Query-based comparison of OBDA specifications

Meghyn Bienvenu[1] and Riccardo Rosati[2]

[1] Laboratoire de Recherche en Informatique
CNRS & Université Paris-Sud, France

[2] Dipartimento di Ingegneria informatica, automatica e gestionale
Sapienza Università di Roma, Italy

Abstract. An ontology-based data access (OBDA) system is composed of one or more data sources, an ontology that provides a conceptual view of the data, and declarative mappings that relate the data and ontology schemas. In order to debug and optimize such systems, it is important to be able to analyze and compare OBDA specifications. Recent work in this direction compared specifications using classical notions of equivalence and entailment, but an interesting alternative is to consider query-based notions, in which two specifications are deemed equivalent if they give the same answers to the considered query or class of queries for all possible data sources. In this paper, we define such query-based notions of entailment and equivalence of OBDA specifications and investigate the complexity of the resulting analysis tasks when the ontology is formulated in *DL-Lite*$_R$.

1 Introduction

Ontology-based data access (OBDA) [13] is a recent paradigm that proposes the use of an *ontology* as a conceptual, reconciled view of the information stored in a set of existing *data sources*. The connection between the ontology and the data sources is provided by declarative *mappings*, that relate the elements of the ontology with the elements of the data sources. The ontology layer is the virtual interface used to access data, through *queries* over the elements of the ontology.

Due to the recent availability of techniques and systems for query processing in this setting [5, 14], the OBDA approach has recently started to be experimented in real applications (see e.g. [1, 7, 10]). In these projects, the construction, debugging and maintenance of the OBDA specification, consisting of the ontology, the schemas of the data sources, and the mapping, is a non-trivial task. Actually, the size and the complexity of the ontology and, especially, the mappings makes the management of such specifications a practical issue in these projects. Providing formal tools for supporting the above activities is therefore very important for the successful deployment of OBDA solutions.

In addition, the OBDA specification plays a major role in query answering, since the form of the specification may affect the system performance in answering queries: different, yet semantically equivalent specifications may give rise to very different execution times for the same query. So, the study of notions of equivalence and formal comparison of OBDA specifications is also important for optimizing query processing in OBDA systems. Indeed, some systems already implement forms of optimization based on transformations of the OBDA specification (an example is [14]).

So far, most of the work in OBDA has focused on query answering, often in a simplified setting without any mappings. Very little attention has been devoted to the formal analysis of OBDA specifications. The first approach that explicitly focuses on the formal analysis of OBDA specifications is [12], whose aim is the identification of semantic anomalies in mappings. Such an approach is based on a classical notion of logical equivalence and entailment between OBDA specifications. While it is very natural to resort to such classical notions, a significant alternative in many cases may be the adoption of *query-based* notions of equivalence and comparison, in which two specifications are compared with respect to a given query or a given class of queries, and are deemed equivalent if they give the same answers to the considered queries for all possible extensions of the data sources. This idea has been already explored in the data exchange and schema mapping literature (see, e.g., [9]) and for description logics for comparing TBoxes and knowledge bases [11, 4]. To the best of our knowledge, it has never been explicitly considered for OBDA specifications.

The majority of work on on OBDA has considered *conjunctive queries (CQs)* as the query language. Therefore, a first natural choice would be to compare OBDA specifications with respect to the whole class of CQs. We thus define and study a notion of *CQ-entailment* between OBDA specifications that formalizes this case. We also consider the important subclass of *instance queries (IQs)*, i.e., queries that ask for the instances of a single concept or role, and analyze the notion of *IQ-entailment* between specifications. Moreover, in many application contexts only a (small) set of predefined conjunctive queries are of interest for the OBDA user(s): in such cases, it may be more appropriate to tailor the comparison of specifications to a specific set of queries. For this reason, we also study in this paper the notions of *single CQ-entailment* and *single IQ-entailment*, which compare specifications with respect to a single CQ or IQ, respectively.

We present a first investigation of the computational complexity of deciding the above forms of entailment for a pair of OBDA specifications. We study ontologies specified in *DL-Lite*$_R$ and three different mapping languages (linear, GAV and GLAV). In all cases, we provide exact complexity bounds for the entailment problem. Our results are summarized in Figure 1. As shown in the table, the complexity of the entailment check ranges from NL (non-deterministic logarithmic space) for linear mappings and IQ-entailment to EXPTIME for CQ-entailment. To obtain these results, we show that instead of considering all possible data instances, it is sufficient to consider a small number of databases of a particular form. We also exploit connections to query containment in the presence of signature restrictions [3] and KB query inseparability [4].

2 Preliminaries

We start from four pairwise disjoint countably infinite set of names: the set of concept names $\mathsf{N_C}$, the set of role names $\mathsf{N_R}$, the set of relation names $\mathsf{N_{rel}}$, the set of constant names $\mathsf{N_I}$ (also called individuals).

To introduce OBDA specifications, we first recall the notion of knowledge base (KB) in description logics (DLs). A DL KB is a pair $\langle \mathcal{T}, \mathcal{A} \rangle$, where: $\mathcal{T}$, called the *TBox*, is the intensional component of the KB, and is constituted by a finite set of axioms expressing intensional knowledge; and $\mathcal{A}$, called the *ABox*, is a finite set of atomic concept and role assertions (set of ground facts). We assume that the concept,

role and constant names occurring in every TBox and ABox belong to $\mathsf{N_C}$, $\mathsf{N_R}$ and $\mathsf{N_I}$, respectively. We denote by $\mathsf{sig}(\mathcal{T})$ and $\mathsf{sig}(\mathcal{A})$ the set of concept and role names occurring in $\mathcal{T}$ and $\mathcal{A}$, respectively.

Although the definitions of Section 3 are general, in Section 4 we will focus on the DL *DL-Lite*$_R$ [6]. A *DL-Lite*$_R$ TBox consists of a finite set of concept inclusions $B \sqsubseteq C$ and role inclusions $R \sqsubseteq S$, where B, C, R, and S are defined according to the following syntax (where A is a concept name and P is a role name):

$$B \to A \mid \exists R \qquad C \to B \mid \neg B \qquad R \to P \mid P^- \qquad S \to R \mid \neg R$$

We now introduce OBDA specifications. As already explained, a mapping assertion specifies the semantic relationship between elements of a DL ontology, specified through a TBox, to elements of a database. Such a relationship is specified through a pair of queries, one over the TBox signature, and the other one over the database signature. In this paper, we focus on the case where both queries involved in the mapping assertion are conjunctive queries: such mapping assertions are called GLAV (for 'global-as-view') mappings in the literature [8].

Mappings are formally defined as follows. An *atom* is an expression $r(\boldsymbol{t})$ where r is a predicate and $\boldsymbol{t}$ is a tuple of variables and constants. Then, a *(GLAV) mapping assertion* m is an expression of the form $q_s(\boldsymbol{x}) \to q_o(\boldsymbol{x})$, where $q_s(\boldsymbol{x})$ (called the *body* of m, $body(m)$) is a conjunction of atoms over predicates from $\mathsf{N_{rel}}$ and constants from $\mathsf{N_I}$, $q_o(\boldsymbol{x})$ (called the *head* of m, $head(m)$) is a conjunction of atoms using predicates from $\mathsf{N_C} \cup \mathsf{N_R}$ and constants from $\mathsf{N_I}$, and $\boldsymbol{x}$, called the *frontier variables* of m, are the variables that appear both in q_o and in q_s. The *arity* of m is the number of its frontier variables. When $q_o(\boldsymbol{x})$ has the form $p(\boldsymbol{x})$ (i.e., $q_o(\boldsymbol{x})$ is a single atom whose arguments are $\boldsymbol{x}$), we call m a *GAV* mapping assertion. A *linear* mapping assertion is a GAV assertion whose body consists of a single atom. A *(GLAV) mapping* $\mathcal{M}$ is a set of mapping assertions. A *GAV mapping* is a mapping constituted of GAV mapping assertions. A *linear mapping* is a set of linear mapping assertions. Without loss of generality, we assume that in every mapping $\mathcal{M}$, every pair of distinct mapping assertions uses pairwise disjoint sets of variables.

An *OBDA specification* is a pair $\Gamma = \langle \mathcal{T}, \mathcal{M} \rangle$, where $\mathcal{T}$ is a TBox and $\mathcal{M}$ is a mapping. Given a mapping assertion m of arity n and an n-tuple of constants $\boldsymbol{a}$, we denote by $m(\boldsymbol{a})$ the assertion obtained from m by replacing the frontier variables with the constants in $\boldsymbol{a}$.

