

Step 1: $\mathcal{T}_1$ is obtained from $\mathcal{T}$ by adding all CIs of the form $\bigsqcap A_i \sqsubseteq \exists R.(\bigsqcap A'_j)$ entailed by $\mathcal{T}$, for concept names $A_i, A'_j \in \mathsf{sig}(\mathcal{T})$.

Step 2: $\mathcal{T}_2 = \mathsf{norm}_\exists(\mathcal{T}_1)$.

Step 3: $\mathcal{T}_3 = \mathsf{norm}_\sqcap(\mathcal{T}_2)$.

Step 4: $\mathcal{M}_c$ is $\mathsf{etm}_{\mathcal{T}_3}(\mathcal{M})$, and $\mathcal{T}_r$ is the *DL-Lite*$_\mathcal{R}$ TBox consisting of all *DL-Lite*$_\mathcal{R}$ axioms over $\mathsf{sig}(\mathcal{T}_3)$ entailed by $\mathcal{T}_3$ (including the trivial ones $N \sqsubseteq N$).

Figure 1: OBDA specification rewriting algorithm RewObda.

two forms of inclusions introduced by $\mathsf{norm}_\exists$ are actually in *DL-Lite*$_\mathcal{R}$, as $\top \sqsubseteq \forall P_{new}.A'_j$ is equivalent to $\exists P^-_{new} \sqsubseteq A'_j$. In Step 3, the algorithm applies to $\mathcal{T}_2$ a further normalization procedure, $\mathsf{norm}_\sqcap$, which introduces a fresh concept name $A_{A_1\sqcap\cdots\sqcap A_n}$ for each concept conjunction $A_1 \sqcap \cdots \sqcap A_n$ appearing in $\mathcal{T}_2$, and adds $A_1 \sqcap \cdots \sqcap A_n \equiv A_{A_1\sqcap\cdots\sqcap A_n}$[5] to the TBox. Note that $\mathsf{norm}_\exists(\mathcal{T}_1)$ and $\mathsf{norm}_\sqcap(\mathcal{T}_2)$ are model-conservative extensions of $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively (Lutz, Walther, and Wolter 2007), as one can easily show. We denote by $\mathsf{rew}(\mathcal{T})$ the resulting TBox $\mathcal{T}_r$, which in general is exponential in the size of $\mathcal{T}$, and by $\mathsf{comp}(\mathcal{T},\mathcal{M})$ the resulting ET-mapping $\mathcal{M}_c$, which in general is infinite.

Example 5. Assume that the domain knowledge is represented by the axiom about bank accounts from Section 1. The normalization of this axiom is the TBox $\mathcal{T}^b = \{\mathsf{Person} \sqsubseteq \forall \mathsf{inNameOf}^-.A_1, \mathsf{CAcc} \sqcap A_1 \sqsubseteq \mathsf{SAcc}\}$. Assume that the database schema $\mathcal{S}^b$ consists of the two relations $\mathsf{ENT}(\mathtt{ID}, \mathtt{TYPE}, \mathtt{EMPID})$, $\mathsf{PROD}(\mathtt{NUM}, \mathtt{TYPE}, \mathtt{CUSTID})$, whose data are mapped to the ontology terms by means of the following mapping $\mathcal{M}$:

m_P: `SELECT ID AS X FROM ENT WHERE ENT.TYPE='P'` $\rightsquigarrow$ $\mathsf{Person}(\mathtt{X})$
m_N: `SELECT NUM AS X, CUSTID AS Y FROM PROD` $\rightsquigarrow$ $\mathsf{inNameOf}(\mathtt{X},\mathtt{Y})$
m_C: `SELECT NUM AS X FROM PROD P WHERE P.TYPE='B'` $\rightsquigarrow$ $\mathsf{CAcc}(\mathtt{X})$

We will work with the corresponding high-level mapping $\mathcal{M}^b$ consisting of the assertions:

$$\begin{aligned} h_\mathsf{P}: &\quad \{x \mid V_\mathsf{Person}(x)\} \rightsquigarrow \mathsf{Person}(x)\\ h_\mathsf{N}: &\quad \{x,y \mid V_\mathsf{inNameOf}(x,y)\} \rightsquigarrow \mathsf{inNameOf}(x,y)\\ h_\mathsf{C}: &\quad \{x \mid V_\mathsf{CAcc}(x)\} \rightsquigarrow \mathsf{CAcc}(x)\end{aligned}$$

Now, consider the OBDA specification $\mathcal{P}^b = \langle \mathcal{T}^b, \mathcal{M}^b, \mathcal{S}^b\rangle$. The RewObda algorithm invoked on $(\mathcal{T}^b, \mathcal{M}^b)$ produces:

- The intermediate TBoxes $\mathcal{T}^b_1$ and $\mathcal{T}^b_2$ coinciding with $\mathcal{T}^b$, and $\mathcal{T}^b_3$ extending $\mathcal{T}^b$ with $A_{\mathsf{CAcc}\sqcap A_1} \equiv \mathsf{CAcc} \sqcap A_1$.
- The ET-mapping $\mathcal{M}^b_c = \mathsf{etm}_{\mathcal{T}^b_3}(\mathcal{M}^b)$, which extends $\mathcal{M}^b$ with the assertions $\{x \mid V_\mathsf{inNameOf}(x,y), V_\mathsf{Person}(y)\} \rightsquigarrow A_1(x)$, $\{x \mid V_\mathsf{CAcc}(x), V_\mathsf{inNameOf}(x,y), V_\mathsf{Person}(y)\} \rightsquigarrow \mathsf{SAcc}(x)$, and $\{x \mid V_\mathsf{CAcc}(x), V_\mathsf{inNameOf}(x,y), V_\mathsf{Person}(y)\} \rightsquigarrow A_{\mathsf{CAcc}\sqcap A_1}(x)$.

The algorithm returns the *DL-Lite*$_\mathcal{R}$ TBox $\mathcal{T}^b_r = \{A_{\mathsf{CAcc}\sqcap A_1} \sqsubseteq \mathsf{CAcc}, A_{\mathsf{CAcc}\sqcap A_1} \sqsubseteq A_1, A_{\mathsf{CAcc}\sqcap A_1} \sqsubseteq \mathsf{SAcc}\}$ and the mapping $\mathcal{M}^b_c$. It is possible to show that $\mathcal{P}^b_{DL\text{-}Lite_\mathcal{R}} = \langle \mathcal{T}^b_r, \mathcal{M}^b_c, \mathcal{S}^b\rangle$ is a CQ-rewriting of $\mathcal{P}^b$ into *DL-Lite*$_\mathcal{R}$. ∎

[5]We use '$\equiv$' to abbreviate inclusion in both directions.

The TBox $\mathcal{T}_3$ obtained as an intermediate result in Step 3 of $\mathsf{RewObda}(\mathcal{T},\mathcal{M})$, is a model-conservative extension of $\mathcal{T}$ that is tailored towards capturing in *DL-Lite*$_\mathcal{R}$ the answers to tree-shaped CQs. This is obtained by introducing in Step 2 sufficiently new role names, and in Step 3 new concept names, so as to capture entailed axioms that generate the tree-shaped parts of models. On the other hand, the ET-mapping $\mathcal{M}_c = \mathsf{comp}(\mathcal{T},\mathcal{M})$ is such that it generates from a database instance a virtual ABox $\mathcal{A}^v$ that is complete with respect to all ABox facts that might be involved in the generation of the tree-shaped parts of models of $\mathcal{T}_r$ and $\mathcal{A}^v$. This allows us to prove the main result of this section.

Theorem 6. *Let $\langle \mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ be an OBDA specification such that $\mathcal{T}$ is a Horn-$\mathcal{ALCHIQ}$ TBox, and let $\langle \mathcal{T}_r, \mathcal{M}_c\rangle = \mathsf{RewObda}(\mathcal{T},\mathcal{M})$. Then $\langle \mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ and $\langle \mathcal{T}_r, \mathcal{M}_c, \mathcal{S}\rangle$ are Σ-CQ inseparable, for $\Sigma = \mathsf{sig}(\mathcal{T})$.*

Clearly, $\langle \mathcal{T}_r, \mathcal{M}_c, \mathcal{S}\rangle$ is a candidate for being a CQ-rewriting of $\langle \mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ into *DL-Lite*$_\mathcal{R}$. However, since $\mathcal{M}_c$ might be an infinite set, $\langle \mathcal{T}_r, \mathcal{M}_c, \mathcal{S}\rangle$ might not be an OBDA specification and hence might not be effectively usable for query answering. Next we address this issue, and show that in some cases we obtain proper CQ-rewritings, while in others we have to resort to approximations.

5 Approximating OBDA Specifications

To obtain from an ET-mapping a proper mapping, we exploit the notion of predicate boundedness in Datalog, and use a bound on the depth of Datalog expansion trees.

An IDB predicate N is said to be *bounded* in a Datalog program Π, if there exists a constant k depending only on Π such that, for every database $\mathcal{D}$, we have $N^k_\Pi(\mathcal{D}) = N^\infty_\Pi(\mathcal{D})$ (Cosmadakis et al. 1988). If N is bounded in Π, then there exists an equivalent Datalog program Π' such that $\Phi_{\Pi'}(N)$ is *finite*, and thus represents a finite union of CQ$^{\neq}$s. It is well known that predicate boundedness for Datalog is undecidable in general (Gaifman et al. 1987). We say that Ω is a *boundedness oracle* if for a Datalog program Π and a predicate N it returns one of the three answers: N is bounded in Π, N is not bounded in Π, or unknown. When N is bounded, Ω returns also a *finite* union of CQ$^{\neq}$s, denoted $\Omega_\Pi(N)$, defining N. Given a constant k, $\Phi^k_\Pi(N)$ denotes the set of trees (and the corresponding union of CQ$^{\neq}$s) in $\Phi_\Pi(N)$ of depth at most k, hence $\Phi^k_\Pi(N)$ is always finite.

