Virtual OBDA over Expressive Ontologies: Rewritings and Approximations*

E. Botoeva[1], D. Calvanese[1], V. Santarelli[2], D. F. Savo[2], A. Solimando[3], and G. Xiao[1]

[1] Free University of Bozen-Bolzano, Italy, *lastname*@inf.unibz.it
[2] Sapienza Università di Roma, Italy, *lastname*@dis.uniroma1.it
[3] DIBRIS, University of Genova, Italy, alessandro.solimando@unige.it

Abstract. In this paper we extend virtual OBDA to more expressive ontology languages than that of *DL-Lite*$_{\mathcal{R}}$, which is the logic underpinning the W3C ontology language OWL 2. We achieve this by relying on two well-known mechanisms, namely conservative rewriting and approximation, and compiling some of the domain semantics into OBDA mappings.

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 [7, 16]. Typically, the data sources are represented as relational data, the ontology is 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.

Making OBDA work efficiently over large amounts of data, requires that query answering over the ontology is *first-order (FO)-rewritable*[1] [8, 1], 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 [24], 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 [9]. For this reason, in many situations the expressive power of *DL-Lite*$_{\mathcal{R}}$ is too restricted to capture real-world scenarios.

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

* This paper is an abridged version of [5].
[1] Recall that FO queries constitute the core of SQL.

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 [20] and approximation [26, 10]. In the former, an ontology in a powerful language 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 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 $\mathsf{CAcc} \sqcap \exists \mathsf{inNameOf}.\mathsf{Person} \sqsubseteq \mathsf{SAcc}$. Further, assume that the information about the accounts and their owners is stored in a database $\mathcal{D}$, and that the predicates CAcc, 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 an SQL query over $\mathcal{D}$. Then this 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 [3]. 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 [27] and CLIPPER systems [14] 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. 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, producing 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.

Omitted proofs can be found in the extended version of this paper [4].

2 Preliminaries

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 OWL 2 fragment.

A Horn-$\mathcal{ALCHIQ}$ TBox in normal form [18, 14] is a finite set of axioms of the following forms: *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 $\leq 1\, R.A$. 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 $\exists R.\top$, we use N to denote either a concept or a role name, and 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). So, 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 $\varphi(\boldsymbol{x})$ to denote that query φ has $\boldsymbol{x} = x_1, \ldots, x_n$ 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 $\mathcal{S}$, called *source queries*. 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 (*virtual*[3]) ABox $\mathcal{A}_{\mathcal{M},\mathcal{D}} = \{N(\boldsymbol{o}) \mid \boldsymbol{o} \in ans(\varphi, \mathcal{D})$ and $\varphi(\boldsymbol{x}) \rightsquigarrow N(\boldsymbol{x})$ 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 $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},\mathcal{D}} \rangle$.

To abstract away the low-level SQL details in source queries, following [13], we split each mapping assertion $m = \varphi(\boldsymbol{x}) \rightsquigarrow N(\boldsymbol{x})$ in $\mathcal{M}$ into two parts. By introducing an intermediate view name V_m for the SQL query $\varphi(\boldsymbol{x})$, we obtain a *low-level* mapping assertion of the form $\varphi(\boldsymbol{x}) \rightsquigarrow V_m(\boldsymbol{x})$, and a *high-level* mapping assertion of the form $V_m(\boldsymbol{x}) \rightsquigarrow N(\boldsymbol{x})$. Hence, in our technical development we deal only with the high-level mappings, and directly consider the intermediate views as our data sources.

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) $q(\boldsymbol{x})$ over a signature Σ is a formula $\exists \boldsymbol{y}.\, \varphi(\boldsymbol{x}, \boldsymbol{y})$, where φ is a conjunction of atoms $N(\boldsymbol{z})$, such that N is a concept or role name in Σ, and $\boldsymbol{z}$ are variables from $\boldsymbol{x}$ and $\boldsymbol{y}$. The set of *certain answers* to a CQ $q(\boldsymbol{x})$ over an ontology $\langle \mathcal{T}, \mathcal{A} \rangle$, denoted $cert(q, \langle \mathcal{T}, \mathcal{A} \rangle)$, is the set of tuples $\boldsymbol{c}$ of elements from $\mathsf{Ind}(\mathcal{A})$ of the same length as $\boldsymbol{x}$, such that $q(\boldsymbol{c})$ (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.

[2] All our results easily extend to the case where objects are constructed from retrieved database values [7].