Given a set of atoms AT, the function gr returns a set $gr(AT)$ of ground atoms obtained from AT by replacing every variable symbol x with a fresh constant symbol c_x that does not occur in the considered mapping or database. We assume without loss of generality that if $AT \neq AT'$, then $gr(AT)$ and $gr(AT')$ use distinct fresh constants.

In this paper, a *database* (instance) is a set of ground atoms using relation names from $\mathsf{N_{rel}}$ and constant names from $\mathsf{N_I}$. Given a mapping $\mathcal{M}$ and a database instance D, we define the *ABox for D and $\mathcal{M}$*, denoted as $\mathcal{A}_{\mathcal{M},D}$, as the following ABox:

$$\{\ \beta \in gr(head(m(\boldsymbol{a}))) \mid m \in \mathcal{M} \text{ and } D \models \exists \boldsymbol{y}.body(m(\boldsymbol{a}))\ \}$$

where we assume that $\boldsymbol{y}$ are the variables occurring in $body(m(\boldsymbol{a}))$. Given an OBDA specification $\Gamma = \langle \mathcal{T}, \mathcal{M} \rangle$ and a database instance D, we define the *models of Γ and*

D, denoted as $Mods(\Gamma, D)$ as the set of models of the KB $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},D} \rangle$. When such a set is empty, we write $\langle \mathcal{T}, \mathcal{M}, D \rangle \models \bot$ (analogously, when a KB $\langle \mathcal{T}, \mathcal{A} \rangle$ has no models, we write $\langle \mathcal{T}, \mathcal{A} \rangle \models \bot$).

We are interested in the problem of answering instance queries and conjunctive queries over a pair composed of an OBDA specification and a database. A *Boolean conjunctive query (CQ)* is an expression of the form $\exists \mathbf{x}(\alpha_1 \wedge \ldots \wedge \alpha_n)$ where every α_i is an atom whose arguments are either constants or variables from $\mathbf{x}$. For a non-Boolean CQ q with answer variables $v_1, \ldots, v_k$, a tuple of constants $\boldsymbol{a} = \langle a_1, \ldots, a_k \rangle$ occurring in $\mathcal{A}$ is said to be a *certain answer* for q w.r.t. $\mathcal{K}$ just in the case that $\mathcal{K} \models q(\boldsymbol{a})$, where $q(\boldsymbol{a})$ is the Boolean query obtained from q by replacing each v_i by a_i. We call *instance query (IQ)* a CQ consisting of a single atom of the form $A(x)$ or $R(x, y)$, with A concept name, R role name, and x, y distinct free variables. We denote by $\mathsf{sig}(q)$ the set of concept and role names occurring in a query q. We use CQ (resp. IQ) to refer the set of all CQs (resp. IQs) over the DL signature $\mathsf{N_C} \cup \mathsf{N_R}$.

Given an OBDA specification $\Gamma = \langle \mathcal{T}, \mathcal{M} \rangle$, a database instance D, and a conjunctive query q, we define the certain answers for q w.r.t. (Γ, D) as the tuples of constants from D that are certain answers for q w.r.t. $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},D} \rangle$. In particular, for Boolean CQs, we say that q is entailed by (Γ, D), denoted by $(\Gamma, D) \models q$ (or $\langle \mathcal{T}, \mathcal{M}, D \rangle \models q$), if $\mathcal{I} \models q$ for every $\mathcal{I} \in Mods(\Gamma, D)$. Note that for non-Boolean queries, we only consider tuples of constants from D, in order to avoid including those fresh constants introducing in $\mathcal{A}_{\mathcal{M},D}$ by grounding existential variables in mapping heads.

3 Query-based Entailment for OBDA Specifications

We start by recalling the classical notion of entailment between OBDA specifications.

Definition 1 (Logical entailment). *An OBDA specification $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle$* logically entails *$\langle \mathcal{T}_2, \mathcal{M}_2 \rangle$, written $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\mathsf{log}} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ if and only the first-order theory $\mathcal{T}_1 \cup \mathcal{M}_1$ logically entails the first-order theory $\mathcal{T}_2 \cup \mathcal{M}_2$.*

We now define the formal notions of query-based entailment between OBDA specifications considered in this paper. First, we introduce a notion of entailment that compares specifications based upon the constraints they impose regarding consistency.

Definition 2 ($\bot$-entailment). *Let q be a query. An OBDA specification $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle$* $\bot$-entails *$\langle \mathcal{T}_2, \mathcal{M}_2 \rangle$, written $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\bot} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$, iff, for every database D,*

$$\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models \bot \quad \Rightarrow \quad \langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models \bot$$

Next, we define a notion of query entailment between OBDA specifications with respect to a *single* query.

Definition 3 (Single query entailment). *Let q be a query. An OBDA specification $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle$* q-entails *$\langle \mathcal{T}_2, \mathcal{M}_2 \rangle$, written $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_q \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$, if and only if $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\bot} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ and for every database D,*

$$\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models q(\boldsymbol{a}) \quad \Rightarrow \quad \langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models q(\boldsymbol{a})$$

When q is an IQ, we call the entailment relation in the preceding definition *single IQ-entailment*, while we call it *single CQ-entailment* if q is an arbitrary CQ.

We can generalize the previous definition to classes of queries as follows.

Definition 4 (Query entailment). *Let $\mathcal{L}$ be a (possibly infinite) set of queries. An OBDA specification $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle$ $\mathcal{L}$-*entails *$\langle \mathcal{T}_2, \mathcal{M}_2 \rangle$, written $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\mathcal{L}} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ iff $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\perp} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ and $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{q} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ for every query $q \in \mathcal{L}$.*

When $\mathcal{L} = \mathsf{IQ}$, we call the preceding entailment relation *IQ-entailment*, and for $\mathcal{L} = \mathsf{CQ}$, we use the term *CQ-entailment*.

Note that, for each of the above notions of entailment, a notion of equivalence between OBDA specifications can be immediately derived, corresponding to entailment in both directions (we omit the formal definitions due to space limitations).

The following property immediately follows from the above definitions.

Proposition 1. *Let $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle, \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ be two OBDA specifications, and let $\mathcal{L}_1$ be a set of queries. Then, $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\mathsf{log}} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ implies $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\mathcal{L}_1} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$. Moreover, if $\mathcal{L}_2 \subseteq \mathcal{L}_1$, then $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\mathcal{L}_1} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ implies $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\mathcal{L}_2} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$.*

As a consequence of the above property, we have that logical entailment implies CQ-entailment, and CQ-entailment implies IQ-entailment. The converse implications do not hold, as the following examples demonstrate.

Example 1. We start by illustrating the difference between logical entailment and CQ-entailment. Consider a database containing instances for the relation *EXAM*(*studentName,courseName,grade,date*). Then, let $\Gamma_1 = \langle \mathcal{T}_1, \mathcal{M}_1 \rangle$, where

$$\mathcal{T}_1 = \{\textit{Student} \sqsubseteq \textit{Person},\ \textit{PhDStudent} \sqsubseteq \textit{Student}\}$$
$$\mathcal{M}_1 = \{\textit{EXAM}(x, y, z, w) \rightarrow \textit{Student}(x)\}$$

and let $\Gamma_2 = \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$, where $\mathcal{T}_2 = \{\textit{Student} \sqsubseteq \textit{Person}\}$ and $\mathcal{M}_2 = \mathcal{M}_1$. It is immediate to verify that $\Gamma_2 \not\models_{\mathsf{log}} \Gamma_1$. However, we have that $\Gamma_2 \models_{\mathsf{CQ}} \Gamma_1$. Indeed, $\Gamma_2 \models_{\mathsf{CQ}} \Gamma_1$ can be intuitively explained by the fact that the mapping $\mathcal{M}_1$ does not retrieve any instances of the concept *PhDStudent* (and there are no subclasses that can indirectly populate it), so the presence of the inclusion *PhDStudent* $\sqsubseteq$ *Student* in $\mathcal{T}_1$ does not have any effect on query answering; in particular, every CQ that mentions the concept *PhDStudent* cannot be entailed both under Γ_1 and under Γ_2. Notice also that, if we modify the mapping $\mathcal{M}_1$ to map *PhDStudent* instead of *Student* (i.e., if $\mathcal{M}_1$ were $\{\textit{EXAM}(x, y, z, w) \rightarrow \textit{PhDStudent}(x)\}$), then CQ-entailment between Γ_2 and Γ_1 would no longer hold.