We introduce a *cutting operator* cut^Ω_k, which is parametric with respect to the cutting depth $k > 0$ and the boundedness oracle Ω, which, when applied to a predicate N and a Datalog program Π, returns a finite union of CQ$^{\neq}$s as follows:

$$\mathsf{cut}^\Omega_k(N,\Pi) = \begin{cases} \Omega_\Pi(N), & \text{if } N \text{ is bounded in } \Pi \text{ w.r.t. } \Omega\\ \Phi^k_\Pi(N), & \text{otherwise.}\end{cases}$$

We apply cutting also to ET-mappings: given an ET-mapping $\mathsf{etm}_\mathcal{T}(\mathcal{M})$, the *mapping* $\mathsf{cut}^\Omega_k(\mathsf{etm}_\mathcal{T}(\mathcal{M}))$ is the (finite) set of mapping assertions $\varphi^N(\vec{x}) \rightsquigarrow N(\vec{x})$ s.t. N is a concept or role name in $\mathcal{T}$, and $\varphi^N \in \mathsf{cut}^\Omega_k(N, \Pi_{\mathcal{T},\mathcal{M}})$ is DB-defined.

The following theorem provides a sufficient condition for CQ-rewritability into *DL-Lite*$_\mathcal{R}$ in terms of the well-known notion of first-order (FO)-rewritability, which we recall here:

a query q is *FO-rewritable* with respect to a TBox $\mathcal{T}$, if there exists a FO query q' such that $cert(q, \langle\mathcal{T}, \mathcal{A}\rangle) = ans(q', \mathcal{A})$, for every ABox $\mathcal{A}$ over $\mathsf{sig}(\mathcal{T})$ (viewed as a database). It uses the fact that if an AQ is FO-rewritable with respect to a Horn-$\mathcal{ALCHIQ}$ TBox $\mathcal{T}$, then it is actually rewritable into a union of CQ$^{\neq}$s, and the fact that if $\mathcal{T}$ is FO-rewritable for AQs (i.e., every AQ is FO-rewritable with respect to $\mathcal{T}$), then each concept and role name is bounded in $\Pi_{\mathcal{T}}$ (Lutz and Wolter 2011; Bienvenu, Lutz, and Wolter 2013).

Theorem 7. *Let $\langle\mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ be an OBDA specification such that $\mathcal{T}$ is a Horn-$\mathcal{ALCHIQ}$ TBox. Further, let $\mathcal{T}_r = \mathsf{rew}(\mathcal{T})$ and $\mathcal{M}' = \mathsf{cut}_k^{\Omega}(\mathsf{comp}(\mathcal{T}, \mathcal{M}))$, for a boundedness oracle Ω and some $k > 0$. If $\mathcal{T}$ is FO-rewritable for AQs, then $\langle\mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ is* CQ-rewritable *into DL-Lite$_{\mathcal{R}}$, and $\langle\mathcal{T}_r, \mathcal{M}', \mathcal{S}\rangle$ is its* CQ-rewriting. *Otherwise, $\langle\mathcal{T}_r, \mathcal{M}', \mathcal{S}\rangle$ is a* sound CQ-approximation *of $\langle\mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ in DL-Lite$_{\mathcal{R}}$.*

The above result provides us with decidable conditions for rewritability of OBDA specifications in several significant cases. It is shown by Bienvenu, Lutz, and Wolter (2013) and Lutz and Wolter (2011) that FO-rewritability of AQs relative to Horn-$\mathcal{SHI}$-TBoxes, Horn-$\mathcal{ALCF}$-TBoxes, and Horn-$\mathcal{ALCIF}$-TBoxes of depth two is decidable. In fact, these FO-rewritability algorithms provide us with a boundedness oracle Ω: for each concept and role name N in $\mathcal{T}$, they return a FO-rewriting of the AQ $N(\vec{x})$ that combined with the mapping $\mathcal{M}$ results in $\Omega_{\Pi_{\mathcal{T},\mathcal{M}}}(N)$.

Unfortunately, a complete characterization of CQ-rewritability into *DL-Lite*$_{\mathcal{R}}$ is not possible if arbitrary FO-queries are allowed in the (low-level) mapping.

Theorem 8. *The problem of checking whether an OBDA specification with an $\mathcal{EL}$ ontology and FO source queries in the mapping is CQ-rewritable into DL-Lite$_{\mathcal{R}}$ is undecidable.*

However, if we admit only unions of CQs in the (low-level) mapping, we can fully characterize CQ-rewritability.

Theorem 9. *The problem of checking whether an OBDA specification with a Horn-$\mathcal{ALCHI}$ ontology of depth one and unions of CQs as source queries in the mapping is CQ-rewritable into DL-Lite$_{\mathcal{R}}$ is decidable.*

6 Implementation and Experiments

To demonstrate the feasibility of our OBDA specification rewriting technique, we have implemented a prototype system called ONTOPROX[6] and evaluated it over synthetic and real OBDA instances. Our system relies on the OBDA reasoner ONTOP[7] and the complete Horn-$\mathcal{SHIQ}$ CQ-answering system CLIPPER[8], used as Java libraries. ONTOPROX also relies on a standard Prolog engine (SWI-PROLOG[9]) and on an OWL 2 reasoner (HERMIT[10]).

Essentially, ONTOPROX implements the rewriting and compiling procedure described in Figure 1, but instead of computing the (possibly infinite) ET-mapping $\mathsf{comp}(\mathcal{T}, \mathcal{M})$, it computes its finite part $\mathsf{cut}_k(\mathsf{comp}(\mathcal{T}, \mathcal{M}))$. So, it gets as input an OWL 2 OBDA specification $\langle\mathcal{T}_{\text{OWL2}}, \mathcal{M}, \mathcal{S}\rangle$ and a positive integer k, and produces a *DL-Lite*$_{\mathcal{R}}$ OBDA specification that can be used with any OBDA system. Below we describe some of the implementation details:

(1) $\mathcal{T}_{\text{OWL2}}$ is first approximated to the Horn-$\mathcal{SHIQ}$ TBox $\mathcal{T}$ by dropping the axioms outside this fragment.

(2) $\mathcal{T}$ is translated into a (possibly recursive) Datalog program Π and saturated with all CIs of the form $\bigsqcap A_i \sqsubseteq \exists R.(\bigsqcap A'_j)$, using functionalities provided by CLIPPER.

(3) The expansions $\mathsf{cut}_k(\Phi_\Pi(X))$ are computed by an auxiliary Prolog program using Prolog meta-programming.

(4) To produce actual mappings that can be used by an OBDA reasoner, the views in the high-level mapping $\mathsf{cut}_k(\mathsf{comp}(\mathcal{T}, \mathcal{M}))$ are replaced with their original SQL definitions using functionalities of ONTOP.

(5) The *DL-Lite*$_{\mathcal{R}}$ closure is computed by relying on the OWL 2 reasoner for Horn-$\mathcal{SHIQ}$ TBox classification.

For the experiments, we have considered two scenarios:

UOBM. The university ontology benchmark (UOBM) (Ma et al. 2006) comes with a $\mathcal{SHOIN}$ ontology (with 69 concepts, 35 roles, 9 attributes, and 204 TBox axioms), and an ABox generator. We have designed a database schema for the generated ABox, converted the ABox to a 10MB database instance for the schema, and manually created the mapping, consisting of 96 assertions[11].

Among others, we have considered the following queries:

Q_1^u:
```
SELECT DISTINCT ?X WHERE
  { ?X a ub:Person . }
```
Q_2^u:
```
SELECT DISTINCT ?X WHERE
  { ?X a ub:Employee . }
```
Q_3^u:
```
SELECT DISTINCT ?X ?Y WHERE
  { ?X rdf:type ub:ResearchGroup .
    ?X ub:subOrganizationOf ?Y . }
```
Q_4^u:
```
SELECT DISTINCT ?X ?Y ?Z WHERE
  { ?X rdf:type ub:Chair .
    ?X ub:worksFor ?Y .
    ?Y rdf:type ub:Department .
    ?Y ub:subOrganizationOf ?Z . }
```

Telecom benchmark. The telecommunications ontology models a portion of the network of a leading telecommunications company, namely the portion connecting subscribers to the operating centers of their service providers. The current specification consists of an OWL 2 ontology with 152 concepts, 53 roles, 73 attributes, 458 TBox axioms, and of a mapping with 264 mapping assertions. The database instance contains 32GB of real-world data.

In the following, we only provide a description of some of the queries because the telecommunications ontology itself is bound by a confidentiality agreement.

- Query Q_1^t asks, for each cable in the telecommunications network, the single segments of which the cable is composed, and the network line (between two devices) that the

[6]https://github.com/ontop/ontoprox/
[7]http://ontop.inf.unibz.it/
[8]http://www.kr.tuwien.ac.at/research/systems/clipper/
[9]http://www.swi-prolog.org/
[10]http://hermit-reasoner.com/
[11]https://github.com/ontop/ontop-examples/tree/master/aaai-2016-ontoprox/uobm

Table 1: Query evaluation with respect to 5 setups (number of answers / running time in seconds)

		ONTOP	LSA	GSA	ONTOPROX	CLIPPER
UOBM	Q_1^u	14,129 / 0.08	14,197 / 0.11	14,197 / 0.43	14,197 / 0.42	14,197 / 21.4
	Q_2^u	1,105 / 0.09	2,170 / 0.15	2,170 / 0.42	2,170 / 0.44	2,170 / 21.3
	Q_3^u	235 / 0.20	235 / 0.24	235 / 0.88	247 / 0.83	247 / 19.6
	Q_4^u	19 / 0.13	19 / 0.15	19 / 0.43	38 / 0.52	38 / 21.4
Telecom	Q_1^t	0 / 2.91	0 / 0.72	0 / 1.91	82,455 / 5.21	N/A
	Q_2^t	0 / 0.72	0 / 0.21	0 / 0.67	16,487 / 198	N/A
	Q_3^t	5,201,363 / 128	5,201,363 / 105	5,201,363 / 538	5,260,346 / 437	N/A

Table 2: ONTOPROX pre-computation time and output size

	UOBM	Telecom
Time (s)	8.47	8.72
Number of mapping assertions	441	907
Number of TBox axioms	294	620
Number of new concepts	26	60
Number of new roles	30	7

cable covers. For each cable, it also returns its bandwidth and its status (functioning, non-functioning, etc.).

