Mapping Analysis in Ontology-based Data Access: Algorithms and Complexity (Discussion Paper)

Domenico Lembo[1], José Mora[1], Riccardo Rosati[1],
Domenico Fabio Savo[1], Evgenij Thorstensen[2]

[1] Sapienza Università di Roma
lastname@dis.uniroma1.it

[2] University of Oslo
evgenit@ifi.uio.no

Abstract. We study the formal analysis of mappings in ontology-based data access (OBDA). Specifically, we focus on the problem of identifying mapping inconsistency and redundancy, two of the most important anomalies for mappings in OBDA. We consider a wide range of ontology languages that comprises OWL 2 and all its profiles, and examine mapping languages of different expressiveness over relational databases. We establish tight complexity bounds for the decision problems associated with mapping inconsistency and redundancy.

1 Introduction

Ontology-based data access (OBDA) is a data integration paradigm that relies on a three-level architecture, constituted by the ontology, the data sources, and the mapping between the two [11, 12]. The ontology is the specification of a conceptual view of the domain of interest, and it is the system interface towards the user, whereas the mapping relates the elements of the ontology with the data at the sources.

One important aspect in OBDA concerns the construction of a system specification, i.e., defining the ontology and the mappings over an existing set of data sources. Mappings are indeed the most complex part of an OBDA specification, since they have to capture the semantics of the data sources and express such semantics in terms of the ontology. The first OBDA experiences in real-world scenarios (e.g., [2, 7]) have shown that the semantic distance between the conceptual and the data layer is often very large, because data sources are mostly application-oriented: this often makes the definition, debugging, and maintenance of mappings a hard and complex task.

In a recent paper [9], we started providing a theoretical basis for mapping management support in OBDA, focusing on the formal analysis of mappings in ontology-based data access. In particular, the two most important semantic anomalies of mappings have been analyzed: inconsistency and redundancy. Roughly speaking, an inconsistent mapping for an ontology and a source schema is a specification that gives rise to logical contradictions with the ontology and/or the source schema. Then, a mapping $\mathcal{M}$ is redundant with respect to an OBDA specification if adding the mapping $\mathcal{M}$ to the specification does not change its semantics. In [9] we defined both a *local* notion of mapping inconsistency and redundancy, which focuses on single mapping assertions, and a *global* notion, where inconsistency and redundancy is considered with respect to a whole mapping specification (set of mapping assertions).

In this discussion paper we concentrate on global mapping consistency and redundancy, and study the computational complexity of verifying both such properties in an

OBDA specification. We consider a wide range of ontology languages that comprises the description logics underlying OWL 2 and all its tractable profiles[1], and examine mapping languages of different expressiveness (the so-called GAV and GLAV mappings [10]) over sources corresponding to relational databases. We provide algorithms and establish tight complexity bounds for the decision problems associated with global mapping inconsistency and mapping redundancy, for both GAV mappings and a large class of GLAV mappings, and for both combined complexity and TBox complexity (which only considers the size of the TBox). The outcome of our analysis is twofold:

- in our framework, it is possible to define *modular techniques* that are able to reduce the analysis of mappings to the composition of standard reasoning tasks over the ontology and over the data sources. This is a non-trivial result, because mappings are formulas combining both ontology and data source elements;
- the above forms of mapping analysis enjoy *nice computational properties*, in the sense that they are not harder than the above mentioned standard reasoning tasks over the ontology and the data sources.

According to the above results, in our OBDA framework, the analysis of mappings is feasible for languages with nice computational properties, as for the three OWL profiles.

2 Framework

We consider ontologies expressed in some Description Logic (DL) language $\mathcal{L}_{\mathcal{O}}$. DLs allow to represent knowledge in terms of *concepts*, denoting sets of objects, *roles*, i.e., binary relations between concepts, and *attributes*, i.e., binary relations between concepts and data types. A DL ontology $\mathcal{O}$ is pair $\langle \mathcal{T}, A \rangle$, where $\mathcal{T}$ is the TBox, i.e., a finite set of logical axioms specifying intensional knowledge, and $\mathcal{A}$ is the ABox, i.e., a finite set of extensional assertions. $Const(\mathcal{A})$ denotes the set of constants in $\mathcal{A}$. By *ontology inconsistency* we mean the task of deciding whether $\mathcal{O}$ has no models, by *instance checking* the task of deciding whether $\mathcal{O}$ infers a ground atom β, by *conjunctive query (CQ) entailment* the task of deciding whether $\mathcal{O}$ infers a Boolean CQ [4].