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

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 [6] to OBDA specifications. We adopt the proposal by [3], 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 in 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 efficient query answering. A possible solution is to exploit the expressive power of the mapping 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 [27], 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 [11], i.e., there exist CQ$^\neq$s $\varphi^N_0, \varphi^N_1, \ldots$ such that $N^\infty_\Pi(\mathcal{D}) = \bigcup_{i\geq 0}\{N(\boldsymbol{a}) \mid \boldsymbol{a} \in ans(\varphi^N_i, \mathcal{D})\}$, for every $\mathcal{D}$. The φ^N_i's are called the *expansions* of N and can be described in terms of expansion trees; cf. [4, Appendix A]. 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 [17], [14], and [28] for Horn-$\mathcal{SHIQ}$, and by [12] for $\mathcal{SHI}$). We assume that $\Pi_\mathcal{T}$ makes use of a special nullary predicate $\bot$ that encodes inconsistency, i.e., for an ABox $\mathcal{A}$, $\langle\mathcal{T}, \mathcal{A}\rangle$ is consistent iff $\bot^\infty_{\Pi_\mathcal{T}}(\mathcal{A})$ 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:

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

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:

[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 $N \sqsubseteq N$).

Fig. 1. OBDA specification rewriting procedure RewObda.

Lemma 1. *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 $\boldsymbol{a}$ in $\mathsf{Ind}(\mathcal{A}_{\mathcal{M},\mathcal{D}})$, we have that $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},\mathcal{D}} \rangle \models N(\boldsymbol{a})$ iff $N(\boldsymbol{a}) \in N^\infty_{\Pi_{\mathcal{T},\mathcal{M}}}(\mathcal{D})$.*

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 3. *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 $\varphi^N(\boldsymbol{x}) \rightsquigarrow N(\boldsymbol{x})$ 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. Procedure $\mathsf{RewObda}(\mathcal{T}, \mathcal{M})$, in Figure 1, 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 procedure applies to $\mathcal{T}_1$ the normalization step $\mathsf{norm}_\exists$, 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 [1]: every CI $\bigsqcap_{i=1}^m A_i \sqsubseteq \exists R.(\bigsqcap_{j=1}^n A'_j)$ in $\mathcal{T}_1$ is replaced with $\bigsqcap_{i=1}^m A_i \sqsubseteq \exists P_{new}$, $P_{new} \sqsubseteq R$, 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 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 procedure applies to $\mathcal{T}_2$ a further normalization step, $\mathsf{norm}_\sqcap$, 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 $\mathsf{norm}_\exists(\mathcal{T}_1)$

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

and $\mathsf{norm}_{\sqcap}(\mathcal{T}_2)$ are model-conservative extensions of $\mathcal{T}_1$ and $\mathcal{T}_2$, respectively [21], 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 1 *Assume that the domain knowledge is represented by the TBox* $\mathcal{T} = \{\ \mathsf{CAcc} \sqcap \exists \mathsf{inNameOf}.\mathsf{Person} \sqsubseteq \mathsf{SAcc}\ \}$ *about bank accounts. Its normalization is* $\mathcal{T}^b = \{\mathsf{Person} \sqsubseteq \forall \mathsf{inNameOf}^{-}.A_1, \mathsf{CAcc} \sqcap A_1 \sqsubseteq \mathsf{SAcc}\}$. *Assume that the database schema* $\mathcal{S}^b$ *consists of the two relations* $\mathsf{ENT}(\mathit{ID}, \mathit{TYPE}, \mathit{EMPID})$, $\mathsf{PROD}(\mathit{NUM}, \mathit{TYPE}, \mathit{CUSTID})$, *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'` $\rightsquigarrow$ $\mathsf{Person}(X)$
m_{N}: `SELECT NUM AS X, CUSTID AS Y FROM PROD` $\rightsquigarrow$ $\mathsf{inNameOf}(X, Y)$
m_{C}: `SELECT NUM AS X FROM PROD P WHERE P.TYPE='B'` $\rightsquigarrow$ $\mathsf{CAcc}(X)$

The corresponding high-level mapping $\mathcal{M}^b$ *consists of the assertions:*

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

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

- *The intermediate TBoxes* $\mathcal{T}_1^b$ *and* $\mathcal{T}_2^b$ *coinciding with* $\mathcal{T}^b$, *and* $\mathcal{T}_3^b$ *extending* $\mathcal{T}^b$ *with* $A_{\mathsf{CAcc} \sqcap A_1} \equiv \mathsf{CAcc} \sqcap A_1$.
- *The ET-mapping* $\mathcal{M}_c^b = \mathsf{etm}_{\mathcal{T}_3^b}(\mathcal{M}^b)$, *which extends* $\mathcal{M}^b$ *with the following three assertions:* $\{x \mid V_{\mathsf{inNameOf}}(x, y), V_{\mathsf{Person}}(y)\} \rightsquigarrow A_1(x)$;
  $\{x \mid V_{\mathsf{CAcc}}(x), V_{\mathsf{inNameOf}}(x, y), V_{\mathsf{Person}}(y)\} \rightsquigarrow \mathsf{SAcc}(x)$;
  $\{x \mid V_{\mathsf{CAcc}}(x), V_{\mathsf{inNameOf}}(x, y), V_{\mathsf{Person}}(y)\} \rightsquigarrow A_{\mathsf{CAcc} \sqcap A_1}(x)$.