- Query Q_2^t asks for each path in the network that runs on fiber-optic cable, to return the specific device from which the path originates, and also requires to provide the number of different channels available in the path.
- Query Q_3^t asks, for each cable in the telecommunications network, the port to which the cable is attached, the slot on the device in which the port is installed, and, for each such slot, its status and its type. For each cable, it also returns its status.

For each OBDA instance $\langle\langle\mathcal{T},\mathcal{M},\mathcal{S}\rangle,\mathcal{D}\rangle$, we have evaluated the number of query answers and the query answering time with respect to five different setups:

(1) The default behavior of ONTOP v1.15, which simply ignores all non-*DL-Lite*$_\mathcal{R}$ axioms in $\mathcal{T}$, i.e., using $\langle\mathcal{T}^1,\mathcal{M},\mathcal{S}\rangle$ where $\mathcal{T}^1$ are all the *DL-Lite*$_\mathcal{R}$ axioms in $\mathcal{T}$.

(2) The local semantic approximation (LSA) of $\mathcal{T}$ in *DL-Lite*$_\mathcal{R}$, i.e., using $\langle\mathcal{T}^2,\mathcal{M},\mathcal{S}\rangle$ where $\mathcal{T}^2$ is obtained as the union, for each axiom $\alpha\in\mathcal{T}$, of the set of *DL-Lite*$_\mathcal{R}$ axioms $\Gamma(\alpha)$ entailed by α (Console et al. 2014).

(3) The global semantic approximation (GSA) of $\mathcal{T}$ in *DL-Lite*$_\mathcal{R}$, i.e., using $\langle\mathcal{T}^3,\mathcal{M},\mathcal{S}\rangle$ where $\mathcal{T}^3$ is the *DL-Lite*$_\mathcal{R}$ closure of $\mathcal{T}$ (Pan and Thomas 2007).

(4) Result of ONTOPROX, $\langle\mathsf{rew}(\mathcal{T}),\mathsf{cut}_5(\mathsf{comp}(\mathcal{T},\mathcal{M})),\mathcal{S}\rangle$.

(5) CLIPPER over the materialization of the virtual ABox.

In Table 1, we present details of the evaluation for some of the queries for which we obtained significant results. In Table 2, we provide statistics about the ONTOPROX pre-computations. The performed evaluation led to the following findings:

- For the considered set of queries LSA and GSA produce the same answers.
- Compared to the default ONTOP behavior, LSA/GSA produces more answers for 2 queries out of 4 for UOBM.
- ONTOPROX produces more answers than LSA/GSA for 2 queries out of 4 for UOBM, and for all Telecom queries. In particular, note that for Q_1^t and Q_2^t, LSA and GSA returned no answers at all.
- For UOBM, ONTOPROX answers are complete, as confirmed by the comparison with the results provided by CLIPPER. We cannot determine completeness for the Telecom queries, because the Telecom database was too large and its materialization in an ABox was not feasible.
- Query answering of ONTOPROX is ~3–5 times slower than ONTOP, when the result sets are of comparable size (note that for Q_2^t the result set is significantly larger).
- The size of the new *DL-Lite*$_\mathcal{R}$ OBDA specifications is comparable with that of the original specifications.

7 Conclusions

We proposed a novel framework for rewriting and approximation of OBDA specifications in an expressive ontology language to specifications in a weaker language, in which the core idea is to exploit the mapping layer to encode part of the semantics of the original OBDA specification, and we developed techniques for *DL-Lite*$_\mathcal{R}$ as the target language.

We plan to continue our work along the following directions: *(i)* extend our technique to Horn-$\mathcal{SHIQ}$, and, more generally, to Datalog rewritable TBoxes (Cuenca Grau et al. 2013); *(ii)* deepen our understanding of the computational complexity of deciding CQ-rewritability of OBDA specifications into *DL-Lite*$_\mathcal{R}$; *(iii)* extend our technique to SPARQL queries under different OWL entailment regimes (Kontchakov et al. 2014); *(iv)* carry out more extensive experiments, considering queries that contain existentially quantified variables. This will allow us to verify the effectiveness of RewObda, which was designed specifically to deal with existentially implied objects.

Acknowledgement. This paper is supported by the EU under the large-scale integrating project (IP) Optique (*Scalable End-user Access to Big Data*), grant agreement n. FP7-318338. We thank Martin Rezk for insightful discussions, and Benjamin Cogrel and Elem Güzel for help with the experimentation.

References

Artale, A.; Calvanese, D.; Kontchakov, R.; and Zakharyaschev, M. 2009. The *DL-Lite* family and relations. *J. of Artificial Intelligence Research* 36:1–69.

Bienvenu, M., and Rosati, R. 2015. Query-based comparison of OBDA specifications. In *Proc. of the 28th Int. Workshop on Description Logic (DL)*, volume 1350 of *CEUR Electronic Workshop Proceedings*.

Bienvenu, M.; Lutz, C.; and Wolter, F. 2013. First-order rewritability of atomic queries in Horn description logics. In *Proc. of the 23rd Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 754–760.

Botoeva, E.; Kontchakov, R.; Ryzhikov, V.; Wolter, F.; and Zakharyaschev, M. 2014. Query inseparability for description logic knowledge bases. In *Proc. of the 14th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR)*, 238–247. AAAI Press.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. of Automated Reasoning* 39(3):385–429.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; Poggi, A.; Rodriguez-Muro, M.; and Rosati, R. 2009. Ontologies and databases: The *DL-Lite* approach. In *5th Reasoning Web Int. Summer School Tutorial Lectures (RW)*, volume 5689 of *LNCS*. Springer. 255–356.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2013. Data complexity of query answering in description logics. *Artificial Intelligence* 195:335–360.

Chandra, A. K., and Merlin, P. M. 1977. Optimal implementation of conjunctive queries in relational data bases. In *Proc. of the 9th ACM Symp. on Theory of Computing (STOC)*, 77–90.

Console, M.; Mora, J.; Rosati, R.; Santarelli, V.; and Savo, D. F. 2014. Effective computation of maximal sound approximations of description logic ontologies. In *Proc. of the 13th Int. Semantic Web Conf. (ISWC)*, volume 8797 of *LNCS*, 164–179. Springer.

Cosmadakis, S. S.; Gaifman, H.; Kanellakis, P. C.; and Vardi, M. Y. 1988. Decidable optimization problems for database logic programs. In *Proc. of the 20th ACM SIGACT Symp. on Theory of Computing (STOC)*, 477–490.

Cuenca Grau, B.; Motik, B.; Stoilos, G.; and Horrocks, I. 2013. Computing datalog rewritings beyond Horn ontologies. In *Proc. of the 23rd Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 832–838.

Di Pinto, F.; Lembo, D.; Lenzerini, M.; Mancini, R.; Poggi, A.; Rosati, R.; Ruzzi, M.; and Savo, D. F. 2013. Optimizing query rewriting in ontology-based data access. In *Proc. of the 16th Int. Conf. on Extending Database Technology (EDBT)*, 561–572. ACM Press.

Eiter, T.; Ortiz, M.; Simkus, M.; Tran, T.-K.; and Xiao, G. 2012. Query rewriting for Horn-SHIQ plus rules. In *Proc. of the 26th AAAI Conf. on Artificial Intelligence (AAAI)*, 726–733. AAAI Press.

Gaifman, H.; Mairson, H. G.; Sagiv, Y.; and Vardi, M. Y. 1987. Undecidable optimization problems for database logic programs. In *Proc. of the 2nd IEEE Symp. on Logic in Computer Science (LICS)*, 106–115.

Giese, M.; Soylu, A.; Vega-Gorgojo, G.; Waaler, A.; Haase, P.; Jiménez-Ruiz, E.; Lanti, D.; Rezk, M.; Xiao, G.; Özçep, Ö. L.; and Rosati, R. 2015. Optique: Zooming in on Big Data. *IEEE Computer* 48(3):60–67.

Hustadt, U.; Motik, B.; and Sattler, U. 2005. Data complexity of reasoning in very expressive description logics. In *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 466–471.

Kazakov, Y. 2009. Consequence-driven reasoning for Horn-SHIQ ontologies. In *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 2040–2045.

Kontchakov, R.; Rezk, M.; Rodriguez-Muro, M.; Xiao, G.; and Zakharyaschev, M. 2014. Answering SPARQL queries over databases under OWL 2 QL entailment regime. In *Proc. of International Semantic Web Conference (ISWC 2014)*, Lecture Notes in Computer Science. Springer.

Lutz, C., and Wolter, F. 2011. Non-uniform data complexity of query answering in description logics. In *Proc. of the 24th Int. Workshop on Description Logic (DL)*, volume 745 of *CEUR Electronic Workshop Proceedings*.

Lutz, C.; Piro, R.; and Wolter, F. 2011. Description logic TBoxes: Model-theoretic characterizations and rewritability. In *Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 983–988.

