Epistemic Integrity Constraints for Ontology-Based Data Management

Citation for published version:
Console, M & Lenzerini, M 2020, Epistemic Integrity Constraints for Ontology-Based Data Management. in *Proceedings of the Thirty-fourth AAAI Conference on Artificial Intelligence.* AAAI Press, pp. 2790-2797, 34th AAAI Conference on Artificial Intelligence, New York, United States, 7/02/20. https://doi.org/10.1609/aaai.v34i03.5667

Digital Object Identifier (DOI):
10.1609/aaai.v34i03.5667

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
Proceedings of the Thirty-fourth AAAI Conference on Artificial Intelligence

Epistemic Integrity Constraints for Ontology-Based Data Management

Marco Console[1] and **Maurizio Lenzerini**[2]
[1]University of Edinburgh
[2]Sapienza, University of Rome

Abstract

Ontology-based data management (OBDM) is a powerful knowledge-oriented paradigm for managing data spread over multiple heterogeneous sources. In OBDM, the data sources of an information system are handled through the reconciled view provided by an ontology, i.e., the conceptualization of the underlying domain of interest expressed in some formal language. In any information systems where the basic knowledge resides in data sources, it is of paramount importance to specify the acceptable states of such information. Usually, this is done via integrity constraints, i.e., requirements that the data must satisfy formally expressed in some specific language. However, while the semantics of integrity constraints are clear in the context of databases, the presence of inferred information, typical of OBDM systems, considerably complicates the matter. In this paper, we establish a novel framework for integrity constraints in the OBDM scenarios, based on the notion of knowledge state of the information system. For integrity constraints in this framework, we define a language based on epistemic logic, and study decidability and complexity of both checking satisfaction and performing different forms of static analysis on them.

1 Introduction

Managing information spread over multiple heterogeneous data sources is a long-standing topic in the fields of data management and knowledge representation. A powerful tool to address this issue is the Ontology-based Data Management (OBDM) paradigm (**?**), where a conceptual specification of the domain of interest, called ontology, is superimposed over a set of pre-existing data sources using suitable mapping assertions (**?**). The resulting information systems, called Ontology-Based Data Management systems, enable users to interact with the data using the vocabulary of the ontology, thus giving a high-level view of the information contained in the data sources, and enhancing query answering (**?**; **?**; **?**) and other data management (**?**) tasks. In this paper, we focus on ontologies expressed in Description Logics (DL) (**?**). DLs are logical formalisms that represent the domain of interest in terms of *concepts*, i.e., classes of objects, and *roles*, i.e., binary relations between objects. Intuitively, the models of the logical theory constituted by the ontology, the mapping, and the data, i.e., the models of the OBDM system, represent all the possible worlds that are consistent with both the conceptual specification and the data. A simple example of OBDM system follows.

Example 1 *Consider a set $\mathfrak{D}$ of data sources, storing HR data from different branches of a company. Branch A stores information about employees and their department in table* $\mathsf{empA}(Name, Code, Department)$. *The central HR office stores information about departments in table* $\mathsf{dep}(Code, Manager)$ *and employee codes in table* $\mathsf{pers}(Name, Code)$. *The following database, denoted by D, represents a possible instance of $\mathfrak{D}$.*

empA

N	*C*	*D*
Bob	*1B*	*D1*

dep

C	*M*
D1	*Tim*
D0	*Jim*

pers

N	*C*
Bob	*1B*
Jim	*2J*
Wim	*3W*

We assume that the domain of interest can be formally described by a DL ontology $\mathcal{T}$ with concepts Emp, for employees, and Dep, for departments, and roles $hasDep$, associating employees to their department, $hasCode$, associating employees to their individual code, and $hasMan$, associating departments to their manager. In this domain, every employee works for at least one department, as formalized by the following axioms in $\mathcal{T}$.

$$Emp \sqsubseteq \exists hasDep \qquad \exists hasDep^{-} \sqsubseteq Dep$$

The set $\mathcal{M}$ of mapping assertions (Figure 1) describes the relationship between the data in $\mathfrak{D}$ and the concepts and roles of $\mathcal{T}$. In the remainder of this section, we will denote by $\mathcal{O}$ the OBDM system constituted by D, $\mathcal{T}$, and $\mathcal{M}$.

In the context of data management, a question that arises naturally is whether the stored data conform with our understanding of the domain. During the years, this fundamental question gave rise to the concept of *integrity constraints* (**?**; **?**), ICs for short. Informally, ICs are a means to describe the acceptable states of a knowledge base, and are very popular in relational databases where, together with suitable mechanisms to enforce satisfaction, they are used

$$\forall x, y, z. empA(x, y, z) \rightarrow \begin{cases} Emp(x) \wedge \\ hasCode(x, y) \wedge \\ hasDep(x, z); \end{cases}$$

$$\forall x, y. dep(x, y) \rightarrow \begin{cases} Dep(x) \wedge \\ hasMan(x, z) \wedge \\ hasDep(z, x) \wedge \\ Emp(z) \end{cases}$$

$$\forall x, y, z. pers(x, y) \rightarrow \begin{cases} Emp(x) \wedge \\ hasCode(x, y) \end{cases}$$

Figure 1: Mapping $\mathcal{M}$ for Example 1.

to prevent meaningless states of the data. In this context, there is a general agreement on languages and semantics suitable for ICs, typically based on *data dependencies*, i.e., special first-order formulae to be evaluated as queries over the database.

Example 2 *Refer to Example 1, and assume that* dep *is the reference table for departments, meaning that every department must appear in* dep*. The following data dependency, also called foreign key constraint, captures this requirement:*

$$\forall x, y, z.\ empA(x, y, z) \rightarrow \exists w. dep(z, w)$$

ICs have received attention also in the fields of DL knowledge bases and ontological reasoning (**?**; **?**) and OBDM (**?**), where, analogously to the case of databases, their goal is to define the acceptable states of an information system. However, databases are often based on the closed-world semantics. Therefore, checking satisfaction of constraints in a database D amounts to verifying whether the corresponding formaulae are true in the logical interpretation represented by D. In contrast, OBDM systems support both modelling of incomplete information and inferences on the data. This means that the semantics of an OBDM system comprise a set of possible logical models extending the world represented by the stored data. For this reason, defining the semantics of integrity constraints is much more challenging. Indeed, any mechanism that is able to validate the ICs in this context requires a way to control the subtle interplay between data, ontology axioms, mapping, and constraints. Unfortunately, controlling the way these elements interact is a task far from being straightforward, and the possible solutions are many. The consequence is that, after years of efforts, there is still no general agreement on what formalism for ICs should be used in ontological reasoning and OBDM. Despite the lack of a generally accepted formalism, however, the need of defining the acceptable states of an OBDM system is very natural as the following example shows.

