ECAI 2014
T. Schaub et al. (Eds.)

© 2014 The Authors and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License.
doi:10.3233/978-1-61499-419-0-219

# Reducing global consistency to local consistency in Ontology-based Data Access

**Marco Console, Maurizio Lenzerini**
**Dipartimento di Ingegneria Informatica, Automatica e Gestionale "Antonio Ruberti'**
**Sapienza Università di Roma, Roma, Italy**

**Abstract.** Ontology-based data access (OBDA) is a new paradigm aiming at accessing and managing data by means of an ontology, i.e., a conceptual representation of the domain of interest in the underlying information system. In the last years, this new paradigm has been used for providing users with suitable mechanisms for querying the data residing at the information system sources. Most of the research has been concentrating on making query answering efficient. However, query answering is not the only service that an OBDA system must provide. Another crucial service is consistency checking. Current approaches to this problem involves executing expensive queries at run-time. In this paper we address a fundamental problem for OBDA system: given an OBDA specification, can we avoid the consistency check on the whole OBDA system (global consistency check), and rely instead on the constraint checking carried out by the DBMS on the data source (local consistency checking)? We present algorithms and complexity analysis for this problem, showing that it can be solved efficiently for a class of OBDA systems that is very relevant in practice.

## 1 Introduction

Ontology-based data access (OBDA) is a new paradigm aiming at accessing and managing the data of an information system by means of an ontology [16]. An OBDA system is constituted by an OBDA specification, representing the intensional level of the system, and a data source (or, a collection of data sources), representing its extensional level. In turn, the OBDA specification is composed of three elements, namely the ontology, the data source schema, and the mapping between the source schema and the ontology. The ontology provides a conceptual representation of the domain of interest, independently from how the individual objects of the domain are structured in the the data sources. The mapping is a set of assertions that specify how the data at the sources can be aggregated in order to form the instances of the concepts and the relationships in the ontology.

Depending on the process defining the OBDA specification, there are two types of OBDA systems, that we call simple and composite. A simple OBDA system models a top-down scenario in which the information system is designed starting from the ontology, and the data sources are defined with the goal of providing correct data structures for storing the instances of concepts and roles in the ontology. A composite OBDA system is built using a bottom-up approach, where the ontology is linked to a set of pre-existing and autonomous data sources, so as to improve the access to the data by the users, who can greatly benefit by an abstract representation of the information system. In both cases, the data sources are usually managed by a data management system. In particular, in this paper we will assume the common situation where data at the sources are stored in a relational database managed by a Data Base Management System (DBMS). In this scenario, the DBMS is responsible of carrying out the task of enforcing consistency of data with respect to a set of integrity constraints, expressed over the source schema.

In the last years, this new paradigm has been adopted for designing innovative mechanisms allowing the users to query the information system through the ontology [18, 19, 7, 14, 5, 4]. There has been a large body of scientific papers addressing the following fundamental problem: can we answer queries posed to the ontology efficiently, at least efficiently with respect to the size of data at the source? A fundamental result in this sense has been provided in [9], where it is shown that there is a class of languages, namely the *DL-Lite* family of Description Logics, that allows for first-order rewriting of (unions of) conjunctive queries. More precisely, given a (union) of conjunctive query expressed over the ontology, one can always compute a first-order perfect rewriting of the query, i.e., an SQL query over the source schema, such that the evaluation of this query over the source data yields exactly the certain answers to the original ontology query. Starting with this results, much research has concentrated on improving the size and the form of the rewriting has produced many interesting results, and is still very active. The ultimate goal is to design sophisticated algorithms for devising efficient query answering systems that can be applied in real world applications, where the size of both the ontology and the source data can be very large.

However, query answering is not the only service that an OBDA system must provide. Another crucial service is consistency checking. Indeed, if the whole system is inconsistent, classical query answering becomes meaningless, and we have to either discard the data at the sources, or to resort to more sophisticated strategies (i.e., inconsistency tolerant query answering) in order to extract useful information from the data. Currently, in OBDA systems based on *DL-Lite*, consistency checking is realized by resorting to query rewriting: more precisely, checking consistency involves computing the certain answers to a specific union of conjunctive query computed on the basis of the axioms in the ontology. Although acceptable, this method is not ideal. Indeed, on one hand, the query usually involves the exploration of a very large portion of source data, and on the other hand, its cost grows with the number of axioms in the ontology, and can constitute a bottleneck in the use of the system. Indeed we observe that, in principle, global consistency checking should be carried out whenever data at the sources change.

In this paper we address a fundamental problem for OBDA system: given an OBDA specification, can we avoid the consistency check on the whole OBDA system, and rely instead on the constraint enforcement carried out by the DBMS on the data source (local consistency checking) in order to ensure global consistency? Note that, if the answer is positive, we can indeed avoid to reason on the whole system for carrying out the consistency checking at run time: whenever the DBMS accepts a database at the source, we know that its data are consistent with the OBDA system. In other words, we know that we can reduce global consistency to local consistency.

In the next sections, we present the first study of this issue, and provide the following contributions. We present a formal framework for characterizing the relation among global and local consistency in OBDA system (section 3). We actually split the relation in two parts, i.e. protection and faithfulness. Note that faithfulness and protection are discussed also in [10], but with the goal of studying methods for characterizing the quality of data sources. For both the notions, we present checking algorithms and complexity results in the case of OBDA specified using *DL-Lite*$_\mathcal{R}$, GLAV mappings, and relational schema with keys, foreign keys and denial constraints (Section 5). We also show technical results at the basis of those algorithms: in particular we illustrate two relevant tasks for reasoning over an OBDA system whose data layer is constituted by a database with incomplete information (Section 4).

Sections 2 and 6 complete the paper. The former presents preliminary notions concerning relational databases, Description Logics, and OBDA systems. The latter concludes the paper by illustrating future directions of our work.

## 2 Preliminaries

In this section we present some preliminary notions on databases and ontologies that are at the basis of the OBDA approach, and we illustrate what is an OBDA system.