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

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 $\mathcal{T}$, tailored towards capturing in *DL-Lite*$_{\mathcal{R}}$ the answers to tree-shaped CQs. This is obtained by introducing in Step 2 sufficiently many new role names, and in Step 3 new concept names, so as to capture entailed axioms that generate the tree-shaped parts of models. Note that at-most number restrictions do not participate in the generation of the tree-shaped parts of models, hence are not considered explicitly in Steps 1 to 3. On the other hand, the ET-mapping $\mathcal{M}_c = \mathsf{comp}(\mathcal{T}, \mathcal{M})$ 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 2. *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 $N^k_\Pi(\mathcal{D}) = N^\infty_\Pi(\mathcal{D})$ [11]. 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 [15]. 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:

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

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 $\varphi^N(\boldsymbol{x}) \rightsquigarrow N(\boldsymbol{x})$ 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 $\Pi_\mathcal{T}$ [22, 2].

Theorem 3. *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 $\mathcal{M}' = \mathsf{cut}^\Omega_k(\mathsf{comp}(\mathcal{T}, \mathcal{M}))$, 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$_\mathcal{R}$.*

This gives us decidable conditions for rewritability of OBDA specifications in several significant cases. It is shown by [2] and [22] 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 $N(\boldsymbol{x})$ 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 4. *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, it follows from decidability of boundedness of monadic Datalog programs [11] that we can fully characterize CQ-rewritability.

Theorem 5. *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

To demonstrate the feasibility of our OBDA specification rewriting technique, we implemented the ONTOPROX[6] prototype system and evaluated it over synthetic and real OBDA instances. It relies on the OBDA reasoner ONTOP[7] and the complete Horn-$\mathcal{SHIQ}$ CQ-answering system CLIPPER[8], used as Java libraries, 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.

[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/

For the experiments, we have considered two scenarios:

UOBM. The university ontology benchmark (UOBM) [23] 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 suitable 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].

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.

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 [10].
(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 $\mathcal{T}$ [25].
(4) Result of ONTOPROX, i.e., $\langle\mathsf{rew}(\mathcal{T}),\mathsf{cut}_5(\mathsf{comp}(\mathcal{T},\mathcal{M})),\mathcal{S}\rangle$.
(5) CLIPPER over the materialization of the virtual ABox.

The details of the evaluation can be found in [4].

7 Conclusions

We proposed a novel framework for rewriting and approximating 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 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 [12]; *(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 [19]; *(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 work is partially 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.

[11] https://github.com/ontop/ontop-examples/tree/master/aaai-2016-ontoprox/uobm

References

1. Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. The *DL-Lite* family and relations. *J. of Artificial Intelligence Research*, 36:1–69, 2009.
2. Meghyn Bienvenu, Carsten Lutz, and Frank Wolter. First-order rewritability of atomic queries in Horn description logics. In *Proc. of the 23rd Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 754–760, 2013.
3. Meghyn Bienvenu and Riccardo Rosati. Query-based comparison of OBDA specifications. In *Proc. of the 28th Int. Workshop on Description Logic (DL)*, volume 1350 of *CEUR Electronic Workshop Proceedings*, 2015.
4. Elena Botoeva, Diego Calvanese, Valerio Santarelli, Domenico Fabio Savo, Alessandro Solimando, and Guohui Xiao. Beyond OWL 2 QL in OBDA: Rewritings and approximations (Extended version). CoRR Technical Report abs/1511.08412, arXiv.org e-Print archive, 2015. Available at `http://arxiv.org/abs/1511.08412`.
5. Elena Botoeva, Diego Calvanese, Valerio Santarelli, Domenico Fabio Savo, Alessandro Solimando, and Guohui Xiao. Beyond OWL 2 QL in OBDA: Rewritings and approximations. In *Proc. of the 30th AAAI Conf. on Artificial Intelligence (AAAI)*, 2016.
6. Elena Botoeva, Roman Kontchakov, Vladislav Ryzhikov, Frank Wolter, and Michael Zakharyaschev. Query inseparability for description logic knowledge bases. In *Proc. of the 14th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR)*, pages 238–247. AAAI Press, 2014.
7. Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Mariano Rodriguez-Muro, and Riccardo Rosati. Ontologies and databases: The *DL-Lite* approach. In *5th Reasoning Web Int. Summer School Tutorial Lectures (RW)*, volume 5689 of *LNCS*, pages 255–356. Springer, 2009.
8. 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.
9. Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Data complexity of query answering in description logics. *Artificial Intelligence*, 195:335–360, 2013.
10. Marco Console, José Mora, Riccardo Rosati, Valerio Santarelli, and Domenico Fabio Savo. Effective computation of maximal sound approximations of description logic ontologies. In *Proc. of the 13th Int. Semantic Web Conf. (ISWC)*, volume 8797 of *LNCS*, pages 164–179. Springer, 2014.
11. Stavros S. Cosmadakis, Haim Gaifman, Paris C. Kanellakis, and Moshe Y. Vardi. Decidable optimization problems for database logic programs. In *Proc. of the 20th ACM SIGACT Symp. on Theory of Computing (STOC)*, pages 477–490, 1988.
12. Bernardo Cuenca Grau, Boris Motik, Giorgos Stoilos, and Ian Horrocks. Computing datalog rewritings beyond Horn ontologies. In *Proc. of the 23rd Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 832–838, 2013.
13. Floriana Di Pinto, Domenico Lembo, Maurizio Lenzerini, Riccardo Mancini, Antonella Poggi, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. Optimizing query rewriting in ontology-based data access. In *Proc. of the 16th Int. Conf. on Extending Database Technology (EDBT)*, pages 561–572. ACM Press, 2013.
14. Thomas Eiter, Magdalena Ortiz, Mantas Simkus, Trung-Kien Tran, and Guohui Xiao. Query rewriting for Horn-SHIQ plus rules. In *Proc. of the 26th AAAI Conf. on Artificial Intelligence (AAAI)*, pages 726–733. AAAI Press, 2012.
15. Haim Gaifman, Harry G. Mairson, Yehoshua Sagiv, and Moshe Y. Vardi. Undecidable optimization problems for database logic programs. In *Proc. of the 2nd IEEE Symp. on Logic in Computer Science (LICS)*, pages 106–115, 1987.