An OBDA specification is a triple $\mathcal{J} = \langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$, where $\mathcal{T}$ is a DL TBox, $\mathcal{S}$ is a source schema, and $\mathcal{M}$ is a mapping between the two. In this paper, we consider TBoxes specified through DLs that are the logical basis of the W3C standard OWL and of its profiles, i.e., $\mathcal{SROIQ}$ [6], which underpins OWL, *DL-Lite*$_R$ [5], which is the basis of OWL 2 QL, RL [8], a simplified version of OWL 2 RL, and $\mathcal{EL}_\perp$, a slight extension of the DL $\mathcal{EL}$ [3], which is the basis of OWL 2 EL. The source schema $\mathcal{S}$ is assumed to be relational, and we consider both *simple schemas*, i.e., without integrity constraints, and *FD schemas*, i.e., simple schemas with functional dependencies (FDs). The mapping is a set of assertions m of the form $\phi(\boldsymbol{x}) \rightsquigarrow \psi(\boldsymbol{x})$, where $\boldsymbol{x}$ are the *frontier variables of* m, denoted $FR(m)$, $\phi(\boldsymbol{x})$, called the *body of* m, and $\psi(\boldsymbol{x})$, called the *head of* m, are CQs over $\mathcal{S}$ and $\mathcal{T}$, respectively. We use $head(m)$ and $body(m)$ to denote the head and the body of m, and with a little abuse of notation we will sometime consider them as sets of atoms. Mappings of the form above are called *GLAV* [10]. Besides them, we refer also to *GLAV$_{BE}$* mappings, which are GLAV mappings where $\psi(\boldsymbol{x})$ is a CQ with a

[1] `http://www.w3.org/TR/owl2-profiles/`

bounded number of occurrences of existential variables, and to *GAV* mappings, which are GLAV mappings where $head(m)$ does not admit existential variables.

The semantics of an OBDA specification $\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$ is given w.r.t. a database D *legal* for $\mathcal{S}$, i.e., an instance for $\mathcal{S}$ that satisfies its constraints. It coincides with the set of first-order interpretations for $\mathcal{T}$ that satisfy both $\mathcal{T}$ and $\mathcal{M}$, called the models of $\mathcal{J}$ w.r.t. D, denoted $Mod(\mathcal{J}, D)$. The notion of TBox satisfaction is the standard one in DL [4], whereas mapping satisfaction corresponds to the classical notion of satisfaction of *sound mappings* used in data integration [10, 11]. If $Mod(\mathcal{J}, D) = \emptyset$ we say that $(\mathcal{J}, D)$ is inconsistent. We also say that a mapping assertion m *is active on* D if the evaluation of $body(m)$ over D is non-empty, and that $\mathcal{M}$ is active on D if all its mapping assertions are active on D. Finally, $Const(D)$ denotes the set of constants in D.

Example 1. Let $\mathcal{S}$ be a source schema where the `plants` relation contains data on extraction facilities, while the `eZones` relation contains data on the areas used for oil and gas extraction. Below, the underlined attributes represent the keys of the relations.

$$\texttt{plants}(\underline{\texttt{id\_pl}}, \texttt{pl\_typ}, \texttt{id\_zn}) \qquad \texttt{eZones}(\underline{\texttt{id\_zn}}, \texttt{zn\_typ})$$

The following RL TBox models a very small portion of the domain of oil and gas production extracted from an ontology developed within the Optique EU project[2]. the TBox focuses on the facilities (concept Facility) used in the oil and gas extraction and on the geographical areas (concept Area) in which they are located (role locatedIn).

$$\mathcal{T} = \{\ \mathsf{Platform} \sqsubseteq \mathsf{Facility}\ (1),\quad \mathsf{MarArea} \sqsubseteq \mathsf{Area}\ (2),\quad \exists \mathsf{locatedIn} \sqsubseteq \mathsf{Facility}\ (3),$$
$$\exists \mathsf{locatedIn}^- \sqsubseteq \mathsf{Area}\ (4), \mathsf{Facility} \sqcap \mathsf{Area} \sqsubseteq \bot\ (5), \exists \mathsf{locatedIn}.\mathsf{MarArea} \sqsubseteq \mathsf{Platform}\ (6)\ \}$$