Lutz, C.; Walther, D.; and Wolter, F. 2007. Conservative extensions in expressive description logics. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 453–458.

Ma, L.; Yang, Y.; Qiu, Z.; Xie, G.; Pan, Y.; and Liu, S. 2006. Towards a complete OWL ontology benchmark. In *Proc. of the 3rd European Semantic Web Conf. (ESWC)*, volume 4011 of *LNCS*, 125–139. Springer.

Motik, B.; Fokoue, A.; Horrocks, I.; Wu, Z.; Lutz, C.; and Cuenca Grau, B. 2009. OWL Web Ontology Language profiles. W3C Recommendation, World Wide Web Consortium. Available at `http://www.w3.org/TR/owl-profiles/`.

Pan, J. Z., and Thomas, E. 2007. Approximating OWL-DL ontologies. In *Proc. of the 21st AAAI Conf. on Artificial Intelligence (AAAI)*, 1434–1439.

Ren, Y.; Pan, J. Z.; and Zhao, Y. 2010. Soundness preserving approximation for TBox reasoning. In *Proc. of the 24th AAAI Conf. on Artificial Intelligence (AAAI)*.

Rodriguez-Muro, M.; Kontchakov, R.; and Zakharyaschev, M. 2013. Ontology-based data access: Ontop of databases. In *Proc. of the 12th Int. Semantic Web Conf. (ISWC)*, volume 8218 of *LNCS*, 558–573. Springer.

Trivela, D.; Stoilos, G.; Chortaras, A.; and Stamou, G. 2015. Optimising resolution-based rewriting algorithms for OWL ontologies. *J. of Web Semantics* 30–49.

A Appendix

A.1 Expansion of Datalog Programs

We recall here the notion of the expansion trees (Cosmadakis et al. 1988). Formally, an *expansion tree* for a predicate N in a Datalog program Π is a *finite* tree φ_Π^N satisfying the following conditions:

- Each node x of φ_Π^N is labeled by a pair of the form (α_x, ρ_x), where α_x is an IDB atom and ρ_x is an instance of a rule of Π such that the head of ρ_x is α_x. Moreover, the variables in the body of ρ_x either occur in α_x or they do not occur in the label of any node above x in the tree.
- The IDB atom labeling the root of φ_Π^N is an N-atom.
- If x is a node, where $\alpha_x = Y(\vec{t})$, $\rho_x = Y(\vec{t}) \leftarrow Y_1(\vec{t_1}), \dots, Y_m(\vec{t_m})$, and the IDB atoms in the body of ρ_x are $Y_{i_1}(\vec{t^{i_1}}), \dots, Y_{i_\ell}(\vec{t^{i_\ell}})$, then x has ℓ children, respectively labeled with the atoms $Y_{i_1}(\vec{t^{i_1}}), \dots, Y_{i_\ell}(\vec{t^{i_\ell}})$. In particular, if ρ_x is an initialization rule (i.e., the body of ρ_x does not contain an IDB predicate), then x is a leaf.

A.2 Proofs of Section 4.1

In the following, for an OBDA specification $\mathcal{P}$ and a database instance $\mathcal{D}$, if $\langle \mathcal{P}, \mathcal{D} \rangle$ has a model, we say that $\mathcal{D}$ is *consistent with* $\mathcal{P}$.

Lemma 3. *Let $\langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDA specification where $\mathcal{T}$ is Datalog rewritable. Then, for every database instance $\mathcal{D}$ of $\mathcal{S}$, concept or role name N of $\mathcal{T}$, and $\vec{a}$ in $\mathsf{Ind}(\mathcal{A}_{\mathcal{M},\mathcal{D}})$, we have that $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},\mathcal{D}} \rangle \models N(\vec{a})$ iff $N(\vec{a}) \in N^{\infty}_{\Pi_{\mathcal{T},\mathcal{M}}}(\mathcal{D})$.*

Proof. We assume that the Datalog translation $\Pi_\mathcal{T}$ of $\mathcal{T}$ satisfies the following properties (see, e.g., Theorem 1 in (Hustadt, Motik, and Sattler 2005) and Proposition 2 in (Eiter et al. 2012)):

$\star$ for every ABox $\mathcal{A}$, $\langle \mathcal{T}, \mathcal{A} \rangle$ is consistent iff $\bot^{\infty}_{\Pi_\mathcal{T}}(\mathcal{A})$ is empty, and if $\langle \mathcal{T}, \mathcal{A} \rangle$ is consistent, then for every concept or role name N of $\mathcal{T}$ and $\vec{a}$ in $\mathsf{Ind}(\mathcal{A})$, $\langle \mathcal{T}, \mathcal{A} \rangle \models N(\vec{a})$ iff $N(\vec{a}) \in N^{\infty}_{\Pi_\mathcal{T}}(\mathcal{A})$.

Recall that for an ABox $\mathcal{A}$ such that $\langle \mathcal{T}, \mathcal{A} \rangle$ is inconsistent, $\langle \mathcal{T}, \mathcal{A} \rangle$ entails all possible facts of the form $N(\vec{a})$ for a concept or role name N of $\mathcal{T}$ and $\vec{a}$ in $\mathsf{Ind}(\mathcal{A}_{\mathcal{M},\mathcal{D}})$. We prove that the translation $\Pi_\mathcal{T}$ containing the auxiliary rules involving $\top_\Delta$ and $\bot$ satisfies a stronger property:

$\star\star$ for every ABox $\mathcal{A}$, for every concept or role name N of $\mathcal{T}$ and $\vec{a}$ in $\mathsf{Ind}(\mathcal{A})$, $\langle \mathcal{T}, \mathcal{A} \rangle \models N(\vec{a})$ iff $N(\vec{a}) \in N^{\infty}_{\Pi_\mathcal{T}}(\mathcal{A})$.

Indeed, it is easy to see that considering that *(i)* $\top_\Delta$ contains precisely $\mathsf{Ind}(\mathcal{A})$, i.e., $\top_\Delta(a) \in {\top_\Delta}^{\infty}_{\Pi_\mathcal{T}}(\mathcal{A})$ for each $a \in \mathsf{Ind}(\mathcal{A})$, and *(ii)* if $\bot$ is true in $\Pi_\mathcal{T}(\mathcal{A})$, then for each concept or role name N of $\mathcal{T}$ and $\vec{a}$ in $\mathsf{Ind}(\mathcal{A}_{\mathcal{M},\mathcal{D}})$, we have that $N(\vec{a}) \in N^{\infty}_{\Pi_\mathcal{T}}(\mathcal{A})$.

As $\Pi_{\mathcal{T},\mathcal{M}} = \Pi_\mathcal{T} \cup \Pi_\mathcal{M}$, the statement of the lemma follows directly from the property $\star\star$ of $\Pi_\mathcal{T}$ and the fact that the rules in $\Pi_\mathcal{M}$ connect two disjoint vocabularies. □

Lemma 10. *Let $\langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDA specification where $\mathcal{T}$ is Datalog rewritable, and $\mathcal{M}' = \mathsf{etm}_\mathcal{T}(\mathcal{M})$. Then for every database instance $\mathcal{D}$ of $\mathcal{S}$, we have that $\mathcal{A}_{\mathcal{M}',\mathcal{D}}$ is exactly the set of all facts entailed by $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},\mathcal{D}} \rangle$, i.e., assertions of the form $A(a)$, $P(a,b)$ for $a, b \in \mathsf{Ind}(\mathcal{A}_{\mathcal{M},\mathcal{D}})$, $A, P \in \mathsf{sig}(\mathcal{T})$.*

Proof. Let $\mathcal{D}$ be a database instance of $\mathcal{S}$, $a \in \mathsf{Ind}(\mathcal{A}_{\mathcal{M},\mathcal{D}})$, A a concept name in $\mathsf{sig}(\mathcal{T})$. Then

- $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},\mathcal{D}} \rangle \models A(a)$ iff (by Lemma 3)
- $A(a) \in A^{\infty}_{\Pi_{\mathcal{T},\mathcal{M}}}(\mathcal{D})$ iff (by the properties of expansions)
- there exists DB-defined $\varphi \in \Phi_{\Pi_{\mathcal{T},\mathcal{M}}}(A)$ such that $a \in \mathit{ans}(\varphi, \mathcal{D})$.

Let $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},\mathcal{D}} \rangle \models A(a)$. By construction of $\mathcal{M}'$, the mapping assertion $\Phi_{\Pi_{\mathcal{T},\mathcal{M}}}(A) \rightsquigarrow A(x)$ is in $\mathcal{M}'$. We conclude that $A(a) \in \mathcal{A}_{\mathcal{M}',\mathcal{D}}$. Now, assume that $A(a) \in \mathcal{A}_{\mathcal{M}',\mathcal{D}}$. It follows that in $\mathcal{M}'$ there is a mapping assertion $\Phi_{\Pi_{\mathcal{T},\mathcal{M}}}(A) \rightsquigarrow A(x)$ and $a \in \mathit{ans}(\varphi, \mathcal{D})$ for some $\varphi \in \Phi_{\Pi_{\mathcal{T},\mathcal{M}}}(A)$. We conclude that $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},\mathcal{D}} \rangle \models A(a)$.

The proof for role assertions is analogous. □

A.3 Proof of Theorem 6

We start by showing a sufficient condition for Σ-CQ inseparability of OBDA specifications.