**Databases.** We consider relational databases, and refer to [1] for a more detailed account of databases. A *schema* $\mathcal{S}$ is a pair $\langle \Sigma_\mathcal{S}, \mathcal{C}_\mathcal{S} \rangle$, where $\Sigma_\mathcal{S}$ is the alphabet of $\mathcal{S}$, and $\mathcal{C}_\mathcal{S}$ is the set of integrity constraints (or simply constraints) of $\mathcal{S}$, which are rules that each database conforming to the schema must obey. A *database* for $\mathcal{S}$, or simply a $\Sigma_\mathcal{S}$-database, is a finite set of ground atoms over the predicates in $\Sigma_\mathcal{S}$ and the constants in an alphabet $\Gamma$ (constants are subject to the unique name assumption). A $\Sigma_\mathcal{S}$-database $D$ is *legal* for $\mathcal{S}$, written $D \models \mathcal{S}$, if satisfies all the integrity constraints in $\mathcal{C}_\mathcal{S}$, written $D \models \mathcal{C}_\mathcal{S}$.

Given $r/t \in \Sigma_\mathcal{S}$, a relation symbol in $\mathcal{S}$ of arity $t$, we call attribute of $r$ an integer ranging from 1 to the $t$. Further, if $r(\mathbf{c})$ is a ground atom, and $A$ is a set of attributes for $r$, then we denote by $\mathbf{c}[A]$ the projection of $r$ onto the attributes in $A$. In general, constraints in $\mathcal{S}$ are expressed as first-order logic (FOL) sentences, or subclasses thereof.

In this paper, we will focus on sets of constraints $\mathcal{C}_S$ constituted by three parts: a set $\mathcal{C}^k_S$ of *single key* dependencies, a set $\mathcal{C}^f_S$ of *foreign key* dependencies, and a set $\mathcal{C}^d_S$ of *denial constraints*.

A *key dependency* (KD for short) has the form $key(r) = A$, where $r$ is a relation symbol in $\Sigma_\mathcal{S}$ and $A$ is a sequence of attributes for $r$. A set of KDs is a set of *single* key dependencies if it contains at most one KD for each relation. A database $D$ satisfies a KD $key(r) = A$ if, for each pair of tuples $r(\mathbf{c}), r(\mathbf{c}') \in D$, if $\mathbf{c}[A] = \mathbf{c}'[A]$ then $\mathbf{c} = \mathbf{c}'$.

An *inclusion dependency* (ID for short) has the form $r[A] \subseteq s[B]$, where $A$ and $B$ are sets of attributes for the relations $r$ and $s$ respectively. A database $D$ satisfies an ID $r[A] \subseteq s[B]$ if, for each ground atom of the form $r(\mathbf{c}) \in D$, there exists a ground atom of the form $s(\mathbf{c}')$ such that $\mathbf{c}[A] = \mathbf{c}'[B]$. Given a set $C^k$ of KDs, a set $C^f$ of IDs is a set of *foreign keys* (FDs for short) for $C^k$ if, for each ID in $C^f$ of the form $r[A] \subseteq s[B]$ and for each KD in $C^k$ of the form $key(s) = C$, we have that $B \subseteq C$.

A *denial constraint* (DC for short) has the form $\beta = \exists \vec{x}.\Phi(\vec{x}) \rightarrow \bot$. A database $D$ satisfies a DC if $\exists \vec{x}.\Phi(\vec{x})$ doesn't hold in $D$.

Sets of constraints containing KDs and FDs have the notably property of conjunctive query answering *finite controllability* (see [20]), i.e. given a database $D$, a conjunctive query $q$ and a set $\mathcal{C}$ of KDs and FDs, the answers to $q$ that are true in all the models of $D \cup I$ coincide with the answers over the *finite* models only. Also, in [6], the authors present a technique for computing the certain answers to conjunctive queries over a knowledge base constituted by a database and a set $\mathcal{C}$ of KDs and FDs. Such certain answers are the tuples that are the answers to a conjunctive query $q$ in every finite databases extending $D$, and satisfying all the constraints in $\mathcal{C}$. In particular, the authors show that the certain answers to $q$ can be computed simply by first computing a new query $\mathcal{R}_\mathcal{C}(q)$, called the perfect rewriting $\mathcal{R}_\mathcal{C}(q)$ of $q$ with respect to $\mathcal{C}$, and then evaluating such query over the database $D$.

**Description Logic Ontologies.** An ontology is a conceptualization of a domain of interest expressed in terms of a formal language. Here, we consider logic-based languages, and, more specifically, Description Logics (DLs) [3]. Generally speaking, a knowledge base expressed in a DL is a pair $\langle \mathcal{T}, \mathcal{A} \rangle$ where the *TBox* $\mathcal{T}$ is the ontology, i.e., a set of axioms specifying universal properties of the concepts and the roles that are relevant in the domain, and the *ABox* $\mathcal{A}$ contains axioms specifying the instances of concepts and roles.

In this paper we focus on ontologies written in the *DL-Lite*$_\mathcal{R}$ [8, 17] formalism, a member of the *DL-Lite* family[1] of tractable Description Logics (DLs). We provide only a short account of *DL-Lite*$_\mathcal{R}$ here.

The syntax of concept, role and attribute *expressions* in *DL-Lite*$_\mathcal{R}$ over an alphabet $\Sigma_\mathcal{T}$ is specified by means of the following grammar (where $A, P, U$ are atomic concepts, roles, and attributes, respectively, and $T_1, \ldots, T_n$ are unbounded pairwise disjoint predefined value-domains):

$$\begin{array}{ll}
B \longrightarrow A \mid \exists Q \mid \delta(U) & E \longrightarrow \rho(U) \\
C \longrightarrow B \mid \neg B & F \longrightarrow T_1 \mid \cdots \mid T_n \\
Q \longrightarrow P \mid P^- & V \longrightarrow U \mid \neg U \\
R \longrightarrow Q \mid \neg Q &
\end{array}$$

A *DL-Lite*$_\mathcal{R}$ TBox $\mathcal{T}$ over an alphabet $\Sigma_\mathcal{T}$ is constituted by:

- the set $\mathcal{T}^+$ of "positive" inclusion assertions between concepts, roles and attributes (e.g., $A \sqsubseteq \exists P^-$);
- the set $\mathcal{T}^-$ of "negative" assertions, i.e. disjointness assertions between concepts, roles and attributes (e.g., $A \sqsubseteq \neg \exists P$);

Note that checking *DL-Lite*$_\mathcal{R}$-KB for satisfiability, i.e., checking if $Mod(\langle \mathcal{T}, \mathcal{A} \rangle) = \{ \mathcal{I} \mid \mathcal{I}$ is an interpretation for $\Sigma_\mathcal{T}$ such that $\mathcal{I} \models \mathcal{T} \}$ is non-empty, can be done in $AC_0$ with respect to $\mathcal{A}$ and in PTIME with respect to $\mathcal{T}$.