The TBox specifies that a Platform is a Facility (1), a MarArea is an Area (2), every object that is locatedIn somewhere is a Facility (3), every object in which something is locatedIn is an Area (4), Facility and Area are disjoint concepts (5), and every object that is locatedIn a MarArea is a Platform (6).
An example of GAV mapping $\mathcal{M}$ between $\mathcal{T}$ and $\mathcal{S}$ follows:

$$m_1 :\ \texttt{plants}(x, y, z) \rightsquigarrow \mathsf{Facility}(x), \mathsf{locatedIn}(x, z)$$
$$m_2 :\ \texttt{plants}(x', \text{`pl'}, y') \rightsquigarrow \mathsf{Platform}(x')$$
$$m_3 :\ \texttt{eZones}(z', \text{`mz'}) \rightsquigarrow \mathsf{MarArea}(z').$$

□

W.l.o.g. we assume that different mapping assertions use different variable symbols. A *freeze* of a set of atoms Γ is a set of ground atoms obtained from Γ by replacing every variable with a *fresh* distinct constant, i.e., a constant that does not occur elsewhere. Different freezes of the same set of atoms are equal up to a renaming of constants. Thus we assume that the freeze of a set of atoms Γ is unique and is obtained by replacing each variable occurrence x with a fresh constant c_x, and denote it by $freeze(\Gamma)$.

Given a mapping assertion m and an n-tuple of constants $\boldsymbol{t}$, we denote by $m(\boldsymbol{t})$ the mapping assertion obtained by replacing $FR(m)$ in m with the constants in $\boldsymbol{t}$.

Given a mapping $\mathcal{M}$ from $\mathcal{S}$ to $\mathcal{T}$ and legal source instance D, the set $Retr(\mathcal{M}, D) = freeze(\{head(m(\boldsymbol{t})) \mid m \in \mathcal{M} \wedge \boldsymbol{t} \in eval(body(m), D)\})$ is the *ABox retrieved by* $\mathcal{M}$ *from* D.[3]

[2] `http://www.optique-project.eu/`

[3] The definition of retrieved ABox we give in this paper is simplified for ease of exposition. The full definition takes into consideration the binding of frontier variables with constants of $\boldsymbol{t}$.

Below we recall the definitions given in [9] that formalize the mapping analysis services that we study in this paper. Given a TBox $\mathcal{T}$, a source schema $\mathcal{S}$, a mapping assertion m, and a mapping $\mathcal{M}$, we have that:

- $\mathcal{M}$ *is globally inconsistent for* $\langle \mathcal{T}, \mathcal{S} \rangle$ if there does not exist a source instance D legal for $\mathcal{S}$ such that $\mathcal{M}$ is active on D and $Mod(\mathcal{J}, D) \neq \emptyset$. For example, assume that $\mathcal{M}$ contains the following mapping assertions:

  $m_1 : \texttt{plants}(x, y, z) \rightsquigarrow \mathsf{Area}(x)$
  $m_2 : \texttt{plants}(x', \text{'pl'}, z') \rightsquigarrow \mathsf{Platform}(x'), \mathsf{locatedIn}(x', z')$.

  $\mathcal{M}$ is globally inconsistent for $\langle \mathcal{T}, \mathcal{S} \rangle$ given in Example 1, because $\mathcal{T} \models \mathsf{Platform} \sqcap \mathsf{Area} \sqsubseteq \bot$ and every activation of m_2 also activates m_1, thus implying $\mathsf{Platform}(x)$ and $\mathsf{Area}(x)$ for the same individual x.
- A mapping $\mathcal{M}'$ *is globally redundant for* $\mathcal{J} = \langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$, if, for every source instance D that is legal for $\mathcal{S}$, $Mod(\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle, D) = Mod(\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \cup \mathcal{M}' \rangle, D)$. For example, consider the following mapping assertions:

  $m_1 : \texttt{plants}(x, y, z), \texttt{eZones}(z, \text{'mz'}) \rightsquigarrow \mathsf{locatedIn}(x, z)$
  $m_2 : \texttt{eZones}(x', \text{'mz'}) \rightsquigarrow \mathsf{MarArea}(x')$
  $m_3 : \texttt{plants}(y', \text{'pl'}, z'), \texttt{eZones}(z', \text{'mz'}) \rightsquigarrow \mathsf{Platform}(y')$

  Then, $\{m_3\}$ is globally redundant for $\langle \mathcal{T}, \mathcal{S}, \{m_1, m_2\} \rangle$, where $\mathcal{T}$ and $\mathcal{S}$ are as in Example 1.