Next, consider $\Gamma_3 = \langle \mathcal{T}_3, \mathcal{M}_3 \rangle$, where $\mathcal{T}_3 = \emptyset$ and

$$\mathcal{M}_3 = \{\textit{EXAM}(x, y, z, w) \rightarrow \textit{Student}(x),\ \ \textit{EXAM}(x, y, z, w) \rightarrow \textit{Person}(x)\}$$

Again, it it immediate to see that $\Gamma_3 \not\models_{\mathsf{log}} \Gamma_2$, while we have that $\Gamma_3 \models_{\mathsf{CQ}} \Gamma_2$. Indeed, $\Gamma_3 \models_{\mathsf{CQ}} \Gamma_2$ follows informally from the fact that the mapping $\mathcal{M}_3$ is able to "extensionally" simulate the inclusion *Student* $\sqsubseteq$ *Person* of $\mathcal{T}_2$, which is sufficient for Γ_3 to entail every CQ in the same way as Γ_2.

Example 2. We slightly modify the previous example to show the difference between CQ-entailment and IQ-entailment. Consider $\Gamma_1 = \langle \mathcal{T}_1, \mathcal{M} \rangle$ and $\Gamma_2 = \langle \mathcal{T}_2, \mathcal{M} \rangle$ where

$$\mathcal{T}_1 = \{\textit{Student} \sqsubseteq \textit{Person},\ \textit{Student} \sqsubseteq \exists \textit{takesCourse}\}$$
$$\mathcal{T}_2 = \{\textit{Student} \sqsubseteq \textit{Person}\}$$
$$\mathcal{M} = \{\textit{EXAM}(x, y, z, w) \rightarrow \textit{Student}(x)\}$$

Type of entailment	Type of mapping	Complexity
logical	GAV / GLAV	NP-complete
	linear	NL-complete
$\bot$	GAV / GLAV	NP-complete
	linear	NL-complete
CQ	GLAV	EXPTIME-hard, in 2EXPTIME
	linear / GAV	EXPTIME-complete
IQ	linear	NL-complete
	GAV / GLAV	NP-complete
single CQ	linear / GAV / GLAV	Π_2^p-complete
single IQ	linear	NL-complete
	GAV / GLAV	NP-complete

Fig. 1. Complexity results for entailment between OBDA specifications in *DL-Lite*$_R$

Then, it can be easily verified that $\Gamma_2 \not\models_{\mathsf{CQ}} \Gamma_1$. Indeed, consider the Boolean CQ $\exists x, y\, \mathit{takesCourse}(x, y)$: for every database D, this query is not entailed by the pair (Γ_2, D), while this is not the case when the specification is Γ_1. On the other hand, we have that $\Gamma_2 \models_{\mathsf{IQ}} \Gamma_1$: in particular, for every database D and for every pair of individuals a, b, neither (Γ_1, D) nor (Γ_2, D) entails the IQ $\mathit{takesCourse}(a, b)$. Finally, let q be the non-Boolean CQ $\exists x\, \mathit{takesCourse}(x, y)$: then, it can be easily verified that the single CQ-entailment $\Gamma_2 \models_q \Gamma_1$ holds; while for the CQ q' of the form $\exists y\, \mathit{takesCourse}(x, y)$, the single CQ-entailment $\Gamma_2 \models_{q'} \Gamma_1$ does not hold.

4 Complexity Results for *DL-Lite*$_R$

In this section, we investigate the computational properties of the different notions of entailment between OBDA specifications defined in the previous section. For this first study, we focus on the case in which the TBox is formulated in *DL-Lite*$_R$ [6], as it is the basis for the OWL 2 QL profile and one of the most commonly considered DLs for OBDA. The results of our complexity analysis are displayed in Figure 1.

In what follows, we formally state the different complexity results and provide some ideas about the proofs. We begin by considering the complexity of deciding classical entailment between OBDA specifications.

Theorem 1. *Classical logical entailment for OBDA specifications based upon DL-Lite*$_R$ *TBoxes is* NP*-complete for GAV or GLAV mappings, and NL-complete for linear mappings.*

Proof. Let $\Gamma_1 = \langle \mathcal{T}_1, \mathcal{M}_1 \rangle$, $\Gamma_2 = \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$. First, it is easy to see that $\Gamma_1 \models_{\mathsf{log}} \Gamma_2$ iff (i) $\mathcal{T}_1 \models \mathcal{T}_2$; and (ii) $\Gamma_1 \models_{\mathsf{log}} \mathcal{M}_2$. Property (i) can be decided in NL [2]. Property (ii) can be decided by an algorithm that, for every assertion $m \in \mathcal{M}_2$, first builds a database D corresponding to $gr(body(m))$ (i.e., obtained by "freezing" the body of m), and then checks whether $\langle \Gamma_1, D \rangle$ entails the CQ corresponding to the head of m whose frontier variables have been replaced by the corresponding constants. This algorithm runs in

NP in the case of GAV and GLAV mappings, and in NL in the case of linear mappings, which implies the overall upper bounds in the theorem statement. The lower bound for GAV mappings can be obtained through an easy reduction of conjunctive query containment to logical entailment, while the one for linear mappings follows from a reduction of the entailment of a concept inclusion axiom in a *DL-Lite*$_R$ TBox. □

We next consider $\bot$-entailment. Our upper bounds rely on the following result that shows it is sufficient to consider a small number of small databases.

Theorem 2. *Let q be a CQ, and let $\Gamma_1 = \langle \mathcal{T}_1, \mathcal{M}_1 \rangle$ and $\Gamma_2 = \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ be OBDA specifications such that $\mathcal{T}_1, \mathcal{T}_2$ are formulated in DL-Lite$_R$, and $\mathcal{M}_1, \mathcal{M}_2$ are GLAV mappings. Then $\Gamma_1 \models_\bot \Gamma_2$ if and only if $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models \bot$ for every database D satisfying the following condition:*[1]

- *Condition 1: D is obtained by (i) taking two mapping assertions m_1, m_2 from $\mathcal{M}_2$, (ii) selecting atoms α_1 and α_2 from $head(m_1)$ and $head(m_2)$ respectively, (iii) identifying in m_1 and m_2 some variables from α_1 and α_2 in such a way that $\langle \mathcal{T}, gr(\{\alpha_1, \alpha_2\}) \rangle \models \bot$, (iv) setting D equal to $gr(body(m_1) \cup body(m_2))$.*

Proof. The one direction is immediate from the definitions. For the interesting direction, let us suppose that $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models \bot$ for every database D satisfying Condition 1. Let us further suppose that we have $\langle \mathcal{T}_2, \mathcal{M}_2, D_0 \rangle \models \bot$, where D_0 may be any database. We thus have $\langle \mathcal{T}_2, \mathcal{A}_{\mathcal{M}_2, D_0} \rangle \models \bot$. It is well known that every minimal inconsistent subset of a *DL-Lite*$_R$ KB contains at most two ABox assertions, so there must exist a subset $\mathcal{A}' \subseteq \mathcal{A}_{\mathcal{M}_2, D}$ with $|\mathcal{A}'| \leq 2$ such that $\langle \mathcal{T}_2, \mathcal{A}' \rangle \models \bot$. Let γ be the conjunction of atoms obtained by taking for each ABox assertion in $\mathcal{A}'$, a mapping assertion that produced it, identifying those variables (and only those variables) needed to produce the ABox assertion(s), and then taking the conjunction of the atoms in the bodies. We observe that by construction $D_\gamma = gr(\gamma)$ satisfies Condition 1 and is such that $\langle \mathcal{T}_1, \mathcal{M}_1, D_\gamma \rangle \models \bot$. By construction, there is a homomorphism of γ into the original database D_0. It follows that $\langle \mathcal{T}_1, \mathcal{M}_1, D_0 \rangle \models \bot$. □

Using the preceding result, we can pinpoint the complexity of $\bot$-entailment.

Theorem 3. *The $\bot$-entailment problem is NP-complete for OBDA specifications based upon DL-Lite$_R$ TBoxes and GAV / GLAV mappings, and NL-complete in the case of linear mappings.*

Proof. We know from Theorem 2 that $\Gamma_1 \models_\bot \Gamma_2$ iff $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models \bot$ for every database D satisfying Condition 1. For the GAV / GLAV case, we compute these databases in polynomial time and for every such database D, we guess a polynomial-size proof that $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models \bot$. For the linear case, we observe that the databases satisfying Condition 1 contain at most 2 tuples each and can be enumerated in logarithmic space. For every such database D, we can check using an NL oracle whether $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models \bot$. Since $\mathrm{L}^{\mathrm{NL}} = \mathrm{NL}$, we obtain an NL procedure. □

[1] Recall that distinct mapping assertions in a mapping have no common variables.