**Ontology-based Data Access.** An OBDA system is constituted by an OBDA specification, the intensional level of the system, and a database, representing the data stored in the sources, i.e., the extensional level of the system.

An OBDA specification provides the characteristics of the three basic components of the system, as specified by the following definition.

[1] Not to be confused with the set of DLs studied in [2], which form the *DL-Lite*$_{bool}$ family.

**Definition 1** *An* OBDA specification *$\mathcal{B}$ is a triple $\langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$, where*

- *$\mathcal{T}$ is a TBox, called the ontology of $\mathcal{B}$, with alphabet $\Sigma_{\mathcal{T}}$;*
- *$\mathcal{S} = \langle \Sigma_{\mathcal{S}}, \mathcal{C}_{\mathcal{S}} \rangle$ is a database schema, called the source schema of $\mathcal{B}$;*
- *$\mathcal{M}$ is a finite set of mapping assertions [12, 15]. between $\mathcal{S}$ and $\mathcal{T}$, called the* mapping *of $\mathcal{B}$, where each mapping assertion is of the form $\forall \vec{x} \phi(\vec{x}) \rightarrow \exists \vec{y} \psi(\vec{x}, \vec{y})$, where $\phi(\vec{x})$ is a conjunctive query over $\Sigma_{\mathcal{S}}$ with free variables $\vec{x}$, and $\psi(\vec{x}, \vec{y})$ is a conjunctive query over the alphabet $\Sigma_{\mathcal{T}}$ with free variables $\vec{x} \cup \vec{y}$.*

As we said before, when we pair an OBDA specification $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ with a $\Sigma_{\mathcal{S}}$-database $D$, we obtain an OBDA system. We define the semantics of an OBDA system by specifying which are the models of $\mathcal{B}$ relative to $D$, denoted by $Mod_D(\mathcal{B})$. Intuitively, if $D$ is not legal with respect to $\mathcal{S}$, such models form the empty set. Otherwise, such models are the interpretations $\mathcal{I}$ for $\Sigma_{\mathcal{T}}$ that satisfy $\mathcal{T}$, and such that the pair $(D, \mathcal{I})$ satisfy all mapping assertions in $\mathcal{M}$, written $(D, \mathcal{I}) \models \mathcal{M}$.

**Definition 2** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDA specification, and let $D$ be a $\Sigma_{\mathcal{S}}$-database. Then $Mod_D(\mathcal{B}) = \{ \mathcal{I} \mid \mathcal{I} \models \mathcal{T},\ (D, \mathcal{I}) \models \mathcal{M}, \text{ and } D \models \mathcal{C}_{\mathcal{S}} \}$.*

Checking whether an OBDA system constituted by $\mathcal{B}$ and $D$ is satisfiable amounts to checking whether $Mod_D(\mathcal{B}) \neq \emptyset$. In practice, the system is managed by suitable software components including a database management system ensuring that $D \models \mathcal{C}_{\mathcal{S}}$. This means that the system managing the data source already filters out those databases $D$ such that $D \not\models \mathcal{C}_{\mathcal{S}}$. Thus, in general, the only thing that remains to be done is to check whether $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma, \emptyset \rangle \rangle = \emptyset$, i.e., whether there exists an interpretation $\mathcal{I}$ for $\Sigma_{\mathcal{T}}$ that satisfies $\mathcal{T}$, and such that the pair $(D, \mathcal{I})$ satisfies all mapping assertions in $\mathcal{M}$.

Besides checking consistency, another crucial task to be carried out in an OBDA system is query answering, which amounts to compute the certain answers of a union of conjunctive query. Informally, a certain answer of a query posed to a system constituted by an OBDA specification $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$, and a $\Sigma_{\mathcal{S}}$-database $D$ is a tuple that is an answer to the query in all the models $Mod_D(\mathcal{B})$. The results in [9] and [11] show that the certain answers to a query $q$ posed to the OBDA specification $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ and the $\Sigma_{\mathcal{S}}$-database $D$ can be computed by first computing the ontology rewriting $\mathcal{R}_{\mathcal{T}}(q)$ of $q$ with respect to $\mathcal{T}$ (with the algorithm in [9]), then computing the mapping rewriting $\mathcal{R}_{\mathcal{T}}(\mathcal{R}(_{\mathcal{T}}(q))$, denoted by $\mathcal{R}_{\mathcal{T},\mathcal{M}}(q)$, and finally by evaluating the query $\mathcal{R}_{\mathcal{T},\mathcal{M}}(q)$, which is a union of conjunctive query over $\Sigma_{\mathcal{S}}$, on the database $D$.

## 3 Framework for global and local consistency

In this section, we introduce the notions we use for characterizing global consistency and local consistency in OBDA.

We start with the formal definition of these two notions. What we formalize here is the simple idea that, given an OBDA specification $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \mathcal{C}_{\mathcal{S}} \rangle \rangle$, and a $\Sigma_{\mathcal{S}}$-database $D$, local consistency refers to the relationship between $D$ and $\mathcal{C}_{\mathcal{S}}$, while global consistency focuses on the relationship between $D$, $\mathcal{T}$ and $\mathcal{M}$, independently of $\mathcal{C}_{\mathcal{S}}$.