3 Complexity of mapping inconsistency

We now study global mapping inconsistency and show that this problem has the same TBox complexity of ontology inconsistency in DL. We also establish combined complexity for the DL languages considered in this paper.

We start with some auxiliary definitions. Let $\mathcal{M}$ be a mapping and let $\mathcal{S}$ be a source schema. A *minimal instance for* $\mathcal{S}$ *that activates* $\mathcal{M}$ is a source instance D legal for $\mathcal{S}$ such that $\mathcal{M}$ is active on D and, for every source instance D' legal for $\mathcal{S}$ such that $\mathcal{M}$ is active on D', there exists a homomorphism h from $Const(D)$ to $Const(D')$ that maps constants occurring in $\mathcal{M}$ to themselves and is such that $h(D) \subseteq D'$, where $h(D) = \{r(h(c_1), \ldots, h(c_n)) \mid r(c_1, \ldots, c_n) \in D\}$.

Then, we define the algorithm $\mathsf{freezeFD}(\mathcal{M}, \mathcal{S})$, which takes as input a mapping $\mathcal{M}$ and a source schema $\mathcal{S}$, and applies the *chase* procedure [1] to the database instance $D = \bigcup_{m \in \mathcal{M}} \mathit{freeze}(\mathit{body}(m))$ using the functional dependencies of $\mathcal{S}$, and considering the constants occurring in D but not occurring in $\mathcal{M}$ as unifiable terms (since they act as "soft constants" differently from the constants occurring in $\mathcal{M}$). Such a chase procedure runs in PTIME and may end up in two ways: (i) it fails, i.e., it derives that two constants occurring in $\mathcal{M}$ should be equal; (ii) it returns a database D' that is obtained from D by unifying constants occurring in D but not occurring in $\mathcal{M}$ according to the equalities induced by the functional dependencies. It is not difficult to show that the database returned by $\mathsf{freezeFD}(\mathcal{M}, \mathcal{S})$ is a minimal instance for $\mathcal{S}$ that activates $\mathcal{M}$.

Theorem 1. *Let $\mathcal{J} = \langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$ be an OBDA specification. Then, $\mathcal{M}$ is globally inconsistent for $\langle \mathcal{T}, \mathcal{S} \rangle$ iff either* $\mathsf{freezeFD}(\mathcal{M}, \mathcal{S})$ *fails or the instance D returned by* $\mathsf{freezeFD}(\mathcal{M}, \mathcal{S})$ *is such that $(\mathcal{J}, D)$ is inconsistent.*

The above theorem immediately implies the following algorithm for deciding the global inconsistency of a GLAV mapping $\mathcal{M}$ for a TBox $\mathcal{T}$ and a source schema $\mathcal{S}$.

Algorithm GlobalInconsistency:
Input: OBDA specification $\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$
if (a) algorithm freezeFD$(\mathcal{M}, \mathcal{S})$ fails
then return true
else
let D be the instance returned by freezeFD$(\mathcal{M}, \mathcal{S})$;
if (b) $(\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle, D)$ is inconsistent
then return true **else** return false

Step (a) of the algorithm (i.e., the execution of freezeFD$(\mathcal{M}, \mathcal{S})$) runs in PTIME. It remains to analyze the complexity of checking inconsistency of $(\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle, D)$.

We first notice that to decide inconsistency of $(\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle, D)$, we can compute the ABox $\mathcal{A} = Retr(\mathcal{M}, D)$ and then check inconsistency of $\langle \mathcal{T}, \mathcal{A} \rangle$. Now, observe that the cost of computing $Retr(\mathcal{M}, D)$ does not depend on the size of the TBox. This implies that, with respect to TBox complexity, the complexity of ontology inconsistency is an upper bound for global mapping inconsistency. Conversely, ontology inconsistency can be easily reduced to global mapping inconsistency, by creating a GAV mapping assertion (with no frontier variables) whose head is the conjunction of the ABox assertions in $\mathcal{A}$. Consequently, the following result holds.