Example 3 *After a detailed analysis, the scenario in Example 1 reveals that any acceptable state of the OBDM system $\mathcal{O}$ should conform to the following requirements:*

1. *Every employee has at most one code.*
2. *Every employee works in at least one department.*
3. *Every department has at least one manager.*

Requirements in Example 3 should not be confused with general statements about the way the world behaves, i.e., these are not ontological axioms. Consider, e.g., Requirement 1. In the real world, an employee may have multiple codes, maybe relative to the different roles she covers. To allow this possibility, the ontology of $\mathcal{O}$ has no axioms entailing that codes are unique for an employee. What we require with Requirement 1 is that, in the current state of the system, no employee has more than one code. Statements of this kind require some form of introspection, and hence they cannot be expressed using ontological axioms. To express these requirements, we need to use ICs. These observations bring us to the following conclusions: while ontological axioms shape the information contained in OBDM systems *a-priori*, ICs should act *a-posteriori* on the system, aiming at validating its current state.

Depending on how its current state is defined, there are different techniques to check and validate the information an OBDM system contains. Two methods proposed during the years are the Entailment Semantics (ES), and the Minimal Herbrand Model Semantics (MHMS).

In the *Entailment Semantics* (**?**), each IC is treated simply as a boolean query, with the idea that it should be evaluated to true in every model of the system. This approach suffers from two main drawbacks. First, checking satisfaction of ICs under ES can very quickly become undecidable, depending on the form of ICs. Second, ES is too strict, and its behavior can become counterintuitive at times. For instance, by referring to Example 1, observe that, since the ontological axioms do not entail that codes are unique for the various employees, there exist models of $\mathcal{O}$ in which the same employee has more than one code. In turn, this proves that, under ES, $\mathcal{O}$ violates Requirement 1 mentioned in Example 3. However, if we inspect the data sources, there is no evidence whatsoever that the requirement is violated, because for no employee in the system we have more than one code. Something analogous happens with Requirement 3. The case of Requirement 2, however, is different. Such requirement is satisfied under ES due to the ontological axioms, although we do not have any evidence in the data sources about the department where Wim works.

Intuitively, the above observations tell us that, under ES, the current state of the data plays a marginal role. On the one hand, violations that are not directly supported by the data in the sources may blame the ontology of the OBDM system, as in the case of Requirement 1 and 3. On the other hand, constraints may be validated by the ontology, although no data coming from the sources can be used for this purpose, as in the case of Requirement 2.

A possible approach to get closer to the information contained in the sources is to use some notion of minimal information. This is the idea of the *Minimal Herbrand Models Semantics*. Intuitively, the unique minimal Herbrand model $\mathcal{H}$ of an OBDM system $\mathcal{O}$ is an interpretation of $\mathcal{O}$ with the following property: for every existentially quantified conjunction of atomic formulae ϕ, $\mathcal{O}$ entails ϕ

if and only if ϕ is true in $\mathcal{H}$. Under MHMS, an OBDM system satisfies an IC if and only if the minimal Herbrand model of the system does. This notion was first advocated for ontological reasoning in (?), and it was recently extended to OBDM systems in (?). Interestingly, checking satisfaction of constraints under MHMS is decidable in many relevant cases. For example, in (?), the authors establish a decidability result for a very expressive ontological language ($\mathcal{ALCHI}$) and constraints expressed by arbitrary first-order formulae. Unfortunately, the proof of decidability relies on algorithms for MSO, and thus the upper-bound provided is only non-elementary. In (?), the authors focus on more restricted languages and present upper-bounds in Π_2^p. Under several restrictions on the languages used, the authors establish also the decidability of relevant static analysis tasks.

However, as the definition of MHMS heavily relies on the minimal Herbrand Model, the violation or satisfaction of an IC under MHMS is still only loosely related to the content of the data sources. In our running example, it is easy to see that the minimal Herbrand model $\mathfrak{I}$ of $\mathcal{O}$ validates Requirements 1 and 2. However, observe that, although $\mathfrak{I}$ satisfies $\exists x.hasDep(Wim, x)$, $\mathfrak{I}$ does not satisfy $\exists x, y.hasDep(Wim, x) \wedge hasMan(x, y)$. Therefore, $\mathcal{O}$ does not satisfy Requirement 3 under MHMS.

If the goal is to manage a set of data sources using an ontology, we argue that the violation or the satisfaction of an IC should be solidly grounded on the knowledge that the data in the sources allow the system to achieve. For this reason, we think that it is worth exploring a new semantics of ICs for OBDM systems, epistemic in nature. We call such semantics *Knowledge Semantics* (KS), because under this semantics, ICs are validated by what the system is sure about, i.e., by what the system knows, rather than by the possible worlds described by the system, or by the minimal information contained in such worlds. In our current example, the employees known by $\mathcal{O}$ are Tim, Jim, Wim and Bob. For each of them, $\mathcal{O}$ knows exactly one code and therefore, under KS, Requirement 1 is satisfied. In other words, KS reads Requirement 1 as follows: for each known employee, the system knows exactly one code. Similarly, Requirement 3 is satisfied under KS, since the departments known by $\mathcal{O}$ are $D1$ and $D0$, and the system knows the manager of each of them. With regard to Requirement 2, we consider two possible readings of the condition. The first reading is the one imposing that all known employees work in a department. If we consider such reading, it is easy to see that the system satisfies the condition under KS. In the second reading the condition states that for each known employee, the department she works for is known. In this case, $\mathcal{O}$ does not satisfy Requirement 2 under KS, because, for employee Wim, we know that a department exists but we do not know its exact identity.

The epistemic approach to ICs was first proposed by Reiter in the context of incomplete databases (?; ?) and then applied to ontological reasoning in (?). Also, in (?) the authors propose query languages with an epistemic operator and suggest the use of these queries as integrity constraints for OBDM systems. In this paper, we investigate this epistemic approach to ICs further, we give a formal definition of what an OBDM system knows about the real world, and we develop a new formalism for ICs based on this notion. The epistemic nature of this language allows us to express ICs that cannot be expressed in other formalisms and distinguish between different readings of the integrity constraints, such as the ones mentioned in the example above. For ICs expressed in this formalism, we study the computational complexity of both checking satisfaction and performing different tasks of static analysis on ICs

Before concluding this section, we want to stress out that our attempt is not to define the single right way to express integrity constraints in OBDM systems. Defining the acceptable states of information systems is a widespread necessity and the possible different scenarios are many. However, we observe that, if satisfied, ICs under KS do not alter the semantics of OBDM systems. For this reason, we believe that our formalism can be used either as it is or as a powerful complement to other forms of constraints.