**Definition 3** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \mathcal{C}_{\mathcal{S}} \rangle \rangle$ be an OBDA specification, and let $D$ be a $\Sigma_{\mathcal{S}}$-database. Then the OBDA system constituted by $\mathcal{B}$ and $D$ is said to be* locally consistent *if $D \models \mathcal{C}_{\mathcal{S}}$, whereas is said to be* globally consistent *if $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \emptyset \rangle \rangle) \neq \emptyset$,*

The above definition captures the idea that, while the domain ontology $\mathcal{T}$ forms the intensional level of the whole system, the database $D$ together with $\mathcal{M}$ determines its extensional level. The schema $\mathcal{S}$ is simply the structure designed for accommodating the data stored at the source, but it does not really contribute to the semantics of the OBDA system. So, the global consistency of the OBDA system does not depend on $\Sigma_{\mathcal{S}}$, and checking global consistency is indeed different from checking satisfiability of $\mathcal{B}$ and $D$, which amounts to check whether $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \mathcal{C}_s \rangle \rangle) \neq \emptyset$. On the other hand, local consistency merely means that the database $D$ is legal with respect to the source schema $\mathcal{S}$, i.e., it satisfies all of its constraints $\mathcal{C}_{\mathcal{S}}$.

Given the above definition, it is immediate to formalize the condition under which global consistency can be reduced to local consistency: global consistency of $\mathcal{B}$ and $D$ can be reduced to local consistency exactly when, for all $\Sigma_{\mathcal{S}}$-databases $D$, $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \emptyset \rangle \rangle) \neq \emptyset$ is equivalent to $D \models \mathcal{C}_{\mathcal{S}}$. In the following, we actually split this notion in two parts, corresponding to the two parts of the equivalence, and we call such parts protection and faithfulness, respectively.

**Definition 4** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDA specification, where $\mathcal{S} = \langle \Sigma_{\mathcal{S}}, \mathcal{C}_{\mathcal{S}} \rangle$. Then, $\mathcal{S}$ is said to protect $\mathcal{T}$ and $\mathcal{M}$ from inconsistency, or simply* protect *$\mathcal{B}$ from inconsistency, if for all $\Sigma_{\mathcal{S}}$-database $D$ such that $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \emptyset \rangle \rangle) = \emptyset$, we have that $D \not\models \mathcal{C}_{\mathcal{S}}$.*

Intuitively, the schema $\mathcal{S}$ protects $\mathcal{B}$ from inconsistency whenever its constraints block every database which would break global consistency. Figure 1 illustrates the notion of protection pictorially. The area shown in the figure represents all the possible $\Sigma_{\mathcal{S}}$-databases. The source schema protects $\mathcal{B}$ from inconsistency exactly when it ensures that the bottom-right quadrant corresponding to $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \emptyset \rangle \rangle) = \emptyset$ and $D \models \mathcal{C}_{\mathcal{S}}$ is empty.

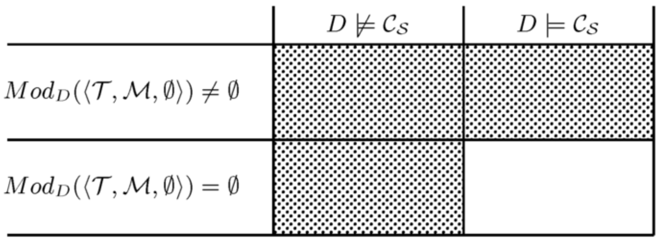
$D \not\models \mathcal{C}_{\mathcal{S}}$
$D \models \mathcal{C}_{\mathcal{S}}$
$Mod_D(\langle \mathcal{T}, \mathcal{M}, \emptyset \rangle) \neq \emptyset$
$Mod_D(\langle \mathcal{T}, \mathcal{M}, \emptyset \rangle) = \emptyset$

**Figure 1.** Protection

**Definition 5** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDA specification, where $\mathcal{S} = \langle \Sigma_{\mathcal{S}}, \mathcal{C}_{\mathcal{S}} \rangle$. Then, $\mathcal{S}$ is said to be faithful to $\mathcal{T}$ and $\mathcal{M}$ in $\mathcal{B}$, or simply* faithful to *$\mathcal{B}$, if for all $\Sigma_{\mathcal{S}}$-database $D$ such that $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \emptyset \rangle \rangle) \neq \emptyset$, we have that $D \models \mathcal{C}_{\mathcal{S}}$.*

Intuitively, the schema $\mathcal{S}$ is faithful to $\mathcal{B}$ if it does not constrain the source in such a way to filter out data that would not cause the OBDA system to fall into inconsistency, i.e., if every $\Sigma_{\mathcal{S}}$-database $D$ that does not cause any inconsistencies to $\mathcal{T}$ and $\mathcal{M}$, does not violate any constraint of $\mathcal{S}$.

Figure 2 illustrates the notion of faithfulness pictorially. The source schema protects $\mathcal{B}$ from inconsistency exactly when it ensures that the top-left quadrant corresponding to $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \emptyset \rangle \rangle) \neq \emptyset$ and $D \not\models \mathcal{C}_{\mathcal{S}}$ is empty.

| | $D \not\models \mathcal{C}_{\mathcal{S}}$ | $D \models \mathcal{C}_{\mathcal{S}}$ |
|---|---|---|
| $Mod_D(\langle \mathcal{T}, \mathcal{M}, \emptyset \rangle) \neq \emptyset$ | | ▨ |
| $Mod_D(\langle \mathcal{T}, \mathcal{M}, \emptyset \rangle) = \emptyset$ | ▨ | ▨ |

**Figure 2.** Faithfulness

We observe that faithfulness is indeed an interesting property. It formalizes the condition under which the semantics of the OBDA system does not depend on the constraints in $\mathcal{C}_{\mathcal{S}}$, as stated in the following theorem.

**Theorem 1** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDA specification, where $\mathcal{S} = \langle \Sigma_{\mathcal{S}}, \mathcal{C}_{\mathcal{S}} \rangle$. Then $\mathcal{S}$ is faithful to $\mathcal{B}$ if and only if $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \mathcal{C}_{\mathcal{S}} \rangle \rangle) = Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \emptyset \rangle \rangle)$.*

Figure 3 combines the two notions of protection and faithfulness, and illustrates the idea that global consistency can be reduced to local consistency exactly when both the top-left and the bottom-right quadrants are empty.

| | $D \not\models \mathcal{C}_{\mathcal{S}}$ | $D \models \mathcal{C}_{\mathcal{S}}$ |
|---|---|---|
| $Mod_D(\langle \mathcal{T}, \mathcal{M}, \emptyset \rangle) \neq \emptyset$ | | ▨ |
| $Mod_D(\langle \mathcal{T}, \mathcal{M}, \emptyset \rangle) = \emptyset$ | ▨ | |

**Figure 3.** Faithfulness and protection

The two notions of protection and faithfulness give raise to two decision problems, that we are going to study in the rest of the paper.