Next we consider entailment with respect to a specific query. We again start by showing it is sufficient to consider a finite number of databases of a particular form.

Theorem 4. *Let q be a CQ, and let $\Gamma_1 = \langle \mathcal{T}_1, \mathcal{M}_1 \rangle$ and $\Gamma_2 = \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ be OBDA specifications such that $\mathcal{T}_1, \mathcal{T}_2$ are formulated in DL-Lite$_R$, and $\mathcal{M}_1, \mathcal{M}_2$ are GLAV mappings. Then $\Gamma_1 \models_q \Gamma_2$ if and only if $\Gamma_1 \models_\perp \Gamma_2$ and $\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models q(\boldsymbol{a})$ implies $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models q(\boldsymbol{a})$ for every database D satisfying the following condition:*

- *Condition 2: D is obtained by (i) taking $k \leq |q|$ mapping assertions $m_1, m_2, \ldots, m_k$ from $\mathcal{M}_2$, (ii) identifying some of the frontier variables in $m_1, m_2, \ldots, m_k$, (iii) letting $D = gr(body(m_1) \cup body(m_2) \cup \ldots \cup body(m_k))$.*

If q is an IQ, then the latter condition can be replaced by:

- *Condition 3: D is obtained by (i) taking a mapping assertion m from $\mathcal{M}_2$ and choosing an atom $\alpha \in head(m)$, (ii) possibly identifying in m the (at most two) frontier variables appearing in α, and (iii) letting $D = gr(body(m))$.*

Proof. Again the one direction is immediate. To show the non-trivial direction, let us suppose that $\Gamma_1 \models_\perp \Gamma_2$ and that $\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models q(\boldsymbol{c})$ implies $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models q(\boldsymbol{c})$ for every tuple $\boldsymbol{c}$ and database D satisfying Condition 2 (we return later to the case of IQs). Let us further suppose that we have $\langle \mathcal{T}_2, \mathcal{M}_2, D_0 \rangle \models q(\boldsymbol{a})$. The first possibility is that $\langle \mathcal{T}_2, \mathcal{M}_2, D_0 \rangle \models \perp$, in which case we have $\langle \mathcal{T}_1, \mathcal{M}_1, D_0 \rangle \models \perp$ because of $\Gamma_1 \models_\perp \Gamma_2$. We thus obtain $\langle \mathcal{T}_1, \mathcal{M}_1, D_0 \rangle \models q_0(\boldsymbol{a})$. The other possibility is that $\langle \mathcal{T}_2, \mathcal{M}_2, D_0 \rangle \models q_0(\boldsymbol{a})$ and $\langle \mathcal{T}_2, \mathcal{M}_2, D_0 \rangle \not\models \perp$. If $\langle \mathcal{T}_1, \mathcal{M}_1, D_0 \rangle \models \perp$, we immediately obtain $\langle \mathcal{T}_1, \mathcal{M}_1, D_0 \rangle \models q_0(\boldsymbol{a})$. Otherwise, let $\mathcal{A}_{\mathcal{M}_2, D_0}$ be the ABox for $\mathcal{M}_2$ and D_0. Since $\langle \mathcal{T}_2, \mathcal{M}_2, D_0 \rangle \models q_0(\boldsymbol{a})$, we have $\langle \mathcal{T}_2, \mathcal{A}_{\mathcal{M}_2, D_0} \rangle \models q_0(\boldsymbol{a})$. It is a well-known property of *DL-Lite*$_R$ that there exists a subset $\mathcal{A}' \subseteq \mathcal{A}_{\mathcal{M}_2, D_0}$ with $|\mathcal{A}'| \leq |q_0|$ such that $\langle \mathcal{T}_2, \mathcal{A}' \rangle \models q_0(\boldsymbol{a})$. Let $|\mathcal{A}'| = k$, and let $\beta_1, \ldots, \beta_k$ be the ABox assertions in $\mathcal{A}'$. For each β_i, we choose a mapping assertion $m_i \in \mathcal{M}_2$ and a homomorphism h_i of $body(m_i)$ into D_0 such that $gr(h_i(head(m)))$ contains β_i. We also select an atom $\alpha_i \in head(m)$ such that $gr(h(\alpha)) = \beta$. Let m_i' be obtained from m_i by identifying frontier variables y and z if $h_i(y) = h_i(z)$, and set $D' = gr(body(m_1') \cup \ldots \cup body(m_k'))$. It is easy to see that D' satisfies Condition 2. Moreover, by construction, the ABox $\mathcal{A}_{\mathcal{M}_2, D'}$ contains a subset $\mathcal{A}''$ that is isomorphic to $\mathcal{A}'$, and so $\langle \mathcal{T}_2, \mathcal{M}_2, D' \rangle \models q_0(\boldsymbol{a}')$ where $\boldsymbol{a}'$ is tuple corresponding to $\boldsymbol{a}$ according to this isomorphism. Applying our assumption, we obtain $\langle \mathcal{T}_1, \mathcal{M}_1, D' \rangle \models q_0(\boldsymbol{a}')$. Using the fact that there is a homomorphism of $body(m_1') \cup \ldots \cup body(m_k')$ into D_0 that is an isomorphism on the frontier variables, we obtain $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models q_0(\boldsymbol{a})$.

Finally, for the case of instance queries, we simply note that we have $k = 1$, and it is only necessary to identify those variables in the head atom of the mapping that leads to introducing the single ABox assertion of interest. This yields Condition 3. □

We pinpoint the complexity of single CQ-entailment, showing it to be Π_2^p-complete.

Theorem 5. *The single CQ-entailment problem is Π_2^p-complete for OBDA specifications based upon DL-Lite$_R$ TBoxes and GLAV mappings. The lower bound holds even for linear mapping assertions and when both TBoxes are empty.*

Proof. For the upper bound, consider two OBDA specifications $\Gamma_1 = \langle \mathcal{T}_1, \mathcal{M}_1 \rangle$ and $\Gamma_2 = \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$. From Theorems 2 and 4, we know that $\Gamma_1 \not\models_q \Gamma_2$ if and only if one of the following holds:

- there is a database D satisfying Condition 2 such that $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \not\models \bot$;
- there is a database D satisfying Condition 2 such that $\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models q(\boldsymbol{a})$, $\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \not\models \bot$, and $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \not\models q(\boldsymbol{a})$.

The first item can be checked using an NP oracle (by Theorem 3). To check the second item, we remark that the size of databases satisfying Condition 2 cannot exceed $max(2, |q|) \cdot \mathsf{maxbody}$, where $\mathsf{maxbody}$ is the maximum number of atoms appearing in the body of a mapping assertion in $\mathcal{M}_2$. It follows that to show that the second item above is violated, we can guess a database D of size at most $max(2, |q|) \cdot \mathsf{maxbody}$ together with a tuple of constants $\boldsymbol{a}$ and a polynomial-size proof that $\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models q(\boldsymbol{a})$, and then we can verify using an NP oracle that $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \not\models q(\boldsymbol{a})$. We therefore obtain a Σ_2^p procedure for deciding the complement of our problem.

For the lower bound, we utilize a result from [3] on query containment over signature-restricted ABoxes. In that paper, it is shown how, given a 2QBF $\forall \boldsymbol{u} \exists \boldsymbol{v} \varphi(\boldsymbol{u}, \boldsymbol{v})$, one can construct a TBox $\mathcal{T}$, Boolean CQs q_1 and q_2, and a signature Σ such that $\forall \boldsymbol{u} \exists \boldsymbol{v} \varphi(\boldsymbol{u}, \boldsymbol{v})$ is valid iff $\mathcal{T}, \mathcal{A} \models q_1 \Rightarrow \mathcal{T}, \mathcal{A} \models q_2$ for all ABoxes $\mathcal{A}$ with $\mathsf{sig}(\mathcal{A}) \subseteq \Sigma$. We will not detail the construction but simply remark that the same TBox $\mathcal{T} = \{T \sqsubseteq V, F \sqsubseteq V\}$ is used for all QBFs, the signature Σ is given by $(\mathsf{sig}(\mathcal{T}) \cup \mathsf{sig}(q_1) \cup \mathsf{sig}(q_2)) \setminus \{V\}$, and the query q_2 is such that $V \notin \mathsf{sig}(q_2)$.