The remainder of this paper is organized as follows. Section 2 contains preliminary definitions used throughout the paper. Section 3 introduces our framework for ICs in OBDM systems. Section 4 presents computational complexity results relative to this framework. Section 5 concludes the paper.

2 Preliminary Definitions

In this section, we briefly review the concepts used in the technical development of this paper. In what follows, we assume basic familiarity with the standard notions of computational complexity and first-order logic, and refer the reader to (?) for a detailed account.

Databases. We assume relational databases over a countably infinite set of constants Δ and refer to (?) for a detailed account on the topic. A database schema $\mathcal{S}$ is a pair $\langle \Sigma_{\mathcal{S}}, \mathcal{C}_{\mathcal{S}} \rangle$ where $\Sigma_{\mathcal{S}}$ is a relational signature, and $\mathcal{C}_{\mathcal{S}}$ is a set of integrity constraints. A $\Sigma_{\mathcal{S}}$-database D is a relational structure in the signature $\Sigma_{\mathcal{S}}$; D is also consistent with $\mathcal{S}$ (alternatively $\mathcal{S}$-database) if it satisfies all the constraints in $\mathcal{C}_{\mathcal{S}}$, written $D \models \mathcal{C}_{\mathcal{S}}$. In general, integrity constraints take the form of sentences in some well-behaved fragment of first-order logic (FOL). A popular such fragment is Data Dependencies, i.e., formulae $\forall \bar{x} \phi(\bar{x}) \rightarrow \exists \bar{y} \psi(\bar{x}, \bar{y})$, where ϕ and ψ are conjunctions of relational and equality atoms. In this paper, we focus on the following widely accepted classes of Data Dependencies: tuple-generating dependencies (tgds), equality-generating dependencies (egds), and denial constraints (dens). In tgds ψ is a conjunction of only positive relational atoms; in egds ψ is a conjunction of equality atoms; in dens ψ is the symbol False ($\perp$). Unfortunately, the unrestricted interaction between tgds, egds, and dens leads easily to the undecidability of many fundamental decision problems. To ensure decidability, sets of tgds, egds, and dens are usually subject to some syntactic requirement. A widely-accepted requirement, common in data exchange and integration as well as in data management, is weak-acyclicity (?). For sets of weak-acyclic tgds, egds, and dens, many reasoning tasks are decidable due to the termination of the *chase* algorithm(**?**).

Besides checking the satisfaction of integrity constraints, a fundamental task to perform with databases is query answering. Given a database D and a query $q(\bar{x})$, i.e., a first-order formula with free variables $\bar{x}$, query answering asks for the set $ans(q, D)$ of tuples of constants $\bar{c}$ such that D satisfies the formula $q(\bar{c})$. Important classes of queries are Conjunctive Queries (CQs), i.e., existentially quantified conjunctions of relational atoms, and Union of Conjunctive Queries, i.e., disjunctions of CQs sharing the same tuple of free variables. Computing $ans(q, D)$ for $q \in UCQ$ can be done in NP, and in AC^0 if we fix the query q. When dealing with queries, a fundamental question is whether a query q is contained in a query q' under a database schema $\mathcal{S}$, i.e., whether $ans(q, D)$ is contained in $ans(q', D)$ for every $\mathcal{S}$-database D. In the literature, this problem is called Query Containment Under Constraints. If $\mathcal{S}$ contains only weakly-acyclic tgds, egds, and dens, checking whether an UCQ q is contained in an UCQ q' under $\mathcal{S}$ is $2EXPTIME$-complete (**?**).

Description Logics Ontologies. An ontology is a conceptualization of a domain of interest expressed in terms of a formal language. Hereafter, we assume ontologies expressed in Description Logics (DLs). A DL knowledge base 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 logical assertions (called ABox assertions) specifying the instances of concepts and roles. In this paper, we focus on ontologies expressed in $DL\text{-}Lite_{\mathcal{A}}$, a member of the $DL\text{-}Lite$ family of DLs. In what follows, we give only a brief account on $DL\text{-}Lite_{\mathcal{A}}$ and refer the reader to (**?**) for a thorough introduction. The syntax of concept, roles and attributes *expressions* in $DL\text{-}Lite_{\mathcal{A}}$ is specified by 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 $DL\text{-}Lite_{\mathcal{A}}$ TBox $\mathcal{T}$ is constituted by axioms of the form $B \sqsubseteq C$, $Q \sqsubseteq R$, $U \sqsubseteq V$, $E \sqsubseteq F$, called inclusion assertions, and axioms of the form (funct Q), (funct U), called functionality assertions. In $DL\text{-}Lite_{\mathcal{A}}$ TBoxes, we further require that roles and attributes occurring in functionality assertions cannot be specialized, i.e., they cannot occur in the right-hand side of positive inclusions.

Given a first-order interpretation $\mathcal{I}$, we define $\{P^-\}^{\mathcal{I}} = \{\langle b, a \rangle \mid \langle a, b \rangle \in P^{\mathcal{I}}\}$; and $\neg B^{\mathcal{I}}$, resp. $\neg Q^{\mathcal{I}}$ and $\neg U^{\mathcal{I}}$, as the complement of $B^{\mathcal{I}}$, resp. $Q^{\mathcal{I}}$ and $U^{\mathcal{I}}$. A first-order interpretation $\mathcal{I}$ satisfies an ABox assertion $A(a)$, respectively $R(a, b)$, if $a^{\mathcal{I}} \in A^{\mathcal{I}}$, resp., $\langle a^{\mathcal{I}}, b^{\mathcal{I}} \rangle \in A^{\mathcal{I}}$. $\mathcal{I}$ satisfies $B \sqsubseteq C$, (respectively $Q \sqsubseteq R$, $U \sqsubseteq V$, $E \sqsubseteq F$), if $B^{\mathcal{I}} \subseteq A^{\mathcal{I}}$ (resp., $Q^{\mathcal{I}} \subseteq R^{\mathcal{I}}$, $U^{\mathcal{I}} \subseteq V^{\mathcal{I}}$, $E^{\mathcal{I}} \subseteq F^{\mathcal{I}}$). We say that a first-order interpretation $\mathcal{I}$ is a model of $\langle \mathcal{T}, \mathcal{A} \rangle$ if $\mathcal{I}$ satisfies the axioms in $\mathcal{T}$ and the assertions in $\mathcal{A}$ and denote the set of models of $\langle \mathcal{T}, \mathcal{A} \rangle$ by $Mod(\langle \mathcal{T}, \mathcal{A} \rangle)$.