**Definition 6** *Given an OBDA specification $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$, the following are decision problems.*
- *Protection: check whether $\mathcal{S}$ protects $\mathcal{B}$.*
- *Faithfulness: check whether $\mathcal{S}$ is faithful to $\mathcal{B}$.*

In the rest of the paper, we present techniques for the two decision problems. For both faithfulness and protection, we will face the problem of checking whether an incomplete database is consistent with respect to a portion of OBDA specification. We address this issue in the next section.

## 4 OBDA over incomplete databases

Existing works on OBDA concentrate on the case where the data source is a traditional database, without incomplete information. In this section we consider the case where the database at the extensional level has incomplete information [13]. More precisely, we first provide the formal definition of the notion of database with incomplete information that we use, and then we address two specific tasks that are relevant in OBDA systems whose database is incomplete. The results presented in this section will be used in Section 5, where we present the techniques for checking protection and faithfulness.

**Definition 7** *Let $\mathcal{S} = \langle \Sigma_{\mathcal{S}}, \mathcal{C}_{\mathcal{S}} \rangle$ be a schema, and let $V$ be an alphabet of variables. An incomplete $\Sigma_{\mathcal{S}}$-database is a $\Sigma_{\mathcal{S}}$-database where variables from $V$ can appear in the arguments of relations.*

Intuitively, an incomplete database $F$ represents all the (complete) databases that are subsets of the set of facts obtained by choosing a set of constants, and then instantiating the variables of $F$ with such constants. Formally, we say that a (complete) $\Sigma_{\mathcal{S}}$-database database $D$ is an *instance* of an incomplete $\Sigma_{\mathcal{S}}$-database $F$, written $D \preceq F$, if there is a homomorphism from $F$ to $D$, i.e., a function $h$ from the constants in $\Sigma_{\mathcal{S}}$ and the variables in $V$ to the constants of $\Sigma_{\mathcal{S}}$ such that $(i)$ $h(c) = c$ for each constant $c$; $(ii)$ $R(t_1, \ldots, t_n)$ is in $F$ if and only if $R(h(t_1), \ldots, h(t_n))$ is in $D$.

The semantics of an OBDA specification with respect to an incomplete database can be given by resorting to the traditional semantics of the OBDA specification, as follows.

**Definition 8** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDA specification, and let $F$ be an incomplete $\Sigma_{\mathcal{S}}$-database. Then $Mod_F(\mathcal{B}) = \{ \mathcal{I} \mid$ there exists a $\Sigma_{\mathcal{S}}$-databases $D$ such that $D \preceq F$, and $Mod_D(\mathcal{B}) \neq \emptyset \}$.*

In principle, it would be interesting to study various reasoning tasks, such as query answering, for OBDA systems constituted by an OBDA specification and an incomplete database. However, such investigation is outside the scope of this paper. Rather, as we said before, in this paper we are interested in two types of tasks: $(i)$ checking consistency of an incomplete database with the constraints $\mathcal{C}_{\mathcal{S}}$, $(ii)$ checking consistency of an OBDA specification $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \langle \Sigma_{\mathcal{S}}, \emptyset \rangle \rangle$ with respect to an incomplete database $F$.

**Consistency with respect to a set of constraints.** The first task can be defined formally as follows: given a source schema $\langle \Sigma_{\mathcal{S}}, \mathcal{C}_{\mathcal{S}} \rangle$, where $\mathcal{C}_{\mathcal{S}}$ is a set of keys, foreign keys, and denials, and a $\Sigma_{\mathcal{S}}$-database with incomplete information $F$, we want to decide whether there exists a complete $\Sigma_{\mathcal{S}}$-database $D$ such that $D \models \mathcal{C}_{\mathcal{S}}$ and $D \preceq F$. The solution we present extends the techniques proposed in [6], in order to take into account denial constraints. To present out solution, we now introduce four notions.

The first ingredient of our solution is the notion of chase of $F$ with respect to the key dependencies $\mathcal{C}^k_{\mathcal{S}}$ in $\mathcal{C}_{\mathcal{S}}$. This is just the traditional chase that repeatedly applies the following rule to an incomplete database $T$, starting from $F$: if $key(R) = \{i_1, \ldots, i_n\}$ is in $\mathcal{C}^k_{\mathcal{S}}$, and $T$ contains the facts $R(x_1, \ldots, x_m), R(y_1, \ldots, y_m)$ where for every $i \in \{i_1, \ldots, i_n\}$, $x_i = y_i$, then the rule makes the two atoms $R(x_1, \ldots, x_m), R(y_1, \ldots, y_m)$ equal in $T$. Intuitively, the chase rule simply "repairs" a key dependency in $T$, by suitably equating variables. The second notion refers to the trivial task of obtaining a complete database (database with constants) $\mathcal{F}(T)$ from an incomplete database $T$: we do so simply by choosing one constant for every variable in $T$ in such a way that different constants are chosen for different variables, and then substituting every variable with the corresponding constant. The third ingredient is the notion of *violation query* associated to a denial in $\mathcal{C}^d_{\mathcal{S}}$. Given a denial $\beta = \exists \vec{x}.\Phi(\vec{x}) \rightarrow \bot$, we define the violation query $\mathcal{V}(\beta)$ associated to $\beta$ as the conjunctive query $\exists \vec{x}.\Phi(\vec{x})$. Intuitively, this is the query that evaluates to true over a database $D$ exactly when such database violates the denial. Finally, the fourth notion that we introduce here is the notion of *perfect rewriting* of a violation query associated to a denial. Given a denial $\beta \in \mathcal{C}^d_{\mathcal{S}}$, the perfect rewriting of $\mathcal{V}(\beta)$ with respect to $\mathcal{C}^f_{\mathcal{S}}$ is the query $\mathcal{R}_{\mathcal{C}^f_{\mathcal{S}}}(\mathcal{V}(\beta))$ obtained by applying the algorithm `IdRewrite` presented in [6] to the query $\mathcal{V}(\beta)$ using the foreign key in $\mathcal{C}^f_{\mathcal{S}}$ as inclusion dependencies. From the properties of this algorithm, we can conclude that a violation of $\beta$ exists in all complete databases in $\{ D \mid D \preceq F \}$ if and only if $\mathcal{R}_{\mathcal{C}^f_\mathcal{S}}(\mathcal{V}(\beta))$ evaluates to true over $\mathcal{F}(chase_{\mathcal{C}^k_\mathcal{S}}(F))$.