In what follows, we will show how given $\mathcal{T}$, q_1, q_2, and Σ as above, we can reduce the problem of testing whether $\mathcal{T}, \mathcal{A} \models q_1$ implies $\mathcal{T}, \mathcal{A} \models q_2$ for all Σ-ABoxes to the problem of single CQ entailment. We will use Σ for our database instances, and we create two copies $\Sigma_1 = \{P^1 \mid P \in \Sigma\}$ and $\Sigma_2 = \{P^2 \mid P \in \Sigma\}$ of the signature Σ to be used in the head of mapping assertions. Next, we define sets of mapping assertions $\mathsf{copy}^1(\Sigma)$ and $\mathsf{copy}^2(\Sigma)$ that simply copies all of the predicates in Σ into the corresponding symbol in Σ_1 (resp. Σ_2). Formally, for $j \in \{1, 2\}$,

$$\mathsf{copy}^j(\Sigma) = \{A(x) \rightarrow A^j(x) \mid A \in \Sigma \cap \mathsf{N_C}\} \cup \{R(x, y) \rightarrow R^j(x, y) \mid R \in \Sigma \cap \mathsf{N_R}\}$$

We further define, given a data signature Λ_1 and DL signature Λ_2, a set $\mathsf{populate}(\Lambda_1, \Lambda_2)$ of mapping assertions that populates the relations in Λ_2 using all possible combinations of the constants appearing in tuples over Λ_1:

$$\begin{aligned}\mathsf{populate}(\Lambda_1, \Lambda_2) = \{&P(x_1, \ldots, x_k) \rightarrow P'(x'_1, \ldots, x'_\ell) \mid P \in \Lambda_1, \mathsf{arity}(P) = k, \\ &P' \in \Lambda_2, \mathsf{arity}(P') = \ell, \{x'_1, \ldots, x'_\ell\} \subseteq \{x_1, \ldots, x_k)\}\end{aligned}$$

Using $\mathsf{copy}^1(\Sigma)$, $\mathsf{copy}^2(\Sigma)$, $\mathsf{populate}(\Sigma, \Sigma^1)$, and $\mathsf{populate}(\Sigma, \Sigma^2)$, we construct the following mappings:

$$\begin{aligned}\mathcal{M}_1 &= \mathsf{populate}(\Sigma, \Sigma^1) \cup \mathsf{copy}^2(\Sigma) \\ \mathcal{M}_2 &= \mathsf{copy}^1(\Sigma) \cup \mathsf{populate}(\Sigma, \Sigma^2) \cup \{T(x) \rightarrow V(x), F(x) \rightarrow V(x)\}\end{aligned}$$

Observe that both mappings are linear. For the query, we let q'_1 (resp. q'_2) be obtained from q_1 (resp. q_2) by replacing every predicate P by P^1(resp. P^2). We also rename

variables so that q_1' and q_2' do not share any variables. We then let q be the CQ obtained by taking the conjunction of q_1' and q_2' and existentially quantifying all variables. In the appendix, we show that $\langle \emptyset, \mathcal{M}_1 \rangle \models_q \langle \emptyset, \mathcal{M}_2 \rangle$ iff $\mathcal{T}, \mathcal{A} \models q_1 \Rightarrow \mathcal{T}, \mathcal{A} \models q_2$ for all Σ-ABoxes. By combining this with the reduction from [3], we obtain a reduction from universal 2QBF to the q-entailment problem, establishing Π_2^p-hardness of the latter. □

If we consider IQs instead, the complexity drops to either NP- or NL-complete.

Theorem 6. *The single IQ-entailment problem is NP-complete for OBDA specifications based upon DL-Lite$_R$ TBoxes and either GAV or GLAV mappings. It is NL-complete if linear mappings are considered.*

Proof. We give the arguments for GAV and GLAV mappings (for linear case, see the appendix). For the NP upper bound, consider two OBDA specifications $\Gamma_1 = \langle \mathcal{T}_1, \mathcal{M}_1 \rangle$ and $\Gamma_2 = \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$, and let q be an IQ. By Theorem 4, $\Gamma_1 \models_q \Gamma_2$ if and only if $\Gamma_1 \models_\bot \Gamma_2$ and $\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models \alpha$ implies $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models \alpha$ for all databases D satisfying Condition 3 and for all Boolean IQs α obtained by instantiating the variable(s) in q with constant(s) from D.

We already know that it is in NP to test whether $\Gamma_1 \models_\bot \Gamma_2$. For the second property, observe that there are only polynomially many databases satisfying Condition 2, since each corresponds to choosing a mapping assertion m in $\mathcal{M}_2$, an atom $\alpha \in head(m)$, and deciding whether or not to identify variables in α. For every such database D, we compute (in polynomial time) the set of Boolean IQs β obtained by instantiating the IQ q with constants from D for which $\langle \mathcal{T}_2, \mathcal{M}_2, gr(head(m)) \rangle \models \beta$. For every such β, we guess a polynomial-size proof that $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models \beta$. If all of our polynomially many guesses succeed, then the procedure returns yes, and otherwise no. By grouping all of the guesses together, we obtain an NP decision procedure.

The NP lower bound is by reduction from the NP-complete CQ containment problem: given two CQs q_1, q_2 both having a single answer variable x, we have $q_1 \subseteq q_2$ iff $\langle \emptyset, \{q_2 \rightarrow A(x)\} \rangle \models_{A(x)} \langle \emptyset, \{q_1 \rightarrow A(x)\} \rangle$, where A is a concept name that does not appear in either of q_1 and q_2. □

Finally, we consider entailment with respect to entire classes of queries. Again, we can show it is sufficient to consider a small number of databases of a particular form.

Theorem 7. *Let $\Gamma_1 = \langle \mathcal{T}_1, \mathcal{M}_1 \rangle$ and $\Gamma_2 = \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ be as in Theorem 4. For $\mathcal{L} \in \{\mathsf{CQ}, \mathsf{IQ}\}$, $\Gamma_1 \models_\mathcal{L} \Gamma_2$ if and only if $\Gamma_1 \models_\bot \Gamma_2$ and $\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models q(\boldsymbol{a})$ implies $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models q(\boldsymbol{a})$ for every $q \in \mathcal{L}$ and every database D that satisfies Condition 3.*

We show that testing CQ-entailment is much more difficult than for single CQs. Both the upper and lower bounds use recent results on KB query inseparability [4].

Theorem 8. *CQ-entailment is EXPTIME-complete for OBDA specifications based upon DL-Lite$_R$ TBoxes and GAV / linear mappings; it is in 2EXPTIME for GLAV.*

Proof. We start with the proof of the membership results. Consider OBDA specifications $\Gamma_1 = \langle \mathcal{T}_1, \mathcal{M}_1 \rangle$ and $\Gamma_2 = \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$. By Theorem 7, $\Gamma_1 \models_\mathsf{CQ} \Gamma_2$ if and only if $\Gamma_1 \models_\bot \Gamma_2$ and $\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models q(\boldsymbol{a})$ implies $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models q(\boldsymbol{a})$ for every choice

of $q(\boldsymbol{a})$ and every database D satisfying Condition 3. We know that testing $\Gamma_1 \models_\perp \Gamma_2$ can be done in NP (Theorem 3). To decide whether the second property holds, we consider each of the (polynomially many) databases satisfying Condition 3. For every such database D, we generate the two ABoxes $\mathcal{A}_{\mathcal{M}_1,D}$ and $\mathcal{A}_{\mathcal{M}_2,D}$ and the corresponding KBs $\mathcal{K}_1 = \langle \mathcal{T}_1, \mathcal{A}_{\mathcal{M}_1,D} \rangle$ and $\mathcal{K}_2 = \langle \mathcal{T}_2, \mathcal{A}_{\mathcal{M}_2,D} \rangle$. In the case of GAV mappings, these KBs are guaranteed to be of polynomial size, whereas for GLAV mappings, they may be (single) exponentially large due to presence of existential variables in the heads of mapping assertions. We then test whether it is the case that for every CQ q over $\mathsf{sig}(\mathcal{K}_2)$, $\mathcal{K}_2 \models q(\boldsymbol{a})$ implies $\mathcal{K}_1 \models q(\boldsymbol{a})$, and we return no if this is not the case. The preceding check corresponds to the Σ-query entailment problem for *DL-Lite*$_R$ KBs, which has been recently studied in [4] and shown to be EXPTIME-complete. We therefore obtain an EXPTIME (resp. 2EXPTIME) procedure for deciding CQ-entailment between OBDA specifications involving GAV (resp. GLAV) mappings.