16. Martin Giese, Ahmet Soylu, Guillermo Vega-Gorgojo, Arild Waaler, Peter Haase, Ernesto Jiménez-Ruiz, Davide Lanti, Martin Rezk, Guohui Xiao, Özgür L. Özçep, and Riccardo Rosati. Optique: Zooming in on Big Data. *IEEE Computer*, 48(3):60–67, 2015.
17. Ullrich Hustadt, Boris Motik, and Ulrike Sattler. Data complexity of reasoning in very expressive description logics. In *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 466–471, 2005.
18. Yevgeny Kazakov. Consequence-driven reasoning for Horn-SHIQ ontologies. In *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 2040–2045, 2009.
19. Roman Kontchakov, Martin Rezk, Mariano Rodriguez-Muro, Guohui Xiao, and Michael Zakharyaschev. Answering SPARQL queries over databases under OWL 2 QL entailment regime. In *Proc. of the 13th Int. Semantic Web Conf. (ISWC)*, LNCS. Springer, 2014.
20. Carsten Lutz, Robert Piro, and Frank Wolter. Description logic TBoxes: Model-theoretic characterizations and rewritability. In *Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 983–988, 2011.
21. Carsten Lutz, Dirk Walther, and Frank Wolter. Conservative extensions in expressive description logics. In *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI)*, pages 453–458, 2007.
22. Carsten Lutz and Frank Wolter. 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*, 2011.
23. Li Ma, Yang Yang, Zhaoming Qiu, GuoTong Xie, Yue Pan, and Shengping Liu. Towards a complete OWL ontology benchmark. In *Proc. of the 3rd European Semantic Web Conf. (ESWC)*, volume 4011 of *LNCS*, pages 125–139. Springer, 2006.
24. Boris Motik, Achille Fokoue, Ian Horrocks, Zhe Wu, Carsten Lutz, and Bernardo Cuenca Grau. OWL Web Ontology Language profiles. W3C Recommendation, World Wide Web Consortium, October 2009. Available at `http://www.w3.org/TR/owl-profiles/`.
25. Jeff Z. Pan and Edward Thomas. Approximating OWL-DL ontologies. In *Proc. of the 21st AAAI Conf. on Artificial Intelligence (AAAI)*, pages 1434–1439, 2007.
26. Yuan Ren, Jeff Z. Pan, and Yuting Zhao. Soundness preserving approximation for TBox reasoning. In *Proc. of the 24th AAAI Conf. on Artificial Intelligence (AAAI)*, 2010.
27. Mariano Rodriguez-Muro, Roman Kontchakov, and Michael Zakharyaschev. Ontology-based data access: Ontop of databases. In *Proc. of the 12th Int. Semantic Web Conf. (ISWC)*, volume 8218 of *LNCS*, pages 558–573. Springer, 2013.
28. Despoina Trivela, Giorgos Stoilos, Alexandros Chortaras, and Giorgos Stamou. Optimising resolution-based rewriting algorithms for OWL ontologies. *J. of Web Semantics*, pages 30–49, 2015.