**Theorem 2** *Let $F$ be a $\Sigma_\mathcal{S}$-database with incomplete information, and let $\mathcal{C}_\mathcal{S}$ be a set of keys, foreign keys and denials. Then $F$ is consistent with $\mathcal{C}_\mathcal{S}$ if and only if the following two conditions hold:*

1. *$chase_{\mathcal{C}^k_\mathcal{S}}(F)$ does not fail, and*
2. *for no $\beta \in \mathcal{C}^d_\mathcal{S}$ we have that $\mathcal{R}_{\mathcal{C}^f_\mathcal{S}}(\mathcal{V}(\beta))$ evaluates to true over $\mathcal{F}(chase_{\mathcal{C}^k_\mathcal{S}}(F))$.*

From the above theorem, it is immediate to derive an algorithm for our task. From the correctness of the above algorithm, we can then derive the following result.

**Theorem 3** *Checking whether $F$ is consistent with $\mathcal{C}_\mathcal{S}$ can be done in PTIME with respect to $F$, $\mathcal{C}^f_\mathcal{S}$, and $\mathcal{C}^f_\mathcal{S}$, and in NP with respect to $\mathcal{C}^d_\mathcal{S}$.*

**Consistency with respect to an OBDA specification.** The second task consists of checking consistency of an incomplete database $F$ with respect to the ontology $\mathcal{T}$ and the mapping $\mathcal{M}$ of an OBDA specification $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle$, i.e., checking whether $Mod_F(\mathcal{B}) \neq \emptyset$. Note that, here, we are not considering the constraints of $\mathcal{S}$. The following theorem presents the fundamental result for this problem.

**Theorem 4** *$Mod_F(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) = \emptyset$ if and only if there exists $\alpha \in \mathcal{T}^-$ such that the certain answers of $\mathcal{V}(\alpha)$ with respect to $\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle$ and $\mathcal{F}(F)$ is false.*

The above theorem directly suggests the following algorithm for checking whether $Mod_F(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) \neq \emptyset$:

1. build a complete $\Sigma_\mathcal{S}$-database $\mathcal{F}(F)$ from the incomplete $\Sigma_\mathcal{S}$-database $F$;
2. check whether for all $\alpha \in \mathcal{T}^-$, we have that the answer to $\mathcal{R}_{\mathcal{T},\mathcal{M}}(\mathcal{V}(\alpha))$ over $\mathcal{F}(F)$ is false.

From the correctness of the above algorithm, we can then derive the following result.

**Theorem 5** *Checking whether $Mod_F(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) \neq \emptyset$ can be done in PTIME with respect to $\mathcal{T}$ and $F$, and in NP with respect to $\mathcal{M}$.*

## 5 Techniques for protection and faithfulness

In this section we present techniques for the two problems introduced in Section 3, namely protection and faithfulness. We start by introducing some preliminary notions that we use in the technical development.

The first notion concerns with the idea of expressing violations of constraints in $\mathcal{C}_\mathcal{S}$, as well as negative axioms in $\mathcal{T}$, by means of a suitable conjunctive query, called violation query. To this aim, we extend the function $\mathcal{V}$ defined in the previous section as follows.

**Definition 9** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \mathcal{C}_\mathcal{S} \rangle \rangle$ be an OBDA specification. Then $\mathcal{V}(\beta)$ is defined as follows:*

- *if $\beta \in \mathcal{C}_\mathcal{S}$ is a* key dependency *of the form $key(R) = \{i_1, ..., i_n\}$, then $\mathcal{V}(\beta)$ is the query constituted by two atoms $R(v_1, \ldots, v_k), R(w_1, \ldots, w_k)$, where $v_j = w_j$ if $j \in \{i_1, ..., i_n\}$, and $v_j \neq w_j$) otherwise.*
- *if $\beta \in \mathcal{C}_\mathcal{S}$ is an* inclusion dependency *of the form $r[i, ..., j] \subseteq s[n, ..., t]$ and $r/k \in \Sigma_\mathcal{S}$ and $s/p \in \Sigma_\mathcal{S}$, then $\mathcal{V}(\beta)$ is the query $\exists \vec{x}.r(x_1, .., x_k)$*
- *if $\beta \in \mathcal{C}_\mathcal{S}$ is a* denial assertion *of the form $\exists \vec{x}.\Phi(\vec{x}) \rightarrow \bot$, then $\mathcal{V}(\beta)$ is the query defined in the previous section, i.e., $\exists \vec{x}.\Phi(\vec{x})$*
- *if $\beta \in \mathcal{T}$ is a* negative inclusion assertion *of the form $A \sqsubseteq \neg B$ (resp., $R \sqsubseteq \neg Q$), then $\mathcal{V}(\beta) = A(x) \wedge B(x)$ (resp., $R(x, y) \wedge Q(x, y)$).*

We call *trivial* an inclusion dependency of the form $r[A] \subseteq r[A]$, i.e. whenever the inclusion is trivially satisfied. We remind the reader that, given an incomplete database $F$, $\mathcal{F}(F)$ denotes the complete database obtained from $F$ by substituting the variables in $F$ with constants. The next lemma shows that the incomplete database $\mathcal{F}(\mathcal{V}(\beta))$ really captures the violations to $\beta$.

**Lemma 1** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \mathcal{C}_\mathcal{S} \rangle \rangle$ be an OBDA specification, $\beta_1 \in \mathcal{C}_\mathcal{S}$, $\beta_2 \in \mathcal{T}^-$, let $D$ be a $\Sigma_\mathcal{S}$-database, and $I$ an interpretation for $\beta_2$. If $D \not\models \beta_1$, then $D$ is an instance of $\mathcal{F}(\mathcal{V}(\beta))$, and if $I \not\models \beta_2$, then $I \models \mathcal{V}(\beta_2)$.*

We start our analysis of *protection* by noticing that, being $\mathcal{T}$ a *DL-Lite*$_\mathcal{R}$ ontology, we can reformulate, without loss of generality, protection into a single axiom basis. We now define the notion of $\mathcal{S}$ protecting $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ from $\alpha$-inconsistencies, where $\alpha$ is a single negative inclusion in the ontology, and show that in order to capture protection of the whole $\mathcal{S}$, we can rely on this axiom-by-axiom definition.