Our lower bound also makes use of the recent work on query inseparability of *DL-Lite*$_R$ knowledge bases. In [4], the following problem is shown to be EXPTIME-complete: given *DL-Lite*$_R$ TBoxes $\mathcal{T}_1$ and $\mathcal{T}_2$ that are consistent with the ABox $\{A(c)\}$, decide whether the certain answers for q w.r.t. $\langle \mathcal{T}_2, \{A(c)\} \rangle$ are contained in those for $\langle \mathcal{T}_1, \{A(c)\} \rangle$ for every CQ q with $\mathsf{sig}(q) \subseteq \mathsf{sig}(\mathcal{T}_2)$. To reduce this problem to the CQ-entailment problem for OBDA specifications, we consider the following linear mapping that populates the concept A with all constants appearing in the unary database relation A' (refer to the proof of Theorem 5 for the definition of $\mathsf{populate}$): $\mathcal{M}_1 = \mathcal{M}_2 = \mathsf{populate}(\{A'\}, \{A\})$. To complete the proof, we show in the appendix that $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\mathsf{CQ}} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ iff $\langle \mathcal{T}_2, \{A(c)\} \rangle \models q(\boldsymbol{a})$ implies $\langle \mathcal{T}_1, \{A(c)\} \rangle \models q(\boldsymbol{a})$ for every CQ q $\mathsf{sig}(q) \subseteq \mathsf{sig}(\mathcal{T}_2)$. □

Our final result shows that IQ-entailment has the same complexity as single IQ-entailment. The proof proceeds similarly to the proof of Theorem 6.

Theorem 9. *IQ-entailment is NP-complete for OBDA specifications based upon DL-Lite*$_R$ *TBoxes and either GAV or GLAV mappings. It is NL-complete if linear mappings are considered.*

5 Conclusion and Future Work

In this paper, we have introduced notions of query-based entailment of OBDA specifications and have analyzed the complexity of checking query-based entailment for different classes of queries and mappings and for TBoxes formulated in *DL-Lite*$_R$.

The present work constitutes only a first step towards a full analysis of query-based forms of comparing OBDA specifications, and can be extended in several directions:

- First, it would be interesting to extend the computational analysis of query entailment to other DLs beyond *DL-Lite*$_R$. For instance, one interesting question for DLs with functional or cardinality restrictions concerns the impact of the Unique Name Assumption on the complexity of (and the techniques for) query entailment.
- Second, other forms of mapping beyond GAV and GLAV could be analyzed. In particular, we would like to see whether decidability of query entailment is preserved if we add some restricted form of inequality or negation to the mapping bodies.

– Third, we could introduce a query signature and only test entailment for queries formulated in the given signature, as has been done for TBox and KB query inseparability [4]. In fact, all of the complexity upper bounds in this paper hold also if we introduce a query signature, but this may not be the case for other DLs.
– Finally, to explore the impact of restricting the set of possible databases, we could extend the computational analysis to database schemas with integrity constraints.

Acknowledgments. This research has been partially supported by the EU under FP7 project Optique (grant n. FP7-318338) and by the French National Research Agency under ANR project PAGODA (grant n. ANR-12-JS02-007-01).

References

1. N. Antonioli, F. Castanò, C. Civili, S. Coletta, S. Grossi, D. Lembo, M. Lenzerini, A. Poggi, D. F. Savo, and E. Virardi. Ontology-based data access: the experience at the Italian Department of Treasury. In *Proc. of the Industrial Track of the 25th Int. Conf. on Advanced Information Systems Engineering (CAiSE)*, 2013.
2. A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The *DL-Lite* family and relations. *J. of Artificial Intelligence Research*, 36:1–69, 2009.
3. M. Bienvenu, C. Lutz, and F. Wolter. Query containment in description logics reconsidered. In *Proc. of the 13th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR)*, 2012.
4. E. Botoeva, R. Kontchakov, V. Ryzhikov, F. Wolter, and M. Zakharyaschev. Query inseparability for description logic knowledge bases. In *Proc. of the 14th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR)*, 2014.
5. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodriguez-Muro, R. Rosati, M. Ruzzi, and D. F. Savo. The Mastro system for ontology-based data access. *Semantic Web J.*, 2(1):43–53, 2011.
6. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. of Automated Reasoning*, 39(3):385–429, 2007.
7. D. Calvanese, M. Giese, P. Haase, I. Horrocks, T. Hubauer, Y. Ioannidis, E. Jiménez-Ruiz, E. Kharlamov, H. Kllapi, J. Klüwer, M. Koubarakis, S. Lamparter, R. Möller, C. Neuenstadt, T. Nordtveit, Ö. Özcep, M. Rodriguez-Muro, M. Roshchin, F. Savo, M. Schmidt, A. Soylu, A. Waaler, and D. Zheleznyakov. Optique: OBDA solution for big data. In *Revised Selected Papers of ESWC 2013 Satellite Events*, volume 7955 of *Lecture Notes in Computer Science*, pages 293–295, 2013.
8. A. Doan, A. Y. Halevy, and Z. G. Ives. *Principles of Data Integration*. Morgan Kaufmann, 2012.
9. G. Gottlob, R. Pichler, and V. Savenkov. Normalization and optimization of schema mappings. *Very Large Database J.*, 20(2):277–302, 2011.
10. E. Kharlamov, M. Giese, E. Jiménez-Ruiz, M. G. Skjæveland, A. Soylu, D. Zheleznyakov, T. Bagosi, M. Console, P. Haase, I. Horrocks, S. Marciuska, C. Pinkel, M. Rodriguez-Muro, M. Ruzzi, V. Santarelli, D. F. Savo, K. Sengupta, M. Schmidt, E. Thorstensen, J. Trame, and A. Waaler. Optique 1.0: Semantic access to big data: The case of Norwegian Petroleum Directorate's FactPages. In *Proc. of the ISWC Posters & Demos Track*, pages 65–68, 2013.
11. B. Konev, R. Kontchakov, M. Ludwig, T. Schneider, F. Wolter, and M. Zakharyaschev. Conjunctive query inseparability of OWL 2 QL TBoxes. In *Proc. of the 25th AAAI Conf. on Artificial Intelligence (AAAI)*, 2011.

12. D. Lembo, J. Mora, R. Rosati, D. F. Savo, and E. Thorstensen. Towards mapping analysis in ontology-based data access. In *Proc. of the 8th Int. Conf. on Web Reasoning and Rule Systems (RR)*, pages 108–123, 2014.
13. A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati. Linking data to ontologies. *J. on Data Semantics*, X:133–173, 2008.
14. M. Rodriguez-Muro, R. Kontchakov, and M. Zakharyaschev. Ontology-based data access: Ontop of databases. In *Proc. of the 12th Int. Semantic Web Conf. (ISWC)*, 2013.

A Additional Proof Details

We recall the notion of canonical model which we will use in some of our proofs. Let $\mathcal{I}$ be a model of a *DL-Lite*$_R$ KB $\langle \mathcal{T}, \mathcal{A} \rangle$. We call $\mathcal{I}$ a *canonical model* of $\langle \mathcal{T}, \mathcal{A} \rangle$ if, for every model $\mathcal{J}$ of $\langle \mathcal{T}, \mathcal{A} \rangle$, there exists a homomorphism $h : \Delta^{\mathcal{I}} \rightarrow \Delta^{\mathcal{J}}$, such that: (i) for every constant name a, $h(a^{\mathcal{I}}) = a^{\mathcal{J}}$; (ii) for every $d \in \Delta^{\mathcal{I}}$ and for every concept name A, if $d \in A^{\mathcal{I}}$ then $h(d) \in A^{\mathcal{J}}$; (iii) for every pair of elements $d, d' \in \Delta^{\mathcal{I}}$ and for every role name R, if $\langle d, d' \rangle \in R^{\mathcal{I}}$ then $\langle h(d), h(d') \rangle \in R^{\mathcal{J}}$.

It is well known that consistent *DL-Lite*$_R$ KBs possess canonical models; we refer the reader to standard references for concrete constructions of canonical models in *DL-Lite*$_R$. Importantly, canonical models characterize conjunctive query answering over *DL-Lite*$_R$ KBs. More precisely, it can be shown that for every CQ q and every tuple $\boldsymbol{a}$ of constants occurring in $\mathcal{A}$, $\langle \mathcal{T}, \mathcal{A} \rangle \models q(\boldsymbol{a})$ iff $\mathcal{I} \models q(\boldsymbol{a})$ with $\mathcal{I}$ a canonical model for $\langle \mathcal{T}, \mathcal{A} \rangle$. This property immediately carries over to the OBDA setting as follows: given an OBDA specification $\Gamma = \langle \mathcal{T}, \mathcal{M} \rangle$, a database instance D, a canonical model $\mathcal{I}$ for $\langle \mathcal{T}, \mathcal{A}_{\mathcal{M},D} \rangle$, a conjunctive query q and a tuple $\boldsymbol{a}$ of constants occurring in $\mathcal{A}_{\mathcal{M},D}$, $(\Gamma, D) \models q(\boldsymbol{a})$ iff $\mathcal{I} \models q(\boldsymbol{a})$.