A *homomorphism* between two interpretations is a mapping between their domains that preserves constants and relations. A model of an ontology $\mathcal{O}$ that can be homomorphically embedded in every model of $\mathcal{O}$ is called a *canonical model* of $\mathcal{O}$, from now on denoted $\mathcal{C}_\mathcal{O}$. The notion of canonical model is important in the context of CQ answering, since answers to CQs are preserved under homomorphisms, i.e., if $q(\vec{a})$ holds in an interpretation $\mathcal{I}$, and there is a homomorphism from $\mathcal{I}$ to an interpretation $\mathcal{I}'$, then $q(\vec{a})$ holds in $\mathcal{I}'$ (Chandra and Merlin 1977). It follows that certain answers can be characterized as the answers over the canonical model.

Lemma 11. *Let $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ be a consistent ontology that has a canonical model $\mathcal{C}_\mathcal{O}$, $q(\vec{x})$ a CQ, and $\vec{a}$ a tuple from $\mathsf{Ind}(\mathcal{A})$. Then $\vec{a} \in \mathit{cert}(q, \mathcal{O})$ iff $\mathcal{C}_\mathcal{O}$ satisfies $q(\vec{a})$.*

It is well known that Horn variants of DLs have the *canonical model property* (Eiter et al. 2012; Botoeva et al. 2014), i.e., every satisfiable ontology $\mathcal{O}$ admits a (possibly infinite) canonical model.

A Σ-homomorphism is a mapping that preserves constants and relations in Σ. Given two interpretations $\mathcal{I}$ and $\mathcal{J}$, we say that $\mathcal{I}$ is (Σ-)homomorphically embeddable into $\mathcal{J}$ if there exists a (Σ-)homomorphism from $\mathcal{I}$ to $\mathcal{J}$. Moreover, $\mathcal{I}$ and $\mathcal{J}$ are (Σ-)homomorphically equivalent if they are (Σ-)homomorphically embeddable into each other. The following characterization can be easily derived from Lemma 11, the property of canonical models, and the definitions of certain answers.

Lemma 12. *Let Σ be a signature, and $\mathcal{P}_1 = \langle \mathcal{T}_1, \mathcal{M}_1, \mathcal{S} \rangle$ and $\mathcal{P}_2 = \langle \mathcal{T}_2, \mathcal{M}_2, \mathcal{S} \rangle$ two OBDA specifications. Assume that for every ABox $\mathcal{A}$, both $\langle \mathcal{T}_1, \mathcal{A} \rangle$ and $\langle \mathcal{T}_2, \mathcal{A} \rangle$ admit a canonical model. If*

- *for every database instance $\mathcal{D}$ of $\mathcal{S}$ that is consistent with both $\mathcal{P}_1$ and $\mathcal{P}_2$, we have that $\mathcal{C}_{\langle \mathcal{T}_1, \mathcal{A}_{\mathcal{M}_1,\mathcal{D}} \rangle}$ and $\mathcal{C}_{\langle \mathcal{T}_2, \mathcal{A}_{\mathcal{M}_2,\mathcal{D}} \rangle}$ are Σ-homomorphically equivalent, and*

- *for every database instance $\mathcal{D}$ of $\mathcal{S}$ that is inconsistent with $\mathcal{P}_1$ or $\mathcal{P}_2$, we have that $\langle \mathcal{T}_1, \mathcal{A}_{\mathcal{M}_1,\mathcal{D}}\rangle$ and $\langle \mathcal{T}_2, \mathcal{A}_{\mathcal{M}_2,\mathcal{D}}\rangle$ entail the same ABox facts over Σ,*

then $\mathcal{P}_1$ and $\mathcal{P}_2$ are Σ-CQ inseparable.

Now we show several properties of the intermediate and final TBoxes obtained during the $\mathsf{RewObda}(\mathcal{T},\mathcal{M})$ procedure. First, we show that, for ABoxes containing a single assertion, the *DL-Lite*$_\mathcal{R}$ TBox $\mathcal{T}_r$ generates a canonical model equivalent to the canonical model generated by the intermediate Horn-$\mathcal{ALCHIQ}$ TBox $\mathcal{T}_3$.

Lemma 13. *Let $\mathcal{T}$ be a Horn-$\mathcal{ALCHIQ}$ TBox, $\mathcal{T}_3$ the TBox obtained in step 3 of $\mathsf{RewObda}(\mathcal{T},\mathcal{M})$, and $\mathcal{T}_r = \mathsf{rew}(\mathcal{T})$. Also, for A a concept name in $\mathsf{sig}(\mathcal{T}_3)$ and $a \in \mathsf{N_I}$, let $\mathcal{A} = \{A(a)\}$ be an ABox such that $\langle \mathcal{T}_3, \mathcal{A}\rangle$ is consistent. Then $\mathcal{C}_{\langle \mathcal{T}_3,\mathcal{A}\rangle}$ is homomorphically equivalent to $\mathcal{C}_{\langle \mathcal{T}_r,\mathcal{A}\rangle}$.*

Proof. In this proof, we call an element σ' a successor of σ in a canonical model $\mathcal{C}$ of $\langle \mathcal{T},\mathcal{A}\rangle$, for $\sigma,\sigma' \in \Delta^\mathcal{C}$, if σ' is added (i.e., generated) to satisfy an existential assertion α of the form $\bigsqcap A_i \sqsubseteq \exists(\bigsqcap Q_j).(\bigsqcap A'_k)$ such that $\sigma \in A_i^\mathcal{C}$. Here we assume a construction of the canonical model that is "minimal" and uniquely defined: a new successor σ' of σ is generated only if there is no other element δ of $\Delta^\mathcal{C}$ that can be used to satisfy α, in this case we employ the following naming convention: σ' is a path of the form $\sigma \cdot w_{(\{Q_j\},\{A'_k\})}$. In particular, if there is a role P in $\mathcal{T}$ such that it only appears on the left-hand side of role inclusions, and $\mathcal{T} \models \{\bigsqcap A_i \sqsubseteq \exists P, P \sqsubseteq Q_j, \exists P^- \sqsubseteq A'_k\}$, then σ has to have a successor δ introduced to satisfy the assertion $\bigsqcap A_i \sqsubseteq \exists P$, so δ can be used to satisfy α and no new successor σ' is generated. Also, if the predecessor δ' of σ is such that $\delta' \in (A'_k)^\mathcal{C}$ and $(\sigma,\delta') \in Q_j^\mathcal{C}$, then no new successor is introduced. Here, we call each fresh role name P_{new} introduced by the normalization procedure $\mathsf{norm}_\exists$ a *generating DL-Lite$_\mathcal{R}$-role* precisely because it satisfies the above properties. Note that for a generating *DL-Lite*$_\mathcal{R}$-role P, the concept $\exists P^-$ has no non-empty sub-concepts and P does not appear in constructs $\exists P.C$ where C is a concept distinct from $\top$. Therefore, the element introduced as a P-successor of σ can be simply named $\sigma \cdot w_P$ (or $\sigma \cdot v_P$ to distinguish between two canonical models).

Observe that $\mathsf{sig}(\mathcal{T}_3) = \mathsf{sig}(\mathcal{T}_r)$. Let A be a satisfiable concept name in $\mathsf{sig}(\mathcal{T}_3)$, and $\mathcal{A} = \{A(a)\}$. Denote by $\mathcal{C}_1$ the canonical model of $\langle \mathcal{T}_3,\mathcal{A}\rangle$, and by $\mathcal{C}_2$ the canonical model of $\langle \mathcal{T}_r,\mathcal{A}\rangle$.

First, for each element in $\Delta^{\mathcal{C}_1}$ distinct from a, we prove that it is of the form $aw_1\cdots w_n$, where each $w_i = w_P$ for some generating *DL-Lite*$_\mathcal{R}$-role P. Suppose that $\sigma \in \Delta^{\mathcal{C}_1}$, $\sigma \in B_i^{\mathcal{C}_1}$, $1 \le i \le k$, and $\mathcal{T}_3 \models \alpha$ where $\alpha = B_1 \sqcap \cdots \sqcap B_k \sqsubseteq \exists Q.(A_1 \sqcap \cdots \sqcap A_m)$. By induction on the length of σ, we find an element $\delta \in \Delta^{\mathcal{C}_1}$ such that $\delta \in A_i^{\mathcal{C}_1}$, $1 \le i \le m$, $(\sigma,\delta) \in Q^{\mathcal{C}_1}$, and δ is of the desired form. Note that without loss of generality we may assume that none of $B_1,\ldots,B_k$ and none of $A_1,\ldots,A_m$ is a fresh concept name introduced by $\mathsf{norm}_\sqcap$ as we can substitute each such name with its definition. Therefore, $\{B_1,\ldots,B_k,A_1,\ldots,A_m\} \subseteq \mathsf{sig}(\mathcal{T})$. Consider the following cases:

(a) If $Q \in \mathsf{sig}(\mathcal{T})$, then by step 1 $\alpha \in \mathcal{T}_1$, and by $\mathsf{norm}_\exists$ in step 2, $\mathcal{T}_2$ (hence, $\mathcal{T}_3$) contains axioms $B_1 \sqcap \cdots \sqcap B_k \sqsubseteq \exists P$, $P \sqsubseteq Q$, $\exists P^- \sqsubseteq A_i$, $1 \le i \le m$, for a fresh role name P. If $\sigma = a$, then we set $\delta = aw_P \in \Delta^{\mathcal{C}_1}$, for which it holds that that $aw_P \in A_i^{\mathcal{C}_1}$, $1 \le i \le m$, and $(a, aw_P) \in Q^{\mathcal{C}_1}$. If $\sigma = \sigma' w_S$ for some generating *DL-Lite*$_\mathcal{R}$-role S, and it is not the case that $\mathcal{T}_3 \models S \sqsubseteq Q^-$ and $\sigma' \in A_i^{\mathcal{C}_1}$, then we set $\delta = \sigma w_P$, for which we have that $\sigma w_P \in A_i^{\mathcal{C}_1}$, $1 \le i \le m$, and $(\sigma, \sigma w_P) \in Q^{\mathcal{C}_1}$. Otherwise we set $\delta = \sigma'$, which is of the desired form.
(b) If $Q \notin \mathsf{sig}(\mathcal{T})$, then Q is a fresh generating *DL-Lite*$_\mathcal{R}$ role introduced by $\mathsf{norm}_\exists$. It means that there exists a CI $\bigsqcap A'_j \sqsubseteq \exists S.(\bigsqcap A''_l)$ in $\mathcal{T}_1$ such that $\mathcal{T}_2$ contains axioms $\bigsqcap A'_j \sqsubseteq \exists Q$, $Q \sqsubseteq S$, and $\top \sqsubseteq \forall Q.A''_l$. Since Q occurs only in these axioms, it must be the case that $\mathcal{T}_3 \models B_1 \sqcap \cdots \sqcap B_k \sqsubseteq \bigsqcap A'_j$ and $\mathcal{T}_3 \models \bigsqcap A''_l \sqsubseteq A_1 \sqcap \cdots \sqcap A_m$. Because of the former, we obtain that $\sigma \in (A'_j)^{\mathcal{C}_1}$, and similarly to (a) that there is an element $\delta \in \Delta^{\mathcal{C}_1}$ such that $\delta \in (A''_l)^{\mathcal{C}_1}$ and $(\sigma,\delta) \in Q^{\mathcal{C}_1}$. Because of the latter, we also have that $\delta \in A_i^{\mathcal{C}_1}$, $1 \le i \le m$.

Second, we show that there exists a Σ-homomorphism from $\mathcal{C}_1$ to $\mathcal{C}_2$, by constructing one. Let $a \in B^{\mathcal{C}_1}$ for a basic concept B. Then it must be that $\mathcal{T}_3 \models A \sqsubseteq B$. By construction, $\mathcal{T}_r \models A \sqsubseteq B$, and therefore $a \in B^{\mathcal{C}_2}$. So we can set $h(a) = a$.

Let $\sigma \in \Delta^{\mathcal{C}_1}$ such that $h(\sigma)$ is set, $h(\sigma) = \delta$, and $\sigma w_P \in \Delta^{\mathcal{C}_1}$. Then $\sigma \in (\exists P)^{\mathcal{C}_1}$, hence $\delta \in (\exists P)^{\mathcal{C}_2}$. Since P is a generating *DL-Lite*$_\mathcal{R}$-role in $\mathcal{T}_3$, and $\mathcal{T}_r$ is derived from $\mathcal{T}_3$, it follows that P is a generating *DL-Lite*$_\mathcal{R}$-role in $\mathcal{T}_r$, hence there exists a successor $\delta v_P \in \Delta^{\mathcal{C}_2}$. By construction of $\mathcal{T}_r$, it follows that we can set $h(\sigma w_P) = \delta v_P$ (that is, the homomorphism conditions are satisfied).

Finally, as for a homomorphism from $\mathcal{C}_2$ to $\mathcal{C}_1$, its existence follows from the fact that $\mathcal{T}_3 \models \mathcal{T}_r$: $\mathcal{C}_2$ is homomorphically embeddable into each model $\mathcal{I}$ of $\langle \mathcal{T}_r,\mathcal{A}\rangle$ and as $\mathcal{T}_3 \models \mathcal{T}_r$, $\mathcal{C}_1$ is a model of $\langle \mathcal{T}_r,\mathcal{A}\rangle$ as well. $\square$

In general, however, $\mathcal{T}_r$ does not preserve ABox entailments for non-singleton ABoxes.

Example 14. Consider the following TBox $\mathcal{T}$, together with the computed $\mathcal{T}_3$ and $\mathcal{T}_r$:

$$\begin{aligned}
\mathcal{T} &= \{B \sqcap C \sqsubseteq A,\ B \sqcap C \sqsubseteq \exists P\}\\
\mathcal{T}_3 &= \mathcal{T} \cup \{A_{B\sqcap C} \equiv B \sqcap C,\ A_{B\sqcap C} \sqsubseteq \exists P\}\\
\mathcal{T}_r &= \{A_{B\sqcap C} \sqsubseteq A, A_{B\sqcap C} \sqsubseteq B, A_{B\sqcap C} \sqsubseteq C, A_{B\sqcap C} \sqsubseteq \exists P\}
\end{aligned}$$

Then, for $\mathcal{A} = \{B(a), C(a)\}$ an ABox, $\langle \mathcal{T}_3,\mathcal{A}\rangle \models A(a)$, however $\langle \mathcal{T}_r,\mathcal{A}\rangle \not\models A(a)$. ■

Next, we extend Lemma 13 towards arbitrary ABoxes. To do so, we consider ABoxes that are closed with respect to $\mathcal{T}_3$, where an ABox $\mathcal{A}$ is *closed with respect to a TBox* $\mathcal{T}$ if $\mathcal{A} = \mathsf{EABox}(\mathcal{T},\mathcal{A})$ where $\mathsf{EABox}(\mathcal{T},\mathcal{A})$ is the set of all membership assertions over $\mathsf{sig}(\mathcal{T})$ and $\mathsf{Ind}(\mathcal{A})$ entailed by $\langle \mathcal{T},\mathcal{A}\rangle$. We say that an ABox $\mathcal{A}$ is *complete* (within $\mathsf{RewObda}(\mathcal{T},\mathcal{M})$), if it is closed with respect to $\mathcal{T}_3$. The following result is a corollary of Lemma 13.

Lemma 15. *Let $\mathcal{T}$ be a Horn-$\mathcal{ALCHIQ}$ TBox, $\mathcal{T}_3$ the TBox obtained in step 3 of* $\mathsf{RewObda}(\mathcal{T},\mathcal{M})$*, and $\mathcal{T}_r = \mathsf{rew}(\mathcal{T})$. Then for each ABox $\mathcal{A}$ that is complete and such that $\langle\mathcal{T}_3,\mathcal{A}\rangle$ is consistent, we have that $\mathcal{C}_{\langle\mathcal{T}_3,\mathcal{A}\rangle}$ and $\mathcal{C}_{\langle\mathcal{T}_r,\mathcal{A}\rangle}$ are homomorphically equivalent.*

Proof. We use the same assumptions for the canonical model as in the proof of Lemma 13. Denote by $\mathcal{C}_1$ the canonical model of $\langle\mathcal{T}_3,\mathcal{A}\rangle$, and by $\mathcal{C}_2$ the canonical model of $\langle\mathcal{T}_r,\mathcal{A}\rangle$ for a complete ABox $\mathcal{A}$. The existence of a homomorphism from $\mathcal{C}_2$ to $\mathcal{C}_1$ is straightforward. We show that there exists a homomorphism from $\mathcal{C}_1$ to $\mathcal{C}_2$. Let $a \in \mathsf{Ind}(\mathcal{A})$, it is sufficient to show that for each successor aw_P of a in $\mathcal{C}_1$, there is a successor av_P of a in $\mathcal{C}_2$. The rest of the proof follows from Lemma 13.

Let σ be a successor of a in $\mathcal{C}_1$. It follows from the proof of Lemma 13 that σ is of the form aw_P where P is a generating *DL-Lite*$_\mathcal{R}$ role and $\langle\mathcal{T}_3,\mathcal{A}\rangle \models \exists P(a)$ (moreover, there exists no individual $b \in \mathsf{Ind}(\mathcal{A})$ such that $R(a,b) \in \mathcal{A}$ for each role R with $\mathcal{T}_3 \models P \sqsubseteq R$ and $B(b) \in \mathcal{A}$ for each concept name B with $\mathcal{T}_3 \models \exists P^- \sqsubseteq B$).

We show that $\langle\mathcal{T}_r,\mathcal{A}\rangle \models \exists P(a)$. Recall that $\mathcal{T}_3$ contains all possible CIs with $\exists P$ on the right-hand side. From $\langle\mathcal{T}_3,\mathcal{A}\rangle \models \exists P(a)$, it follows that there exists a CI $A_1 \sqcap \cdots \sqcap A_n \sqsubseteq \exists P$ in $\mathcal{T}_3$, $n \geq 1$, such that $A_i(a) \in \mathcal{A}$. If $n = 1$, then $\mathcal{T}_r \models A_1 \sqsubseteq \exists P$, hence $\langle\mathcal{T}_r,\mathcal{A}\rangle \models \exists P(a)$. Assume that $n > 1$, then by step 3 $\mathcal{T}_3$ contains $A_{A_1\sqcap\cdots\sqcap A_n} \equiv A_1 \sqcap \cdots \sqcap A_n$, therefore $\mathcal{T}_r$ contains $A_{A_1\sqcap\cdots\sqcap A_n} \sqsubseteq \exists P$, and since $\mathcal{A}$ is closed with respect to $\mathcal{T}_3$, $\mathcal{A}$ contains $A_{A_1\sqcap\cdots\sqcap A_n}(a)$. Finally, we obtain that $\mathcal{T}_r \models \exists P(a)$.

Now, as no individual b can be used as a P-successor of a, in $\mathcal{C}_2$ there is a successor av_P of a. □

The result above is significant, because the mapping $\mathcal{M}_c = \mathsf{comp}(\mathcal{T},\mathcal{M})$ is such that it generates from a database instance a virtual ABox that is complete (within $\mathsf{comp}(\mathcal{T},\mathcal{M})$). Finally, combined with Lemmas 10 and 15, we are ready to prove Theorem 6.