**Definition 10** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDA specification, and $\alpha$ a negative inclusion assertion in $\mathcal{T}^-$. Then $\mathcal{S}$ protects $\mathcal{B}$ from $\alpha$-inconsistencies if for all $\Sigma_\mathcal{S}$-database $D$, if $Mod_D(\langle \mathcal{T}^+ \cup \{\alpha\}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) = \emptyset$, then $D \not\models \mathcal{C}_\mathcal{S}$.*

**Lemma 2** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDA specification. Then $\mathcal{S}$ protects $\mathcal{B}$ if and only if for all $\alpha \in \mathcal{T}^-$, we have that $\mathcal{S}$ protects $\mathcal{B}$ from $\alpha$-inconsistencies.*

Lemma 2 suggests that, for checking whether $\mathcal{S}$ does not protect $\mathcal{B}$, we can just look for a counterexample, i.e., a $\Sigma_\mathcal{S}$-database $D$ and an $\alpha \in \mathcal{T}^-$ such that $Mod_D(\langle \mathcal{T}^+ \cup \{\alpha\}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) = \emptyset$, and $D \models \mathcal{C}_\mathcal{S}$. Being $\mathcal{T}$ a *DL-Lite*$_\mathcal{R}$ ontology, this means that we can restrict our attention to those $\mathcal{D}$ such that $\mathcal{D} \models \mathcal{R}_{\mathcal{T},\mathcal{M}}(\mathcal{V}(\alpha))$. By exploiting this property, the next theorem shows that, in searching for a counterexample, we need to take into account just a finite number of $\Sigma_\mathcal{S}$-databases.

**Theorem 6** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDA specification, and $\alpha$ a negative inclusion assertion in $\mathcal{T}$. Then $\mathcal{S}$ does not protect $\mathcal{B}$ from $\alpha$-inconsistencies if and only if for at least one $q \in \mathcal{R}_{\mathcal{T},\mathcal{M}}(\mathcal{V}(\alpha)))$, we have that $\mathcal{F}(q)$ is consistent with $\mathcal{S}$.*

*Proof.* (Sketch) *If-part* Let $q \in \mathcal{R}_{\mathcal{T},\mathcal{M}}(\mathcal{V}(\alpha)))$ be such that $\mathcal{F}(q)$ is consistent with $\mathcal{S}$. Because of the constraints in $\mathcal{S}$, we can assume that there exists $D$, a finite model for $F(q) \cup \mathcal{S}$. Clearly, $D$ is a model for the whole $\mathcal{R}_{\mathcal{T},\mathcal{M}}(\mathcal{V}(\alpha))$. Since $\mathcal{T}$ is a *DL-Lite*$_\mathcal{R}$ ontology, $Mod_D(\langle \mathcal{T}^+ \cup \{\alpha\}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) = \emptyset$, and $D$ is legal with respect to $\mathcal{C}_\mathcal{S}$, we conclude that $\mathcal{S}$ does not protect $\mathcal{B}$ from $\alpha$-inconsistencies.

*Only-if-part* Assume $\mathcal{S}$ does not protect $\mathcal{B}$ from $\alpha$-inconsistencies. Then for at least one database $D$ we have that $Mod_D(\langle \mathcal{T}^+ \cup \{\alpha\}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) = \emptyset$ but $D \models \mathcal{C}_\mathcal{S}$. Since $\mathcal{T}$ is an *DL-Lite*$_\mathcal{R}$ TBox, the violation of $\alpha$ can be represented by $\mathcal{R}_{\mathcal{T},\mathcal{M}}(\mathcal{V}(\alpha))$ over $\Sigma_\mathcal{S}$. Let

$q \in \mathcal{R}_{\mathcal{T},\mathcal{M}}(\mathcal{V}(\alpha))$ be one of the conjunctive queries evaluating true over $D$. This means that $D$ is a model for $\mathcal{C}_\mathcal{S} \cup \{q\}$, and therefore, by the definition of $\mathcal{F}$, $D \models \mathcal{F}(q) \cup \mathcal{C}_\mathcal{S}$ too, thus proving that $\mathcal{F}(q) \cup \mathcal{S}$ admits a finite model. □

Note that Theorem 6 directly provides us with an algorithm for checking protection in OBDA specifications. From the correctness of this algorithm we get the following result.

**Theorem 7** *Protection can be solved in PTIME with respect to $\mathcal{T}$ and $\mathcal{M}$, and in NP with respect to $\mathcal{S}$.*

We now turn our attention to *faithfulness*. We start by noticing that, similarly to protection, we can recast the definition of faithfulness into a single axiom basis.

**Definition 11** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \mathcal{C}_\mathcal{S} \rangle \rangle$ be an OBDA specification, and $\beta \in \mathcal{C}_\mathcal{S}$ a constraint of $\mathcal{S}$. Then $\beta$ is faithful to $\mathcal{B}$ if for all $\Sigma_\mathcal{S}$-database $D$, if $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) \neq \emptyset$, then $D \models \beta$.*

**Lemma 3** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \mathcal{S} \rangle$ be an OBDA specification. Then $\mathcal{S}$ is not faithful to $\mathcal{B}$ if and only if for at least one $\beta \in \mathcal{C}_\mathcal{S}$, $\beta$ is not faithful to $\mathcal{B}$.*

As lemma 3 points out, in order to look for counterexamples to faithfulness, we can just focus on databases violating one of the $\beta \in \mathcal{C}_\mathcal{S}$. By means of lemma 1, we can further restrict our attention to $\Sigma_\mathcal{S}$-databases coinciding with the instances of $\mathcal{F}(\mathcal{V}(\beta))$. This is what next theorem proves.