The next lemma establishes the claim from the proof of Theorem 5.

Lemma 1. *$\langle \emptyset, \mathcal{M}_1 \rangle \models_q \langle \emptyset, \mathcal{M}_2 \rangle$ iff $\mathcal{T}, \mathcal{A} \models q_1 \Rightarrow \mathcal{T}, \mathcal{A} \models q_2$ for all Σ-ABoxes.*

Proof. First suppose that $\langle \emptyset, \mathcal{M}_1 \rangle \models_q \langle \emptyset, \mathcal{M}_2 \rangle$, and let $\mathcal{A}$ be an ABox such that $\mathsf{sig}(\mathcal{A}) \subseteq \Sigma$ and $\mathcal{T}, \mathcal{A} \models q_1$. Let D be the database instance consisting of the facts in $\mathcal{A}$. Since $\mathcal{M}_2$ contains $\mathsf{copy}^1(\Sigma)$ as well as $\{T(x) \rightarrow V(x), F(x) \rightarrow V(x)\}$, it follows that $\langle \emptyset, \mathcal{M}_2, D \rangle \models q_1'$. Moreover, because $\mathcal{M}_2$ contains $\mathsf{populate}(\Sigma, \Sigma^2)$, the corresponding ABox $\mathcal{A}_{\mathcal{M}_2,D}$ contains all Σ^2-facts that can be built using constants from D (hence $\mathcal{A}$). Using the fact that q_2' only uses predicates from Σ^2, we can infer that $\langle \emptyset, \mathcal{M}_2, D \rangle \models q_2'$ and hence that $\langle \emptyset, \mathcal{M}_2, D \rangle \models q$. By our assumption that $\langle \emptyset, \mathcal{M}_1 \rangle \models_q \langle \emptyset, \mathcal{M}_2 \rangle$, we obtain $\langle \emptyset, \mathcal{M}_1, D \rangle \models q$, which implies that $\langle \emptyset, \mathcal{M}_1, D \rangle \models q_2'$. We then observe that $\mathcal{A}_{\mathcal{M}_1,D}$ contains $P^2(\boldsymbol{c})$ iff $\mathcal{A}$ contains the corresponding assertion $P(\boldsymbol{c})$. It follows that the homomorphism witnessing that $\langle \emptyset, \mathcal{M}_1, D \rangle \models q_2'$ can be reproduced in $\mathcal{A}$ using the original predicates, so $\mathcal{A} \models q_2$.

For the other direction, suppose that for all Σ-ABoxes, $\mathcal{T}, \mathcal{A} \models q_1$ implies $\mathcal{T}, \mathcal{A} \models q_2$. Let D be a database instance such that $\langle \emptyset, \mathcal{M}_2, D \rangle \models q$. It follows that $\langle \emptyset, \mathcal{M}_2, D \rangle \models q_1'$. Note that since the left-hand side of mapping assertions in $\mathcal{M}_2$ only use predicates from Σ, we may assume w.l.o.g. that $\mathsf{sig}(D) \subseteq \Sigma$. Let $\mathcal{A}_D$ be the Σ-ABox containing the facts in D. From $\langle \emptyset, \mathcal{M}_2, D \rangle \models q_1'$ and the fact that the

two inclusions in $\mathcal{T}$ simulate the effect of the mapping assertions $T(x) \rightarrow V(x)$ and $F(x) \rightarrow V(x)$, we can infer that $\mathcal{T}, \mathcal{A}_D \models q_1$. Applying our assumption, we obtain $\mathcal{T}, \mathcal{A}_D \models q_2$, which yields $\langle \emptyset, \mathcal{M}_1, D \rangle \models q_2'$ because of the mapping assertions in $\mathsf{copy}^2(\Sigma)$. We can also show that $\langle \emptyset, \mathcal{M}_1, D \rangle \models q_1'$ using the mapping assertions in $\mathsf{populate}(\Sigma, \Sigma^1)$, from which we obtain $\langle \emptyset, \mathcal{M}_1, D \rangle \models q$, as desired. □

We give the missing proof of the linear case for Theorem 6.

Proof of Theorem 6 (continued). In the case of linear mappings, we can again use Theorem 4 to restrict the number and form of the databases that need to be considered. By Theorem 2, we can check in NL whether $\Gamma_1 \models_\perp \Gamma_2$. We can next enumerate, in logarithmic space, all of the databases satisfying Condition 3. We can then iterate over all the relevant IQs based upon q and the constants in D and use a call to an NL oracle to check whether the IQ is entailed from $\langle \Gamma_1, D \rangle$. The procedure returns yes if all of the checks succeed. Since $\mathrm{L}^{\mathrm{NL}} = \mathrm{NL}$, we obtain an NL decision procedure.

For the NL lower bound, we can reduce concept subsumption to single IQ entailment with linear mappings as follows: $\mathcal{T} \models A \sqsubseteq B$ iff $\langle \mathcal{T}, \{A(x) \rightarrow A(x)\} \rangle \models_{B(x)} \langle \mathcal{T}, \{A(x) \rightarrow B(x)\} \rangle$. □

We now give a detailed proof sketch for Theorem 7.

Proof of Theorem 7. We start by considering the case in which $\mathcal{L} = \mathsf{CQ}$. Suppose that $\Gamma_1 \models_\perp \Gamma_2$ and $\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models q(\boldsymbol{a})$ implies $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models q(\boldsymbol{a})$ for every $q \in \mathsf{CQ}$ and every database D that satisfies Condition 3. Further suppose that $\langle \mathcal{T}_2, \mathcal{M}_2, D_0 \rangle \models q_0(\boldsymbol{a})$, where D_0 may be any database. The first possibility is that $\langle \mathcal{T}_2, \mathcal{M}_2, D_0 \rangle \models \perp$, in which case $\Gamma_1 \models_\perp \Gamma_2$ yields $\langle \mathcal{T}_1, \mathcal{M}_1, D_0 \rangle \models \perp$ and thus, $\langle \mathcal{T}_1, \mathcal{M}_1, D_0 \rangle \models q_0(\boldsymbol{a})$. The other possibility is that $\langle \mathcal{T}_2, \mathcal{M}_2, D_0 \rangle \models q_0(\boldsymbol{a})$ and $\langle \mathcal{T}_2, \mathcal{M}_2, D_0 \rangle \not\models \perp$. If $\langle \mathcal{T}_1, \mathcal{M}_1, D_0 \rangle \models \perp$, we are done. Otherwise, let $\mathcal{A}_{\mathcal{M}_1, D_0}$ and $\mathcal{A}_{\mathcal{M}_2, D_0}$ be the corresponding ABoxes, and let $\mathcal{I}_1$ and $\mathcal{I}_2$ be the canonical models of $\langle \mathcal{T}_1, \mathcal{A}_{\mathcal{M}_1, D_0} \rangle$ and $\langle \mathcal{T}_2, \mathcal{A}_{\mathcal{M}_2, D_0} \rangle$ respectively. Since $\langle \mathcal{T}_2, \mathcal{M}_2, D_0 \rangle \models q_0(\boldsymbol{a})$, we have $\mathcal{I}_2 \models q_0(\boldsymbol{a})$, and thus there must exist a match π for $q_0(\boldsymbol{a})$ in $\mathcal{I}_2$ (i.e., a homomorphism of q_0 into $\mathcal{I}_2$ that sends the answer variables $\boldsymbol{x}$ of q_0 to $\boldsymbol{a}$). This match gives rise to an equivalence relation $\sim_\pi$ on the atoms in q, defined as follows: $\alpha \sim_\pi \alpha'$ just in the case that α and α' share a variable v such that $\pi(v)$ is not a constant. This equivalence relation in turn induces a partition of q_0 into connected subqueries $q_1, \ldots, q_n$, where each of the queries q_i corresponds to one of the equivalence classes of $\sim_\pi$. Each of the queries q_i satisfies exactly one of the following conditions:

- q_i consists of a single atom all of whose variables are mapped by π to constants in $\mathcal{A}_{\mathcal{M}_2, D}$;
- every atom in q_i contains a variable that is not mapped to a constant of $\mathcal{A}_{\mathcal{M}_2, D}$, and there is a unique constant a_i in $\mathcal{A}_{\mathcal{M}_2, D}$ to which one or more variables of q_i is mapped;
- q_i contains no constants and none of its variables are mapped to constants in $\mathcal{A}_{\mathcal{M}_2, D}$.