Theorem 6. *Let $\langle\mathcal{T},\mathcal{M},\mathcal{S}\rangle$ be an OBDA specification, $\Sigma =$* $\mathsf{sig}(\mathcal{T})$*, and $\langle\mathcal{T}_r,\mathcal{M}_c\rangle = \mathsf{RewObda}(\mathcal{T},\mathcal{M})$. Then, for each database instance $\mathcal{D}$ of $\mathcal{S}$ and for each Σ-query q, we have that $cert(q,\langle\mathcal{T},\mathcal{M},\mathcal{S}\rangle,\mathcal{D}) = cert(q,\langle\mathcal{T}_r,\mathcal{M}_c,\mathcal{S}\rangle,\mathcal{D})$.*

Proof. **(a)** Let $\mathcal{D}$ be an instance of $\mathcal{S}$ inconsistent with $\langle\mathcal{T},\mathcal{M},\mathcal{S}\rangle$. By Lemma 10 $\mathcal{A}_{\mathcal{M}_c,\mathcal{D}} = \mathsf{EABox}(\langle\mathcal{T}_3,\mathcal{A}_{\mathcal{M},\mathcal{D}}\rangle)$, and since $\mathcal{D}$ is inconsistent with $\langle\mathcal{T}_3,\mathcal{M},\mathcal{S}\rangle$, we have that $\mathcal{A}_{\mathcal{M}_c,\mathcal{D}}$ contains all possible facts over $\mathsf{sig}(\mathcal{T})$. The axioms in $\mathcal{T}_r$ do not add more facts.

(b) We show that for each database instance $\mathcal{D}$ of $\mathcal{S}$ consistent with $\langle\mathcal{T},\mathcal{M},\mathcal{S}\rangle$, the canonical models $\mathcal{C}_{\langle\mathcal{T},\mathcal{A}_{\mathcal{M},\mathcal{D}}\rangle}$ and $\mathcal{C}_{\langle\mathcal{T}_r,\mathcal{A}_{\mathcal{M}_c,\mathcal{D}}\rangle}$ are Σ-homomorphically equivalent.

(I) We observe that by Lemma 10, it follows that $\mathcal{A}_{\mathcal{M}_c,\mathcal{D}}$ is closed with respect to $\mathcal{T}_3$.

(II) We show that $\mathcal{C}_{\langle\mathcal{T},\mathcal{A}\rangle}$ is Σ-homomorphically equivalent to $\mathcal{C}_{\langle\mathcal{T}_3,\mathcal{A}\rangle}$, where $\mathcal{A}$ is an ABox and $\Sigma = \mathsf{sig}(\mathcal{T})$. The interesting direction is the existence of a Σ-homomorphism from $\mathcal{C}_{\langle\mathcal{T}_3,\mathcal{A}\rangle}$ to $\mathcal{C}_{\langle\mathcal{T},\mathcal{A}\rangle}$. Since $\mathcal{T}_3$ is a model-conservative extension of $\mathcal{T}$, $\mathcal{C}_{\langle\mathcal{T},\mathcal{A}\rangle}$ can be extended without changing the interpretations of symbols in Σ to a model $\mathcal{I}$ of $\langle\mathcal{T}_3,\mathcal{A}\rangle$. By definition of canonical model, there exists a homomorphism from $\mathcal{C}_{\langle\mathcal{T}_3,\mathcal{A}\rangle}$ to $\mathcal{I}$ and since $\mathcal{I}$ agrees with $\mathcal{C}_{\langle\mathcal{T},\mathcal{A}\rangle}$ on Σ, we obtain that there exists a Σ-homomorphism from $\mathcal{C}_{\langle\mathcal{T}_3,\mathcal{A}\rangle}$ to $\mathcal{C}_{\langle\mathcal{T},\mathcal{A}\rangle}$.

(III) We show that $\mathsf{EABox}_\Sigma(\mathcal{T},\mathcal{A}_{\mathcal{M}_c,\mathcal{D}}) = \mathsf{EABox}_\Sigma(\mathcal{T},\mathcal{A}_{\mathcal{M},\mathcal{D}})$, where $\Sigma = \mathsf{sig}(\mathcal{T})$ and $\mathsf{EABox}_\Sigma(\mathcal{T},\mathcal{A})$ is the projection of $\mathsf{EABox}(\mathcal{T},\mathcal{A})$ on Σ. Assume that $A(a) \in \mathsf{EABox}(\mathcal{T}_3,\mathcal{A}_{\mathcal{M}_c,D})$ such that $A(a) \notin \mathsf{EABox}(\mathcal{T},\mathcal{A}_{\mathcal{M},\mathcal{D}})$. Then A is a fresh concept introduced in step 3, hence $A \notin \Sigma$ and $\mathsf{EABox}_\Sigma(\mathcal{T}_3,\mathcal{A}_{\mathcal{M}_c,\mathcal{D}}) = \mathsf{EABox}_\Sigma(\mathcal{T},\mathcal{A}_{\mathcal{M},\mathcal{D}})$. Combining it with (II), we conclude that $\mathsf{EABox}_\Sigma(\mathcal{T},\mathcal{A}_{\mathcal{M}_c,\mathcal{D}}) = \mathsf{EABox}_\Sigma(\mathcal{T},\mathcal{A}_{\mathcal{M},\mathcal{D}})$.

Now, by Lemma 15 and (I), by (II) and by (III) we obtain the following (Σ-)homomorphic equivalences $\equiv$ ($\equiv_\Sigma$):

$$\mathcal{C}_{\langle\mathcal{T}_r,\mathcal{A}_{\mathcal{M}_c,\mathcal{D}}\rangle} \equiv \mathcal{C}_{\langle\mathcal{T}_3,\mathcal{A}_{\mathcal{M}_c,\mathcal{D}}\rangle} \equiv_\Sigma \mathcal{C}_{\langle\mathcal{T},\mathcal{A}_{\mathcal{M}_c,\mathcal{D}}\rangle} \equiv_\Sigma \mathcal{C}_{\langle\mathcal{T},\mathcal{A}_{\mathcal{M},\mathcal{D}}\rangle}.$$

Hence, $\mathcal{C}_{\langle\mathcal{T}_r,\mathcal{A}_{\mathcal{M}_c,\mathcal{D}}\rangle}$ and $\mathcal{C}_{\langle\mathcal{T},\mathcal{A}_{\mathcal{M},\mathcal{D}}\rangle}$ are Σ-homomorphically equivalent.

Finally, by Lemma 12 and **(a)**, **(b)**, we conclude that $\langle\mathcal{T},\mathcal{M},\mathcal{S}\rangle$ and $\langle\mathcal{T}_r,\mathcal{M}_c,\mathcal{S}\rangle$ are Σ-CQ inseparable. □

A.4 Proofs of Section 5

Theorem 7. *Let $\langle\mathcal{T},\mathcal{M},\mathcal{S}\rangle$ be an OBDA specification such that $\mathcal{T}$ is a Horn-$\mathcal{ALCHIQ}$ TBox. Further, let $\mathcal{T}_r = \mathsf{rew}(\mathcal{T})$ and $\mathcal{M}' = \mathsf{cut}_k^\Omega(\mathsf{comp}(\mathcal{T},\mathcal{M}))$, for a boundedness oracle Ω and some $k > 0$. If $\mathcal{T}$ is FO-rewritable for AQs, then $\langle\mathcal{T},\mathcal{M},\mathcal{S}\rangle$ is* CQ-rewritable *into DL-Lite$_\mathcal{R}$, and $\langle\mathcal{T}_r,\mathcal{M}',\mathcal{S}\rangle$ is its* CQ-rewriting. *Otherwise, $\langle\mathcal{T}_r,\mathcal{M}',\mathcal{S}\rangle$ is a* sound CQ-approximation *of $\langle\mathcal{T},\mathcal{M},\mathcal{S}\rangle$ in DL-Lite$_\mathcal{R}$.*

Proof. We note that since $\mathcal{T}$ is a TBox of depth 1 (it is assumed to be in normal form), if $\mathcal{T}$ is FO-rewritable for AQs then by (Lutz and Wolter 2011, Lemma 5) $\mathcal{T}$ is FO-rewritable for CQs. □

Theorem 8. *The problem of checking whether an OBDA specification with an $\mathcal{EL}$ ontology and FO source queries in the mapping is CQ-rewritable into DL-Lite$_\mathcal{R}$ is undecidable.*

Proof. Proof by reduction from the satisfiability problem of first-order logic.

Let φ be a closed first-order formula. We construct an OBDA specification $\mathcal{P} = \langle\mathcal{T},\mathcal{M},\mathcal{S}\rangle$ such that $\mathcal{P}$ is CQ-rewritable into *DL-Lite*$_\mathcal{R}$ iff φ is unsatisfiable.

We set $\mathcal{T} = \{\exists R.A \sqsubseteq A\}$. $\mathcal{S}$ contains all predicates in φ, a binary relation $tableR$ and a unary relation $tableA$ such that $tableR, tableA$ do not occur in φ. $\mathcal{M}$ consists of two mapping assertions: $tableR(x,y) \wedge \varphi \rightsquigarrow R(x,y)$ and $tableA(x) \rightsquigarrow A(x)$.

Assume that φ is unsatisfiable. Then for each database instance $\mathcal{D}$ of $\mathcal{S}$ we have that R is empty in $\mathcal{A}_{\mathcal{M},\mathcal{D}}$. It is straightforward to see that $\langle\emptyset,\mathcal{M},\mathcal{S}\rangle$ is a CQ-rewriting of $\mathcal{P}$ into *DL-Lite*$_\mathcal{R}$.

Assume that φ is satisfiable and, for the sake of contradiction, suppose that $\mathcal{P}$ is CQ-rewritable into *DL-Lite*$_\mathcal{R}$ and $\mathcal{P}' = \langle\mathcal{T}',\mathcal{M}',\mathcal{S}\rangle$ is such a CQ-rewriting where $\mathcal{T}'$ is a *DL-Lite*$_\mathcal{R}$ TBox. Now, consider an instance of the reachability problem $G = (V,E)$ and two vertices $s,t \in V$.