**Theorem 8** *Let $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \mathcal{C}_\mathcal{S} \rangle \rangle$ be an OBDA specification and let $\beta \in \mathcal{C}_\mathcal{S}$ be a non-trivial constraint of $\mathcal{S}$. Then $\beta$ is not faithful to $\mathcal{B}$ if and only if we have that $Mod_{\mathcal{F}(\mathcal{V}(\beta))}(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) \neq \emptyset$.*

*Proof.* (Sketch) *If-part.* Suppose that, for the given $\beta$, $Mod_{\mathcal{F}(V(\beta))}(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) = \emptyset$ holds and $\beta$ is not faithful to $\mathcal{B}$ . Let $D$ be a database such that $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) \neq \emptyset$, and $D \not\models \beta$. Since $D \not\models \beta$, $\beta$ is not trivial, and $D \models \mathcal{V}(\beta)$. By exploiting Lemma 1, we can show that this implies that $D$ is an instance of $\mathcal{F}(\mathcal{V}(\beta))$, and this in turn implies that $Mod_{\mathcal{F}(\mathcal{V}(\beta))}(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) \neq \emptyset$, which is a contradiction.

*Only-if-part.* Suppose that $\mathcal{S}$ is faithful to $\mathcal{B}$ with respect to $\beta$-violations, and $Mod_{\mathcal{F}(\mathcal{V}(\beta))}(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) \neq \emptyset$. By construction we have that, starting from $\mathcal{F}(\mathcal{V}(\beta))$ we can easily build a $\Sigma_\mathcal{S}$-database $D$ that is an instance of $\mathcal{F}(\mathcal{V}(\beta))$ such that $Mod_D(\langle \mathcal{T}, \mathcal{M}, \langle \Sigma_\mathcal{S}, \emptyset \rangle \rangle) \neq \emptyset$, and $D \not\models \{\beta\}$, proving that $\mathcal{S}$ is not faithful to $\mathcal{B}$ with respect to $\beta$-violations, and therefore leading to a contradiction. □

Lemma 3, and theorem 8 directly provides us with an algorithm for checking faithfulness. From the correctness of this algorithm, we can derive the following complexity result.

**Theorem 9** *Faithfulness can be solved in PTIME with respect to $\mathcal{S}$ and $\mathcal{T}$, and in NP with respect to $\mathcal{M}$.*

## 6 Conclusions and future work

We have presented a first study on the relationship between global consistency and local consistency in OBDA. We have proposed a formal framework for these two notions, and illustrated several results for a significant class of OBDA systems.

We plan to continue our investigation along different directions. For instance, we aim at considering the case of OBDA systems where the source schema contains constraints that do not fall into the class of constraints studied here, or where the DLs used for expressing the ontology goes beyond *DL-Lite*$_\mathcal{R}$. Also, another issue that we just started exploring is to check, given an ontology and a mapping to an alphabet $\Sigma_\mathcal{S}$, whether there exists a set of constraints in a given class that makes the schema both faithful and protecting with respect to the OBDA specification. This would be a great tool to use in a scenario where we have to design the database of the OBDA system in such a way that specific formal quality criteria are satisfied.

**Acknowledgements:** Work partially supported by the EU under FP7, project Optique (Scalable End-user Access to Big Data), grant n. FP7-318338.

## REFERENCES

[1] Serge Abiteboul, Richard Hull, and Victor Vianu, *Foundations of Databases*, Addison Wesley Publ. Co., 1995.
[2] Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev, 'The *DL-Lite* family and relations', *J. of Artificial Intelligence Research*, **36**, 1–69, (2009).
[3] *The Description Logic Handbook: Theory, Implementation, and Applications*, eds., Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, Cambridge University Press, 2010. Paperback edition.
[4] Meghyn Bienvenu, Balder ten Cate, Carsten Lutz, and Frank Wolter, 'Ontology-based data access: A study through disjunctive datalog, CSP, and MMSNP', in *Proc. of PODS 2013*. ACM Press, (2013).
[5] Andrea Calì, Georg Gottlob, and Andreas Pieris, 'New expressive languages for ontological query answering', in *Proc. of AAAI 2011*, pp. 1541–1546, (2011).
[6] Andrea Calì, Domenico Lembo, and Riccardo Rosati, 'Query rewriting and answering under constraints in data integration systems', in *IJCAI*, pp. 16–21, (2003).
[7] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Mariano Rodriguez-Muro, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo, 'The Mastro system for ontology-based data access', *Semantic Web J.*, **2**(1), 43–53, (2011).
[8] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati, '*DL-Lite*: Tractable description logics for ontologies', in *Proc. of AAAI 2005*, pp. 602–607, (2005).
[9] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati, 'Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family', *J. of Automated Reasoning*, **39**(3), 385–429, (2007).
[10] Marco Console and Maurizio Lenzerini, 'Data quality in ontology-based data access: The case of consistency', in *Proc. of AAAI 2014*.
[11] Marc Friedman, Alon Levy, and Todd Millstein, 'Navigational plans for data integration', in *Proc. of AAAI'99*, pp. 67–73. AAAI Press, (1999).
[12] Alon Y. Halevy, 'Answering queries using views: A survey', *VLDB Journal*, **10**(4), 270–294, (2001).
[13] Tomasz Imielinski and Witold Lipski Jr., 'Incomplete information in relational databases', *J. of the ACM*, **31**(4), 761–791, (1984).
[14] Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, and Michael Zakharyaschev, 'The combined approach to ontology-based data access', in *Proc. of IJCAI 2011*, pp. 2656–2661, (2011).
[15] Maurizio Lenzerini, 'Data integration: A theoretical perspective.', in *Proc. of PODS 2002*, pp. 233–246, (2002).
[16] Maurizio Lenzerini, 'Ontology-based data management', in *Proc. of CIKM 2011*, pp. 5–6, (2011).
[17] Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati, 'Linking data to ontologies', *J. on Data Semantics*, **X**, 133–173, (2008).
[18] Mariano Rodriguez-Muro and Diego Calvanese, 'High performance query answering over *DL-Lite* ontologies', in *Proc. of KR 2012*, pp. 308–318, (2012).
[19] Mariano Rodriguez-Muro and Diego Calvanese, 'Quest, an OWL 2 QL reasoner for ontology-based data access', in *Proc. of OWLED 2012*, volume 849 of *CEUR*, `ceur-ws.org`, (2012).
[20] Riccardo Rosati, 'On the decidability and finite controllability of query processing in databases with incomplete information', in *Proc. of PODS 2006*, pp. 356–365, (2006).