Using our assumptions, we can show that the matches for each of these subqueries can be reproduced in $\mathcal{I}_1$ in such a way that we obtain a match for $q_0(\boldsymbol{a})$ in $\mathcal{I}_1$, yielding

$\langle \mathcal{T}_1, \mathcal{M}_1, D_0 \rangle \models q_0(\boldsymbol{a})$. The proof relies on standard arguments and known properties of canonical models. For each query q_i ($1 \leq i \leq n$), we proceed as follows:

1. We create a new query q_i' which is obtained by identifying those variables in q_i that are mapped to the same constant (and leaving all other variables untouched). If q_i satisfied the first condition, then all of its variables are designated as answer variables. If the second condition was satisfied, then only the unique remaining variable that maps to a constant is considered as an answer variable. If the third condition holds, then there are no answer variables. By construction, we have $\mathcal{I}_2 \models q_i'(\boldsymbol{b})$, with $\boldsymbol{b}$ being the tuple of constants obtained by applying π to the answer variables of q_i'.
2. From $\mathcal{I}_2 \models q_i'(\boldsymbol{b})$, we have $\langle \mathcal{T}_2, \mathcal{A}_{\mathcal{M}_2,D} \rangle \models q_i'(\boldsymbol{b})$. Using standard arguments based upon canonical models, we can show that in fact there is a single ABox assertion $\beta \in \mathcal{A}_{\mathcal{M}_2,D}$ such that $\langle \mathcal{T}_2, \{\beta\} \rangle \models q_i'(\boldsymbol{b})$. Note in particular that β must contain all constants in the tuple $\boldsymbol{b}$.
3. We now choose some mapping assertion $m \in \mathcal{M}_2$ and a homomorphism h of $body(m)$ into D_0 such that $gr(h(head(m)))$ contains β. We select an atom $\alpha \in head(m)$ such that the grounding of $h(\alpha)$ is β. We then let m' be obtained from m by identifying variables y and z if both belong to the atom α and $h(y) = h(z)$ (thus, we identify at most one pair of variables).
4. We let $D_m' = gr(m')$, let h' be the homomorphism that maps each variable in $body(m')$ to the corresponding constant, let β' be the grounding of $h'(\alpha)$, and let $\boldsymbol{b}'$ be the tuple obtained by substituting in $\boldsymbol{b}$ the corresponding constants from β'. By construction, D_m' satisfies Condition 3. Moreover, from $\langle \mathcal{T}_2, \{\beta\} \rangle \models q_i'(\boldsymbol{b})$, we can infer $\langle \mathcal{T}_2, \{\beta'\} \rangle \models q_i'(\boldsymbol{b}')$, hence $\langle \mathcal{T}_2, \mathcal{M}_2, D_m' \rangle \models q_i'(\boldsymbol{b}')$.
5. Since D_m' satisfies Condition 3, we can apply our assumption to obtain $\langle \mathcal{T}_1, \mathcal{M}_1, D_m' \rangle \models q_i'(\boldsymbol{b}')$, from which we can infer $\langle \mathcal{T}_1, \mathcal{M}_1, D_0 \rangle \models q_i(\boldsymbol{b})$.

It follows from the construction of the q_i and Point 5 that $\langle \mathcal{T}_1, \mathcal{M}_1, D_0 \rangle \models q_0(\boldsymbol{a})$.

So far, we have considered the case in which $\mathcal{L} = \mathsf{CQ}$, but the same reasoning can be applied to show the result for IQ-entailment. Indeed, the only difference is that there are fewer cases to consider, as the unique query atom must have all of its variables mapped to ABox constants. □

We next prove the claim from the proof of Theorem 8.

Lemma 2. *$\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\mathsf{CQ}} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ iff $\langle \mathcal{T}_2, \{A(c)\} \rangle \models q(\boldsymbol{a})$ implies $\langle \mathcal{T}_1, \{A(c)\} \rangle \models q(\boldsymbol{a})$ for every CQ q with $\mathsf{sig}(q) \subseteq \mathsf{sig}(\mathcal{T}_2)$.*

Proof. First suppose that $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\mathsf{CQ}} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$ and $\langle \mathcal{T}_2, \{A(c)\} \rangle \models q(\boldsymbol{a})$. Let D be the data instance consisting of the single fact $A'(c)$. Then $\mathcal{A}_{D,\mathcal{M}_2} = \{A(c)\}$, so $\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models q(\boldsymbol{a})$. Using our assumption that $\langle \mathcal{T}_1, \mathcal{M}_1 \rangle \models_{\mathsf{CQ}} \langle \mathcal{T}_2, \mathcal{M}_2 \rangle$, we obtain $\langle \mathcal{T}_1, \mathcal{M}_1, D \rangle \models q(\boldsymbol{a})$, and hence, $\langle \mathcal{T}_1, \{A(c)\} \rangle \models q(\boldsymbol{a})$.

For the other direction, suppose that $\langle \mathcal{T}_2, \{A(c)\} \rangle \models q(\boldsymbol{a})$ implies $\langle \mathcal{T}_1, \{A(c)\} \rangle \models q(\boldsymbol{a})$ for every CQ q with $\mathsf{sig}(q) \subseteq \mathsf{sig}(\mathcal{T}_2)$, and let D be a data instance and q_0 a CQ with $\mathsf{sig}(q_0) \subseteq \mathsf{sig}(\mathcal{T}_2)$ such that $\langle \mathcal{T}_2, \mathcal{M}_2, D \rangle \models q_0(\boldsymbol{a})$. Let $\mathcal{I}_2$ be a canonical model for $\langle \mathcal{T}_2, \mathcal{A}_{D,\mathcal{M}_2} \rangle$. Because of the form of the mapping $\mathcal{M}_2$, we know that $\mathcal{I}_2$ is the disjoint union of the canonical models of the KBs $\langle \mathcal{T}_2, A(d) \} \rangle$ over all constants d such

that $A'(d) \in D$. If q_0 is connected, then there must exist a single constant d from D such that $\langle \mathcal{T}_2, \{A(d)\}\rangle \models q_0(\boldsymbol{a})$ (in which case $\boldsymbol{a} = \langle d, \ldots, d\rangle$). We can thus apply our assumption to obtain $\langle \mathcal{T}_1, \{A(d)\}\rangle \models q_0(\boldsymbol{a})$, hence $\langle \mathcal{T}_1, \mathcal{M}_1, D\rangle \models q_0(\boldsymbol{a})$. If q_0 has several connected subqueries, then we can apply the same argument for each connected subquery, again concluding that $\langle \mathcal{T}_1, \mathcal{M}_1, D\rangle \models q_0(\boldsymbol{a})$. □

We now give the proof of Theorem 9.

Proof of Theorem 9. Consider OBDA specifications $\Gamma_1 = \langle \mathcal{T}_1, \mathcal{M}_1\rangle$ and $\Gamma_2 = \langle \mathcal{T}_2, \mathcal{M}_2\rangle$. By Theorem 7, $\Gamma_1 \models_{\mathsf{IQ}} \Gamma_2$ if and only if $\Gamma_1 \models_{\perp} \Gamma_2$ and $\langle \mathcal{T}_2, \mathcal{M}_2, D\rangle \models q(\boldsymbol{a})$ implies $\langle \mathcal{T}_1, \mathcal{M}_1, D\rangle \models q(\boldsymbol{a})$ for every IQ q and every database D satisfying Condition 3. We can therefore use almost exactly the same arguments as in the proof of Theorem 6 to obtain an NP upper bound for the GAV / GLAV case and NL for linear mappings. The only difference in the argument is that we need to consider all polynomially many (and logspace-enumerable) IQs that can be built using the constants in D and the concept and role names from Γ_2.

For the NP lower bound, we can use the same reduction from CQ containment as in Theorem 6. The NL lower bound can be shown by a slight modification of the reduction from Theorem 6: $\mathcal{T} \models A \sqsubseteq B$ iff $\langle \mathcal{T}, \{T(x) \to A(x)\}\rangle \models_{\mathsf{IQ}} \langle \mathcal{T}, \{T(x) \to A(x), T(x) \to B(x)\}\rangle$. □