Ontology-Based Data Management. An OBDM specification $\mathcal{B}$ is a triple $\langle \mathcal{T}, \mathcal{M}, \Sigma \rangle$ where $\mathcal{T}$ is a description logic ontology, Σ is a relational signature, and $\mathcal{M}$ is a set of mapping assertions from Σ to $\mathcal{T}$, i.e., FOL sentences of the form $\forall \bar{x}.\phi(\bar{x}) \rightarrow \exists \bar{y}.\psi(\bar{x}, \bar{y})$, where ϕ and ψ are first-order formulae on Σ and on the alphabet of $\mathcal{T}$, respectively. In this paper, we focus on two classes of OBDM specifications: *lightweight* and *trivial*. In a lightweight OBDM specification, $\mathcal{T}$ is a $DL\text{-}Lite_{\mathcal{A}}$ TBox and $\mathcal{M}$ is a set of conjunctive GAV mapping assertions, i.e., formulae of the form $\forall \bar{x}.\phi(\bar{x}) \rightarrow \psi(\bar{x})$, where both ϕ and ψ are conjunctions of atoms (**?**). In trivial OBDM specifications, $\mathcal{T}$ is an ontology on alphabet Σ and empty set of axioms, and $\mathcal{M}$ is the identity mapping. Clearly, trivial specifications are also lightweight.

Given an OBDM specification $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \Sigma \rangle$ and a Σ-database D, the pair $\mathcal{O} = \langle \mathcal{B}, D \rangle$ is called *OBDM system*. Semantics of $\mathcal{O}$ are defined as follows: a first-order interpretation $\mathcal{I}$ is a model for the OBDM system $\mathcal{O}$ if $\mathcal{I}$ is a model for $\mathcal{T}$, and the pair $\langle D, \mathcal{I} \rangle$ satisfies the mapping $\mathcal{M}$. As customary, constants from D are assumed to satisfy the Unique Name Assumption, i.e., each constant is equal only to itself. The set of models of an OBDM system $\mathcal{O}$ will be denoted by $Mod(\mathcal{O})$; if $Mod(\mathcal{O}) \neq \emptyset$ we will say that $\mathcal{O}$ is satisfiable, unsatisfiable otherwise. If $\mathcal{B}$ is lightweight, we call $\mathcal{O} = \langle \mathcal{B}, D \rangle$ lightweight OBDM system. For every lightweight OBDM system $\mathcal{O}$, there exists a first-order sentence $\pi(\mathcal{O})$ such that $\mathcal{I} \in Mod(\mathcal{O})$ if and only if $\mathcal{I}$ is a model for $\pi(\mathcal{O})$ (**?**).

In the context of OBDM systems, two fundamental reasoning tasks are Satisfiability and Query Answering. *Satisfiability* is the problem of checking whether $Mod(\mathcal{O}) \neq \emptyset$, for input OBDM system $\mathcal{O}$. If $\mathcal{B}$ is lightweight, there exists a UCQ with inequalities $v_{\mathcal{B}}$, called violation query for $\mathcal{B}$, such that $\mathcal{O}$ is satisfiable if and only if $ans(v_{\mathcal{B}}, D) = \emptyset$. The query $v_{\mathcal{B}}$ can be computed in time polynomial w.r.t. $\mathcal{B}$, so the problem of checking whether $\mathcal{O}$ is satisfiable is in NP, and in AC^0 if $\mathcal{B}$ is fixed. Given an OBDM system $\mathcal{O} = \langle \mathcal{B}, D \rangle$ and a conjunctive query $q(\bar{x})$, *Query Answering* asks for the set $cert(q, \mathcal{O})$ of tuples of constants $\bar{c}$ such that the sentence $q(\bar{c})$ is true in every model of $\mathcal{O}$. If $\mathcal{B}$ is lightweight, for every $q \in UCQ$ there exists a UCQ $Rew(q, \mathcal{B})$ such that $\bar{c} \in cert(q, \mathcal{O})$ if and only if $\bar{c} \in ans(Rew(q, \mathcal{B}), D)$. In general, Query Answering is NP-Complete, and it can be solved in AC^0 if we fix the query and the specification.

Epistemic Logics. In what follows, we use the logic $\mathcal{OL}$ defined in (**?**). Formulae in $\mathcal{OL}$ are defined as follows: atomic formulae and the symbol $\perp$ are in $\mathcal{OL}$, if ϕ and ψ are in $\mathcal{OL}$ so are $\phi \wedge \psi$, $\phi \vee \psi$, $\neg\phi$, $\exists x.\phi(x)$, $\forall x.\phi(x)$, $\mathbf{K}(\phi)$, and $\mathbf{O}(\phi)$. A formula $\phi \in \mathcal{OL}$ is *objective* if no occurrence of $\mathbf{K}$ and $\mathbf{O}$ appears in ϕ. To interpret formulae in $\mathcal{OL}$, we use epistemic interpretations over a domain Δ. An epistemic interpretation is a pair $\langle W, w \rangle$ where W is a set of first-order interpretations with domain Δ and w is an interpretation in

W. Let ϕ be a formula in $\mathcal{OL}$ and let $\mathfrak{E} = \langle W, w\rangle$ be an epistemic interpretation, $\mathfrak{E}$ satisfies ϕ, written $\mathfrak{E} \models \phi$, if the following conditions hold.

- ϕ is an atomic formula and $w \models \phi$.
- $\phi = \neg\phi'$ and $\mathfrak{E} \not\models \phi'$
- $\phi = \phi' \wedge \phi''$ and both $\mathfrak{E} \models \phi'$ and $\mathfrak{E}, w \models \phi''$ hold.
- $\phi = \exists x.\phi'(x)$ and $\mathfrak{E} \models \phi'(c)$, for some $c \in (\Delta)$.
- $\phi = \mathbf{K}(\phi')$, and $w' \in W \Longrightarrow \langle W, w'\rangle \models \phi'$.
- $\phi = \mathbf{O}(\phi')$, and $w' \in W \Longleftrightarrow \langle W, w'\rangle \models \phi'$.