Theorem 2. *For both simple and FD schemas, for both GAV and GLAV mappings, and for every ontology language $\mathcal{L}_{\mathcal{O}}$, the TBox complexity of global mapping inconsistency is the same as the TBox complexity of ontology inconsistency in $\mathcal{L}_{\mathcal{O}}$.*

To establish combined complexity, we define a second way to decide inconsistency of $(\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle, D)$, since computing the retrieved ABox $Retr(\mathcal{M}, D)$ requires exponential time in combined complexity. Let $\mathcal{M}$ be a GLAV_{BE} mapping, and let D be a source instance. It is possible to show that the size of $Retr(\mathcal{M}, D)$ is polynomial with respect to the size of $\mathcal{M}$ and D.[4] It follows that, in the case of GLAV_{BE} mappings, inconsistency of $(\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle, D)$ can be decided by checking the existence of a polynomial subset $\mathcal{A}'$ of $Retr(\mathcal{M}, D)$ such that $\langle \mathcal{T}, \mathcal{A}' \rangle$ is inconsistent.

Given a mapping assertion m, a *grounding for* m is the mapping assertion obtained from m by replacing every variable in m with a constant symbol. A grounding for a mapping $\mathcal{M}$ is a set $\{m_g \mid \exists\, m \in \mathcal{M} \text{ s.t. } m_g \text{ is a grounding for } m\}$. Now let D be a source instance. A grounding $\mathcal{G}$ for $\mathcal{M}$ is *generated by* D if, for every $m_g \in \mathcal{G}$, every atom in $body(m_g)$ occurs in D. Given a grounding $\mathcal{G}$ for $\mathcal{M}$, the *ABox induced by* $\mathcal{G}$, denoted as $\mathcal{A}(\mathcal{G})$, is defined as the set of atoms occurring in the heads of the mapping assertions of $\mathcal{G}$. The following algorithm that makes use of the notion of grounding introduced above is able to decide inconsistency of $(\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle, D)$.

Algorithm OBDAInconsistency:
Input: OBDA specification $\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$, with GLAV_{BE} mapping, source instance D
if there exists a polynomial grounding $\mathcal{G}$ for $\mathcal{M}$
such that $\mathcal{G}$ is generated by D **and** the ontology $\langle \mathcal{T}, \mathcal{A}(\mathcal{G}) \rangle$ is inconsistent
then return true **else** return false

[4] Notice that this property does not hold for arbitrary GLAV mappings.

We are now able to analyze the combined complexity of the algorithm GlobalInconsistency when step (b) is executed through the algorithm OBDAInconsistency. As shown before step (a) can always be executed in PTIME. Then, when we execute check (b) we obtain an NP upper bound for those DLs for which ontology consistency is in PTIME, e.g., for *DL-Lite*$_R$, RL, and $\mathcal{EL}_\perp$, and a N2EXPTIME upper bound for $\mathcal{SROIQ}$, for which ontology consistency is already in N2EXPTIME. Such bounds are in fact exact, as stated by the following theorem.

Theorem 3. *For both simple and FD schemas, and for both GAV and GLAV*$_{BE}$ *mappings, the combined complexity of global mapping inconsistency is: (i) NP-complete if the ontology language is DL-Lite*$_R$*,* RL*, or* $\mathcal{EL}_\perp$*, and (ii) N2EXPTIME-complete if the ontology language is* $\mathcal{SROIQ}$*.*

4 Complexity of mapping redundancy

We now show that global mapping redundancy has the same TBox complexity as instance checking for GAV mappings and as CQ entailment over an ontology for GLAV mappings. We also study the combined complexity for the DLs considered in this paper.

We observe that a mapping $\mathcal{M}'$ is globally redundant for an OBDA specification iff each subset of $\mathcal{M}'$ is redundant. We thus pose $\mathcal{M}' = \{m\}$. From now on, we do not consider the trivial case when m is body-inconsistent for $\mathcal{S}$. Under the body-consistency assumption, a minimal instance for $\mathcal{S}$ that activates $\{m\}$ always exists. However, all the complexity results of this section also hold without this assumption.