Let $\mathcal{D}$ be a database instance that satisfies φ and such that for each $(v, u) \in E$, $(v, u) \in \mathit{tableR}$ and $s \in \mathit{tableA}$. It is the standard reduction of the reachability problem to query answering in $\mathcal{EL}$, therefore $t \in \mathit{cert}(A(x), \langle\mathcal{P}, \mathcal{D}\rangle)$ iff t is reachable from s in G. Since $\mathcal{P}'$ is a rewriting of $\mathcal{P}$and $\mathcal{T}'$ is a *DL-Lite*$_\mathcal{R}$ TBox, there exists a FO-query $q_A(x)$ such that $\mathit{cert}(A(x), \langle\mathcal{P}, \mathcal{D}\rangle) = \mathit{cert}(A(x), \langle\mathcal{P}', \mathcal{D}\rangle) = \mathit{ans}(q_A(x), \mathcal{D})$. Thus, we obtain that $t \in \mathit{ans}(q_A(x), \mathcal{D})$ iff t is reachable from s in G. It means that we can solve the reachability problem by evaluating a FO-query over the database encoding the graph, which contradicts the NLOGSPACE-hardness of the reachability problem. Contradiction rises from the assumption that $\mathcal{P}$ is CQ-rewritable into *DL-Lite*$_\mathcal{R}$. □

Theorem 9. *The problem of checking whether an OBDA specification with a Horn-$\mathcal{ALCHI}$ ontology of depth one and unions of CQs as source queries in the mapping is CQ-rewritable into DL-Lite$_\mathcal{R}$ is decidable.*

Proof. Let $\mathcal{P} = \langle\mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ be an OBDA specification (here we do not split $\mathcal{M}$ into high- and low-level mappings). We construct a monadic Datalog program Π without inequalities worst case exponential in the size of $\mathcal{T}$ and $\mathcal{M}$ such that

$$\Pi \text{ is program bounded iff } \mathcal{P} \text{ is CQ-rewritable into } \textit{DL-Lite}_\mathcal{R}, \quad (1)$$

where Π is said to be program bounded if each predicate N mentioned in Π is bounded in Π, and a Datalog program Π is *monadic* if all its IDB predicates are monadic (unary). It is known that program boundedness of monadic Datalog programs without inequalities in decidable in 3EXPTIME (Cosmadakis et al. 1988). Thus, we obtain a 4EXPTIME algorithm for deciding CQ-rewritability into *DL-Lite*$_\mathcal{R}$.

Let $\mathcal{T}_3$ be the TBox obtained as an intermediate result in Step 3 of $\mathsf{RewObda}(\mathcal{T}, \mathcal{M})$. Then Π is the monadic Datalog program such that

$$A^\infty_\Pi(\mathcal{D}) = A^\infty_{\Pi_{\mathcal{T}_3,\mathcal{M}}}(\mathcal{D}), \quad (2)$$

for each instance $\mathcal{D}$ of $\mathcal{S}$ and each concept A in $\mathcal{T}_3$,

and φ^A is DB-defined for each $\varphi^A \in \Phi_\Pi(A)$. Observe that the Datalog translation $\Pi_{\mathcal{T}_3,\mathcal{M}}$ of the Horn-$\mathcal{ALCHI}$ TBox $\mathcal{T}_3$ is a Datalog program without inequalities. Therefore, we have that Π is a monadic Datalog program without inequalities and its program boundedness is decidable. Moreover, observe that $\Phi_{\Pi_1}(P)$ is a finite union of CQs for each role name P in $\mathcal{T}_3$. We first prove (1), then we show how Π is constructed.

Assume that Π is program bounded and let Ω be a boundedness oracle for it. Note that since Π is bounded, for a concept name A, $\mathsf{cut}^\Omega_k(A, \Pi)$ does not depend on the value of k. Next, let $\mathcal{T}_r = \mathsf{rew}(\mathcal{T})$, and $\mathcal{M}_c$ be the set of

- mapping assertions $\varphi^A(x) \rightsquigarrow A(x)$ such that A is a concept name in $\mathcal{T}_3$ and $\varphi^A \in \mathsf{cut}^\Omega_k(A, \Pi)$, and of
- mapping assertions $\varphi^P(x, y) \rightsquigarrow P(x, y)$ such that P is a role name in $\mathcal{T}_3$ and $\varphi^P \in \Phi_{\Pi_{\mathcal{T}_3,\mathcal{M}}}(P)$.

It is straightforward to see that $\langle\mathcal{T}_r, \mathcal{M}_c, \mathcal{S}\rangle$ is a CQ-rewriting of $\langle\mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$ into *DL-Lite*$_\mathcal{R}$.

Assume that Π is CQ-rewritable into *DL-Lite*$_\mathcal{R}$ and $\langle\mathcal{T}', \mathcal{M}', \mathcal{S}\rangle$ is its CQ-rewriting where the source queries in $\mathcal{M}'$ are unions of CQs. Let N be a concept or role name in $\mathcal{T}$ and denote by $q_N(\vec{x})$ the rewriting of the query $N(\vec{x})$ into a union of CQs over $\mathcal{S}$ with respect to $\langle\mathcal{T}', \mathcal{M}', \mathcal{S}\rangle$ (recall that the rewriting of $N(\vec{x})$ with respect to $\mathcal{T}'$ is a union of CQs, and since $\mathcal{M}'$ contains unions of CQs as source queries, $q_N(\vec{x})$ is also a union of CQs).

We construct now a Datalog program Π' consisting of the rules $N(\vec{x}) \leftarrow \varphi^N(\vec{x})$, for a concept or role name N in $\mathcal{T}$ and a CQ $\varphi^N(\vec{x}) \in q_N(\vec{x})$. Obviously, Π' is program bounded. Since $\langle\mathcal{T}', \mathcal{M}', \mathcal{S}\rangle$ is a CQ-rewriting of $\langle\mathcal{T}, \mathcal{M}, \mathcal{S}\rangle$, and $\mathcal{T}_3$ is a model-conservative extension of $\mathcal{T}$, we have that $N^\infty_{\Pi'}(\mathcal{D}) = N^\infty_{\Pi_{\mathcal{T},\mathcal{M}}}(\mathcal{D}) = N^\infty_{\Pi_{\mathcal{T}_3,\mathcal{M}}}(\mathcal{D})$, for each instance $\mathcal{D}$ of $\mathcal{S}$ and each concept or role name N in $\mathcal{T}$. Next, because of (2), we have that $A^\infty_{\Pi'}(\mathcal{D}) = A^\infty_\Pi(\mathcal{D})$ for each instance $\mathcal{D}$ of $\mathcal{S}$ and each concept name A in $\mathcal{T}$. Now, we set the finite union of CQs $\Omega_\Pi(A)$ for each concept name A in $\mathcal{T}_3$:

- if A is a concept name in $\mathcal{T}$, then $\Omega_\Pi(A) = \Phi_{\Pi'}(A)$
- otherwise, A is introduced for a concept conjunction $A_1 \sqcap \cdots \sqcap A_n$ in Step 3, then $\Omega_\Pi(A)$ is the DNF of the formula $\Phi_{\Pi'}(A_1) \wedge \cdots \wedge \Phi_{\Pi'}(A_n)$ where each $\Phi_{\Pi'}$ is viewed as a formula in DNF.

Hence, we obtain that Π is program bounded.

We now show how Π is constructed from $\Pi_{\mathcal{T}_3,\mathcal{M}}$.

First, we remove from $\Pi_{\mathcal{T}_3,\mathcal{M}}$ the rules which are not reachable from the database predicates. Namely, let Π_1 be the set of rules $\pi = \mathit{head} \leftarrow X_1, \ldots, X_n$ in $\Pi_{\mathcal{T}_3,\mathcal{M}}$ such that there are sets of rules $\rho_1, \ldots, \rho_m$ in $\Pi_{\mathcal{T}_3,\mathcal{M}}$ such that ρ_m is a set of rules from $\Pi_\mathcal{M}$, the predicates in the bodies of the rules in ρ_{i-1} are exactly the predicates in the heads of the rules in ρ_i, for $2 \le i \le m$, and $\rho_1 = \{\pi\}$. It should be clear that $N^\infty_{\Pi_1}(\mathcal{D}) = N^\infty_{\Pi_{\mathcal{T}_3,\mathcal{M}}}(\mathcal{D})$, for each instance $\mathcal{D}$ of $\mathcal{S}$ and each concept or role name N in $\mathcal{T}_3$.

Then Π is the monadic Datalog program such that for each instance $\mathcal{D}$ of $\mathcal{S}$ and each concept name A in $\mathcal{T}_3$, $A^\infty_\Pi(\mathcal{D}) = A^\infty_{\Pi_1}(\mathcal{D})$. We obtain Π by substituting each occurrence in the body of a rule of an atom of the form $R(x, y)$, for R a role in $\mathcal{T}$, by $\Phi_{\Pi_1}(R)$, and by removing all rules whose head predicates are roles. Namely for the former, let $\rho = \mathit{head} \leftarrow \varphi, R(x, y)$ be a rule in Π_1. Then we replace ρ with the rules, $\mathit{head} \leftarrow \varphi, \psi$, for each CQ $\psi \in \Phi_{\Pi_1}(R)$. We repeat this procedure until we get a Datalog program Π where no atom of the form $R(x, y)$, for a role R in $\mathcal{T}_3$, occurs in the a body of a rule. Observe that $\Phi_{\Pi_1}(R)$ is always finite.

It is easy to see that Π is as required. □