Semantics of $\bot$, $\vee$, $\rightarrow$, and $\forall$ are assumed as customary. Moreover, to ease the complexity of the proofs, we assume that the domain Δ is the set of constants used for databases.

3 Knowledge States and Integrity Constraints

As we argued in the introduction, a promising approach to ICs in the OBDM paradigm relates the acceptable states of an information system to its current knowledge state. To formalize this intuition, we first need to formally define the state of knowledge of an OBDM system. In our framework, this is done by means of the logic $\mathcal{OL}$.

Definition 1 *An epistemic interpretation $\mathfrak{E}$ is an epistemic state of an OBDM system $\mathcal{O}$ if $\mathfrak{E} \models \mathbf{O}(\pi(\mathcal{O}))$.*

The set of epistemic states of an OBDM system $\mathcal{O}$ will be denoted by $\mathcal{E}(\mathcal{O})$. Intuitively, in every epistemic state $\langle W, w\rangle$ of $\mathcal{O}$, the set W represents what $\mathcal{O}$ knows about the world while w represents what $\mathcal{O}$ believes to be true. To define our formalism for integrity constraints we focus on the former and observe that, while the epistemic states of $\mathcal{O}$ may be infinitely many, the knowledge of $\mathcal{O}$ is uniquely defined, as the following proposition shows.

Proposition 1 *If $\mathcal{O}$ is a lightweight OBDM system, every pair $\langle W, w\rangle, \langle W', w'\rangle \in \mathcal{E}(\mathcal{O})$ is such that $W = W'$.*

Proposition 1 is not the only interesting characteristic of knowledge states and, in fact, they capture many of the intuitions discussed in the introduction. To prove this statement, we need to establish a proper framework for ICs in OBDM systems. To this end, we start by augmenting OBDM specifications by a set of $\mathcal{OL}$ formulae that will be interpreted as integrity constraints.

Definition 2 *An OBDM specification with constraints $\mathcal{B}_{\mathcal{C}}$ is a tuple $\langle \mathcal{T}, \mathcal{M}, \Sigma, \mathcal{C}\rangle$ where $\mathcal{B} = \langle \mathcal{T}, \mathcal{M}, \Sigma\rangle$ is an OBDM specification and $\mathcal{C}$ is a set of $\mathcal{OL}$ sentences.*

In what follows, given an OBDM specification with constraints $\mathcal{B}_{\mathcal{C}}$, we denote by $\mathcal{B}$ the underlying OBDM specification (without constraints). Similarly to the standard case, OBDM systems with constraints are pairs $\langle \mathcal{B}_{\mathcal{C}}, D\rangle$ where D is a database and $\mathcal{B}_{\mathcal{C}}$ is an OBDM specification with constraints. In the following definition, we formalize what it means for $\langle \mathcal{B}_{\mathcal{C}}, D\rangle$ to satisfy the set of constraints $\mathcal{C}$.

Definition 3 *An OBDM system with constraints $\langle \mathcal{B}_{\mathcal{C}}, D\rangle$ satisfies $\chi \in \mathcal{OL}$ if $\mathfrak{E} \models \chi$, for every $\mathfrak{E} \in \mathcal{E}(\langle \mathcal{B}, D\rangle)$.*

If $\mathcal{O}$ satisfies every $\chi \in \mathcal{C}$, we will say that $\mathcal{O}$ satisfies its constraints. With this notion in place, we are ready to define semantics for OBDM systems with constraints. As informally discussed in the introduction, these semantics are shaped by the ontology and the data and validated by the integrity constraints.

Definition 4 *Let $\mathcal{O} = \langle \mathcal{B}_{\mathcal{C}}, D\rangle$. The set $Mod(\mathcal{O})$ is equal to $Mod(\langle \mathcal{B}, D\rangle)$ if $\mathcal{O}$ satisfies $\mathcal{C}$, $Mod(\mathcal{O}) = \emptyset$ otherwise.*

If $Mod(\mathcal{O}) \neq \emptyset$, we say that $\mathcal{O}$ is satisfiable, unsatisfiable otherwise. Definition 4 ensures that the semantics of satisfiable systems are unaltered by the constraints.

With this notion of semantics in place, we can focus on the definition of a fragment of $\mathcal{OL}$ suited to express ICs. From a computational standpoint, the use of a restricted fragment of $\mathcal{OL}$ is necessary, as the following proposition shows.

Proposition 2 *Let $\mathcal{B}$ be a trivial OBDM specification and let $\mathcal{C}$ be a set of $\mathcal{OL}$ formulae. It is undecidable to check whether a system $\langle \mathcal{B}_{\mathcal{C}}, D\rangle$ is satisfiable.*

Proof 1 (Intuition) *Let $\mathcal{C} = \{\phi\}$, where ϕ is an objective FOL sentence, and let $\mathcal{O} = \langle \mathcal{B}, D\rangle$ be a satisfiable OBDM system. The system with constraints $\langle \mathcal{B}_{\mathcal{C}}, D\rangle$ is satisfiable if and only if $\mathcal{I} \models \phi$ for every $\mathcal{I} \in Mod(\mathcal{O})$. A reduction from Validity of FOL sentences follows straightforwardly.*

Undecidability, however, is not the only reason why only specific fragments of $\mathcal{OL}$ are suitable as ICs, as the use of some formulae may lead to unwanted behaviours. Consider, e.g., Proposition 2. In the informal proof given, we can see how the use of an objective sentence ϕ as integrity constraint for a system $\mathcal{O}$ is equivalent to requiring that ϕ is entailed by $\mathcal{O}$. In turn, this corresponds to assuming for ϕ the Entailment Semantics. With the goal of having a language that is computationally well-behaved and soundly grounded on the intuitions presented in the introduction, we now present Epistemic Dependencies (EDs).

Definition 5 *An epistemic dependency is an $\mathcal{OL}$ formula*

$$\forall \bar{x}.\mathbf{K}(\exists \bar{z}.\bigwedge_i \phi_i(\bar{x}, \bar{z})) \rightarrow \exists \bar{y}.\mathbf{K}(\exists \bar{w}.\psi(\bar{x}, \bar{y}, \bar{w}))$$

where $\bar{x}, \bar{y}, \bar{z}$ are disjoint tuples of variables, each ϕ_i is a relational atom, and ψ is either:

- *a conjunction of relational atoms, or*
- *a conjunction of equality atoms over $\bar{x}$, or*
- *the symbol $\bot$.*

While EDs may not be the only fragment of $\mathcal{OL}$ that can express suitable ICs, we observe that epistemic dependencies satisfy many of our desiderata. In the following example, we show how the Knowledge Semantics of all the requirements in Example 3 can be faithfully translated into epistemic dependencies.