Theorem 4. *Let* $\mathcal{J} = \langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$ *be an OBDA specification and* m *a mapping assertion. Then,* m *is globally redundant for* $\mathcal{J}$ *iff there exists a minimal instance* D *for* $\mathcal{S}$ *that activates* $\{m\}$ *such that* $Mod(\mathcal{J}, D) = Mod(\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \cup \{m\} \rangle, D)$*.*

Based on the above theorem, below we provide an algorithm that establishes whether m is globally redundant for $\mathcal{J}$ by checking whether a suitable Boolean CQ is entailed by $\mathcal{J}$ coupled with the minimal instance that activates $\{m\}$ returned by the algorithm freezeFD$(\{m\}, \mathcal{S})$. In the following, with a little abuse of notation, we denote with *freeze*$(FR(m))$ the tuple obtained by freezing the frontier variables of m.

Algorithm mapRedundancy:
Input: OBDA specification $\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$, mapping assertion m
(a) $D \leftarrow$ freezeFD$(\{m\}, \mathcal{S})$;
let σ be the substitution derived by freezeFD$(\{m\}, \mathcal{S})$;
$\boldsymbol{t}_F \leftarrow \sigma(\mathit{freeze}(FR(m)))$;
$q_h \leftarrow \mathit{head}(m(\boldsymbol{t}_F))$;
if (b) $(\mathcal{J}, D) \models q_h$
then return true **else** return false

In the algorithm, σ denotes the substitution of terms derived by the application of freezeFD$(\{m\}, \mathcal{S})$, i.e., $\sigma = \{x_1 \to y_1, \ldots, x_n \to y_n\}$ where each y_i is a constant (either fresh or non-fresh) and each x_i is a fresh constant in *freeze*$(\mathit{body}(m))$; σ is applied to the tuple obtained by freezing the frontier variables of m, in order to propagate the term substitutions derived by the chase. The resulting tuple is denoted $\boldsymbol{t}_F$. Notice that, for simple source schemas, σ is the identity and thus it has no effect. Finally, mapRedundancy verifies whether the Boolean query q_h, corresponding to the head of the mapping m whose frontier variables are substituted with $\boldsymbol{t}_F$, is entailed by $(\mathcal{J}, D)$.

The following theorem states that mapRedundancy is sound and complete with respect to the problem of establishing global mapping redundancy.

Theorem 5. *Let $\mathcal{J} = \langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$ be an OBDA specification and m a mapping assertion. Then, m is globally redundant for $\mathcal{J}$ iff mapRedundancy$(\mathcal{J}, m)$ returns true.*

As said in Section 3, step (a) can be executed in PTIME for both simple and FD schemas. As for step (b), the first technique we present is tailored to establish TBox complexity. The property that we exploit says that $(\mathcal{J}, D) \models q_h$ iff $\langle \mathcal{T}, Retr(\mathcal{M}, D) \rangle \models q_h$. Since both step (a) in mapRedundancy and the size of $Retr(\mathcal{M}, D)$ do not depend on the TBox $\mathcal{T}$, for TBox complexity we have that:

- In the case of GAV mappings, the check in step (b) corresponds to a linear number (in the size of $head(m)$) of instance checking tasks in the language $\mathcal{L}_\mathcal{O}$ used for $\mathcal{T}$.
- In the case of GLAV mappings, the check in step (b) corresponds to a single Boolean CQ entailment task in $\mathcal{L}_\mathcal{O}$.

Thus, mapRedundancy, together with the techniques for step (a) and (b) discussed above, allows us to obtain upper bounds for the TBox complexity of global mapping redundancy. More precisely, the complexity of instance checking in $\mathcal{L}_\mathcal{O}$ is an upper bound for GAV mappings, while the complexity of CQ entailment in $\mathcal{L}_\mathcal{O}$ is an upper bound for GLAV. Such bounds are in fact exact, as stated by the following theorem.

Theorem 6. *For both simple and FD schemas, and for every ontology language $\mathcal{L}_\mathcal{O}$, the TBox complexity of global mapping redundancy for GAV and GLAV mappings is the same as the TBox complexity of instance checking in $\mathcal{L}_\mathcal{O}$ and TBox complexity of CQ entailment in $\mathcal{L}_\mathcal{O}$, respectively.*

Similarly to the case of global mapping inconsistency, to establish combined complexity of global mapping redundancy we need to resort to a different strategy for step (b). Namely, we exploit the following algorithm for checking CQ entailment over an OBDA specification $\mathcal{J}$ with GLAV_{BE} mappings and a source instance D.