Example 4 *The following EDs capture Requirement $1-3$ in Example 3 under Knowledge Semantics.*

1. $\forall x, y, y'.\mathbf{K}(hasCode(x, y) \wedge hasCode(x, y')) \rightarrow y = y'$;
2. $\forall x.\mathbf{K}(Emp(x)) \rightarrow \exists y.\mathbf{K}(hasDep(x, y))$;
3. $\forall x.\mathbf{K}(Dep(x)) \rightarrow \exists y.\mathbf{K}(hasDir(x, y))$.

Consider now ED $1-3$ in the OBDM system $\mathcal{O}$ defined in Example 1. For each $\mathfrak{E} \in \mathcal{E}(\mathcal{O})$, we have:

1. *$\mathfrak{E} \models \mathbf{K}(hasCode(x, y))$ if and only if $x = Bob$ and $y = 1B$, or $x = Jim$ and $y = 2J$, or $x = Wim$ and $y = 3W$. Hence, $\mathcal{O}$ satisfies ED 1.*

2. *$\mathfrak{E} \models \mathbf{K}(Emp(Wim))$ but $\mathfrak{E} \not\models \mathbf{K}(hasDep(Wim, c))$ for any constant c. Hence, $\mathcal{O}$ does not satisfy ED 2.*
3. *$\mathfrak{E} \models \mathbf{K}(Dep(x))$ if and only if $x = D0$ or $x = D1$, and $\mathfrak{E} \models \mathbf{K}(hasDir(x, y))$ if $x = D0$ and $y = Tim$, or $x = D1$ and $y = Jim$. Hence, $\mathcal{O}$ satisfies ED 3.*

Example 4, shows how the violation or satisfaction of an epistemic dependency is really grounded on the data contained in the sources. Consider, e.g., ED 2. While the OBDM system $\mathcal{O}$ entails the axiom $Emp \sqsubseteq \exists hasDep$, $\mathcal{O}$ violates ED 2 due to employee Wim. In turn, while $\mathcal{O}$ does not entail that employee codes are unique, $\mathcal{O}$ satisfies ED 1. This is due to the fact that, in the data sources, every employee has a unique code.

In the reminder of this section, we discuss standard reasoning tasks in the context of OBDM specifications and systems with constraints. The first of these tasks is *checking satisfiability*, i.e., given an OBDM system with constraints $\mathcal{O}$, check whether $Mod(\mathcal{O}) \neq \emptyset$. From Definition 4, it follows that checking whether $\langle \mathcal{B}_\mathcal{C}, D\rangle$ is satisfiable can be done in two steps: first, check whether $\langle \mathcal{B}, D\rangle$ alone is satisfiable, second, check whether $\langle \mathcal{B}_\mathcal{C}, D\rangle$ satisfies its constraints. While the complexity of the former is well-known for many ontological languages, we present the complexity of checking whether a lightweight OBDM system satisfies a set of epistemic dependencies in the following section.

Another important task that can be performed with OBDM systems with constraints is *query answering*. Definition 4 ensures that, after checking whether a system $\langle \mathcal{B}_\mathcal{C}, D\rangle$ is satisfiable, the set of constraints $\mathcal{C}$ does not affect query answering and can be safely disregarded. This observation is formalized in the following proposition.

Proposition 3 *Given a satisfiable OBDM system with constraints $\langle \mathcal{B}_\mathcal{C}, D\rangle$, a tuple $\bar{c}$ is in $cert(\phi(\bar{x}), \langle \mathcal{B}_\mathcal{C}, D\rangle)$ if and only if $\bar{c} \in cert(\phi(\bar{x}), \langle \mathcal{B}, D\rangle)$.*

In light of Proposition 3, in what follows we will not discuss query answering any further.

When dealing with standard ICs, an important task is to check whether a constraint χ is redundant with respect to a set of constraints C, i.e., check whether χ is satisfied whenever all the constraints in C are. In OBDM specifications, redundancy can be translated as follows: given an OBDM specification $\mathcal{B}_\mathcal{C}$ and an epistemic dependency κ, check whether every satisfiable system $\langle \mathcal{B}_\mathcal{C}, D\rangle$ also satisfies κ. As we will show in the following section, checking redundancy of epistemic dependencies is undecidable even in the case of trivial OBDM specification.

Redundancy, however, is not the only static analysis task that can be performed with OBDM specifications with constraints. Other interesting tasks are, e.g., checking *faithfulness* and checking *protection* (**?**). Faithfulness and protection are defined as follows.

Definition 6 *Let $\mathcal{B}_\mathcal{C} = \langle \mathcal{T}, \mathcal{M}, \Sigma_\mathcal{S}, \mathcal{C}\rangle$ be an OBDM specification with constraints and let $\mathcal{S} = \langle \Sigma_\mathcal{S}, \mathcal{C}_\mathcal{S}\rangle$ denote a database schema. Then:*

- *$\mathcal{S}$ is faithful to $\mathcal{B}_\mathcal{C}$ if, for every $\Sigma_\mathcal{S}$-database D, D is consistent with $\mathcal{S}$ if $Mod(\langle \mathcal{B}_\mathcal{C}, D\rangle) \neq \emptyset$;*
- *$\mathcal{S}$ protects $\mathcal{B}_\mathcal{C}$ if, for every $\Sigma_\mathcal{S}$-database D, $Mod(\langle \mathcal{B}_\mathcal{C}, D\rangle) \neq \emptyset$ if D is consistent with $\mathcal{S}$.*

Informally, faithfulness and protection are a way to evaluate a given data source schema at the conceptual level, via the OBDM specification. In this context, faithfulness and protection are relevant schema-level measures of the quality of data, as well as useful tools for the OBDM paradigm. For example, when OBDM specifications are paired with data sources that evolve through time, checking satisfiability may become a frequent task. If the source schema protects the specification, however, satisfiability can be safely delegated to the systems managing the data sources (**?**). In the same spirit, protection and faithfulness can be used as effectual designing tools for data source schemas. The following example illustrates protection.

Example 5 *Consider the OBDM specification $\mathcal{B}$ defined in Example 1, the database schema $\mathcal{S}$ in Example 2, and the epistemic dependencies in Example 4. The schema $\mathcal{S}$ protects $\mathcal{O}$ from ED 3. Informally, this is because the departments known by a system $\mathcal{O} = \langle \mathcal{B}, D\rangle$ are those stored in tables* dep *and* empA*. For the former, $\mathcal{O}$ always knows at least one manager while, for the latter, it may not. However, due to the integrity constraints in $\mathcal{S}$, the departments stored in table* empA *are a subset of those stored in* dep*. In turn, this implies that every known department is associated to a manager that is also known.*

4 Decidability and Complexity

In this section, we study the complexity of performing the reasoning tasks presented in Section 3. To this end, we start by defining relevant decisions problems.

- **Satisfiability:** given an OBDM system with constraints $\mathcal{O}$, check whether $Mod(\mathcal{O}) \neq \emptyset$.
- **Protection:** given an OBDM specification with constraints $\mathcal{B}_\mathcal{C}$ and a database schema $\mathcal{S}$, check whether $\mathcal{S}$ protects $\mathcal{B}_\mathcal{C}$.
- **Constraint Implication:** given an OBDM specification with constraints $\mathcal{B}_\mathcal{C}$ and a formula $\chi \in \mathcal{OL}$, check whether every satisfiable system $\langle \mathcal{B}_\mathcal{C}, D\rangle$ satisfies χ.
- **Faithfulness:** given an OBDM specification with constraints $\mathcal{B}_\mathcal{C}$ and a database schema $\mathcal{S}$, check whether $\mathcal{S}$ is faithful to $\mathcal{B}_\mathcal{C}$.

For our complexity analysis, we will assume integrity constraints expressed as epistemic dependencies and lightweight OBDM specifications. This assumption will lead to decidability for Protection and even tractability, if the specification is fixed, for Satisfiability. Unfortunately, Constraint Implication and Faithfulness for epistemic dependencies are inherently undecidable. To show that undecidability only depends on the constraints, for Constraint Implication and Faithfulness we will present undecidability results using trivial OBDM specifications.

Satisfiability. To present our complexity results, we need to prove several preliminary lemmas. First, we show that the satisfaction of epistemic formulae in OBDM systems is closely related to the problem of computing certain answers to queries. This is formalized in the following lemma.

Lemma 1 *Let $\mathcal{O}$ be a satisfiable OBDM system and let $\phi(\bar{x})$ be a objective $\mathcal{OL}$ formula. The formula $\mathbf{K}(\phi(\bar{c}))$, for some tuple of constants $\bar{c}$, is true in the knowledge states of $\mathcal{O}$ if and only if $\bar{c} \in cert(\phi(\bar{x}), \mathcal{O})$.*

To check whether $\mathcal{O} = \langle \mathcal{B}_\mathcal{C}, D\rangle$ satisfies its constraints, we can encode each epistemic dependency κ in $\mathcal{C}$ as a pair of queries, and make use of Lemma 1 to check whether $\mathcal{O}$ satisfies κ. This encoding is presented in the following definition.

Definition 7 *Let κ be the epistemic dependency*

$$\forall \bar{x}.\mathbf{K}(\exists \bar{z}. \bigwedge_i \phi_i(\bar{x}, \bar{z})) \rightarrow \exists \bar{y}.\mathbf{K}(\exists \bar{w}.\psi(\bar{x}, \bar{y}, \bar{w})).$$

The queries $b_\kappa(\bar{x})$ and $h_\kappa(\bar{x})$ are defined as follows:
- *$b_\kappa(\bar{x}) = \exists \bar{z}.\phi(\bar{x}, \bar{z})$;*
- *if ψ is a conjunction of relational atoms, then $h_\kappa(\bar{x}, \bar{y}) = \exists \bar{w}.\psi(\bar{x}, \bar{y}, \bar{w})$;*
- *if ψ is a conjunction of equality atoms $\bigwedge_i(x_1^i = x_2^i)$, then $h_\kappa(\bar{x}) = b_\kappa(\bar{x}) \wedge \bigwedge_i(x_1^i = x_2^i)$;*
- *if $\psi(\bar{x}) = \bot$, then h_κ is the empty query.*

Given an epistemic dependency κ, $b_\kappa^\mathcal{B}(\bar{x})$ will denote the UCQ $Rew(b_\kappa(\bar{x}), \mathcal{B})$ and $h_\kappa^\mathcal{B}(\bar{x})$ will denote the UCQ $\exists \bar{y}.Rew(h_\kappa(\bar{x}, \bar{y}))$. Due to Lemma 1, $b_\kappa^\mathcal{B}(\bar{x})$ and $h_\kappa^\mathcal{B}(\bar{x})$ can be used to test the satisfaction of κ over $\mathcal{B}$. We formalize this claim in the following lemma.

Lemma 2 *Let $\mathcal{B}$ be a lightweight OBDM specification. A satisfiable OBDM system with constraints $\langle \mathcal{B}_\mathcal{C}, D\rangle$ satisfies an epistemic dependency κ if and only if $ans(b_\kappa^\mathcal{B}(\bar{x}), D)$ is contained in $ans(h_\kappa^\mathcal{B}(\bar{x}), D)$.*

With Lemma 2 in place, we are now ready to prove the complexity of checking satisfiability of lightweight OBDM systems with constraints. We present this result in the following theorem.

Theorem 1 *Let $\mathcal{B}$ be a lightweight OBDM specifications and let $\mathcal{C}$ be a set of epistemic dependencies. For input $\langle \mathcal{B}_\mathcal{C}, D\rangle$, Satisfiability is Π_2^p-complete. If $\mathcal{B}_\mathcal{C}$ is fixed and the only input is D, Satisfiability is in AC^0.*

To prove membership in Π_2^p, we can show an non-deterministic algorithm that guesses a tuple $\bar{c}$ that satisfies a disjunct in $b_\kappa^\mathcal{B}(\bar{x})$ and checks whether $\bar{c}$ does not satisfy any of the disjuncts in $h_\kappa^\mathcal{B}(\bar{x})$. For hardness, we can show a reduction from the problem of checking the satisfaction of tgds ((**?**)). To prove membership in AC^0, observe that $\forall \bar{x}.\exists \bar{y}.b_\kappa^\mathcal{B}(\bar{x}) \wedge h_\kappa^\mathcal{B}(\bar{x}, \bar{y})$ is a first-order formula that can be evaluated directly over D.