Algorithm CQEntailment:
Input: OBDA specification $\langle \mathcal{T}, \mathcal{S}, \mathcal{M} \rangle$,with GLAV_{BE} mapping, source instance D, CQ q
  if there exists a polynomial grounding $\mathcal{G}$ for $\mathcal{M}$
  such that $\mathcal{G}$ is generated by D
  and $\langle \mathcal{T}, \mathcal{A}(\mathcal{G}) \rangle \models q$
  then return true **else** return false

It can be shown that in the case of GLAV_{BE} mappings, we can perform step (b) of the algorithm mapRedundancy by executing CQEntailment$(\mathcal{J}, D, q_h)$.

As for combined complexity, in the following we consider simple source schemas for the lower bounds and FD source schemas for the upper bounds. First, step (b) can be executed through the nondeterministic algorithm CQEntailment. Consequently, this algorithm provides an NP upper bound for the case of GLAV_{BE} mappings if, for the ontology language $\mathcal{L}_\mathcal{O}$, CQ entailment is in NP, i.e., for *DL-Lite*$_R$, RL, and $\mathcal{EL}_\bot$. The matching NP lower bounds can be proved already for GAV mappings, by an easy reduction of conjunctive query containment in relational databases. In the case of $\mathcal{SROIQ}$, for GLAV$_{\text{BE}}$ mappings we are not able to even prove decidability of global mapping redundancy (since decidability of CQ entailment in this language is currently an open problem too), while for the GAV case we can easily derive a N2EXPTIME exact bound.

Theorem 7. *For both simple and FD source schemas: (i) in the case of DL-Lite*$_R$*,* RL*, or* $\mathcal{EL}_\bot$*, global mapping redundancy is NP-complete w.r.t. combined complexity for both GAV and GLAV*$_{BE}$ *mappings; (ii) in the case of* $\mathcal{SROIQ}$*, global mapping redundancy is N2EXPTIME-complete w.r.t. combined complexity for GAV mappings.*

5 Conclusions

The analysis presented in this paper can be extended in different directions. We aim to establish tight combined complexity bounds for general GLAV mappings, and extend our study to other forms of mappings (beyond GLAV), admitting, for instance, forms of negation in the source queries. We also want to consider settings that go beyond the OWL framework, considering, for instance, languages of the Datalog+/- family.

Acknowledgments. This research has been partially supported by the EU under FP7 Large-scale integrating project Optique (grant n. FP7-318338).

References

1. S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison Wesley Publ. Co., 1995.
2. N. Antonioli, F. Castanò, S. Coletta, S. Grossi, D. Lembo, M. Lenzerini, A. Poggi, E. Virardi, and P. Castracane. Ontology-based data management for the italian public debt. In *Proc. of FOIS*, pages 372–385, 2014.
3. F. Baader, S. Brandt, and C. Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proc. of IJCAI*, pages 364–369, 2005.
4. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2nd edition, 2007.
5. 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. *JAR*, 39(3):385–429, 2007.
6. I. Horrocks, O. Kutz, and U. Sattler. The even more irresistible $\mathcal{SROIQ}$. In *Proc. of KR*, pages 57–67, 2006.
7. E. Kharlamov et al. Optique 1.0: Semantic access to big data: The case of norwegian petroleum directorate's factpages. In *Proc. of ISWC*, pages 65–68, 2013.
8. R. Kontchakov and M. Zakharyaschev. An introduction to description logics and query rewriting. In *RW Tutorial Lectures*, number 8714 in LNCS, pages 195–244. Springer, 2014.
9. D. Lembo, J. Mora, R. Rosati, D. F. Savo, and E. Thorstensen. Towards mapping analysis in ontology-based data access. In *Proc. of RR*, pages 108–123, 2014.
10. M. Lenzerini. Data integration: A theoretical perspective. In *Proc. of PODS*, pages 233–246, 2002.
11. 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.
12. M. Rodriguez-Muro, R. Kontchakov, and M. Zakharyaschev. Ontology-based data access: Ontop of databases. In *Proc. of ISWC*, volume 8218 of *LNCS*, pages 558–573. Springer, 2013.