Protection. First, we observe that, given a specification $\mathcal{B}_\mathcal{C}$, protection can be tested separately over $\mathcal{B}$ and $\mathcal{C}$. To formalize this intuition, we need to introduce some additional notation. Given an OBDM specification $\mathcal{B}_\mathcal{C} = \langle \mathcal{T}, \mathcal{M}, \Sigma_\mathcal{S}, \mathcal{C}\rangle$ and a database schema $\mathcal{S}$, we say that $\mathcal{S}$ protects $\mathcal{B}_\mathcal{C}$ from $\mathcal{B}$ if $\mathcal{S}$ protects the specification $\langle \mathcal{T}, \mathcal{M}, \Sigma_\mathcal{S}, \emptyset\rangle$. Moreover, we say that $\mathcal{S}$ protects $\mathcal{B}_\mathcal{C}$ from $\chi \in \mathcal{C}$ if $\mathcal{S}$ protects the specification $\langle \mathcal{T}, \mathcal{M}, \Sigma_\mathcal{S}, \{\chi\}\rangle$. With these definitions in place, we can prove the following.

Lemma 3 *A database schema $\mathcal{S}$ protects an OBDM specification $\mathcal{B}_\mathcal{C}$ if and only if $\mathcal{S}$ protects $\mathcal{B}_\mathcal{C}$ from $\mathcal{B}$, and $\mathcal{S}$ protects $\mathcal{B}$ from χ, for every $\chi \in \mathcal{C}$.*

Lemma 3 will be the basic building block of our algorithm for Protection. Assume a specification $\mathcal{B}_\mathcal{C}$, where $\mathcal{B}$ is lightweight and $\mathcal{C}$ is a set of epistemic dependencies. Informally, while a technique to check whether $\mathcal{S}$ protects $\mathcal{B}_\mathcal{C}$ from $\mathcal{B}$ is known in the literature (**?**), checking whether $\mathcal{S}$ protects $\mathcal{B}_\mathcal{C}$ from κ can be done via query containment. This intuition is formalized in the following lemma.

Lemma 4 *A database schema $\mathcal{S} = \langle \Sigma_\mathcal{S}, \mathcal{C}_\mathcal{S}\rangle$ protects the lightweight OBDM specification $\mathcal{B}_\mathcal{C}$ from the epistemic dependency $\kappa \in \mathcal{C}$ if and only if the following hold:*
- *$\mathcal{S}$ protects $\mathcal{B}_\mathcal{C}$ from $\mathcal{B}$, and*
- *$ans(b_\kappa^\mathcal{B}(\bar{x}), D) \subseteq ans(h_\kappa^\mathcal{B}(\bar{x}), D)$, for every $D \models \mathcal{C}_\mathcal{S}$.*

With Lemma 4 in place, we can finally prove the complexity of Protection for lightweight OBDM specifications and epistemic dependencies.

Theorem 2 *For input $\mathcal{B}_\mathcal{C}$ and $\mathcal{S}$, where $\mathcal{B}$ is a lightweight OBDM specifications, $\mathcal{C}$ is a set of epistemic dependencies, and $\mathcal{S}$ is a database schema with weakly acyclic tgds, egds, and dens, Protection is $2EXPTIME$-complete.*

Intuitively, the proof of Theorem 2 relies on the finiteness of the chase of weakly-acyclic sets of weakly acyclic tgds, egds, and dens.

Constraint Implication and Faithfulness As we briefly discussed at the beginning of this section, both Constraint Implication and Faithfulness are undecidable even for the class of OBDM specifications that we called *trivial*. To prove these claims, we need to introduce some additional notation and some preliminary results. A k-tgd is an epistemic dependency of the form $\forall \bar{x}.\mathbf{K}(\phi(\bar{x})) \rightarrow \exists \bar{y}\mathbf{K}(\psi(\bar{x}, \bar{y}))$, where ψ is a conjunction of relational atoms. Given a k-tgd κ, by $\kappa_\bot$ we denote the tgd obtained by removing from κ every occurrence of $\mathbf{K}$, i.e., $\kappa_\bot = \forall \bar{x}.\phi(\bar{x}) \rightarrow \exists \bar{y}\psi(\bar{x}, \bar{y})$. Intuitively, κ can be used to simulate $\kappa_\bot$, as the following lemma shows.

Lemma 5 *Let $\mathcal{B}$ be a trivial OBDM specification and let $\mathcal{C}$ be a set of k-tgds. The OBDM system $\langle \mathcal{B}_\mathcal{C}, D\rangle$ satisfies its constraints if and only if D satisfies $\kappa_\bot$, for every $\kappa \in \mathcal{C}$.*

Informally, Lemma 5 shows that epistemic dependencies can simulate binary database dependencies. To prove undecidability of Constraint Implication, we can use this result to show a reduction from the Finite Implication problem, i.e., given a schema $\mathcal{S} = \langle \Sigma_\mathcal{S}, \mathcal{C}_\mathcal{S}\rangle$ and a tgd τ, check whether every S-database satisfies τ. Undecidability of Finite Implication was proved in (**?**) even for binary tuple-generating dependencies.

Theorem 3 *Constraint Implication is undecidable for trivial OBDM specifications and epistemic dependencies.*

Let $\kappa_\bot$ be a binary tgd, and let κ be the associated k-tgd. To prove Theorem 3, given an instance of Finite Implication $\langle \mathcal{S}, \kappa_\bot\rangle$, we can construct a specification $\mathcal{B}_\mathcal{C}$, where $\mathcal{B}$ is trivial, such that $\mathcal{B}_\mathcal{C}$ implies κ if and only if $\mathcal{S}$ finitely entails $\kappa_\bot$. A similar construction proves the following.

Theorem 4 *With input $\mathcal{B}_{\mathcal{C}}$ and τ, where $\mathcal{B}$ is a trivial OBDM specification, $\mathcal{C}$ is a set of epistemic dependencies, and τ is a tgd, Faithfulness is undecidable.*

5 Conclusions

We provided a framework for integrity constraints in OBDM systems based on the notion of what such systems know and should know about the real world. In this framework, we defined a language for constraints and studied the complexity of satisfaction and different forms of static analysis. As future directions, we would like to study the decidability and complexity of different languages for ontologies, mappings, and constraints.

6 Acknowledgements

This work was partly supported by EPSRC (grants M025268 and N023056) and by MIUR (PRIN 2017 project "HOPE").