Query Definability and Its Approximations in Ontology-based Data Management

Gianluca Cima, Federico Croce, Maurizio Lenzerini

Query Definability and Its Approximations in Ontology-based Data Management

Gianluca Cima
gianluca.cima@u-bordeaux.fr
CNRS & University of Bordeaux
Bordeaux, France

Federico Croce
croce@diag.uniroma1.it
Sapienza University of Rome
Rome, Italy

Maurizio Lenzerini
lenzerini@diag.uniroma1.it
Sapienza University of Rome
Rome, Italy

ABSTRACT

Given an input dataset (i.e., a set of tuples), query definability in Ontology-based Data Management (OBDM) amounts to finding a query over the ontology whose certain answers coincide with the tuples in the given dataset. We refer to such a query as a *characterization* of the dataset with respect to the OBDM system. Our first contribution is to propose approximations of perfect characterizations in terms of recall (complete characterizations) and precision (sound characterizations). A second contribution is to present a thorough complexity analysis of three computational problems, namely verification (check whether a given query is a perfect, or an approximated characterization of a given dataset), existence (check whether a perfect, or a best approximated characterization of a given dataset exists), and computation (compute a perfect, or best approximated characterization of a given dataset).

CCS CONCEPTS

• **Information systems** → *Ontologies*; **Query languages**; • **Computing methodologies** → **Knowledge representation and reasoning**.

KEYWORDS

Ontology Based Data Management; Semantic Technologies

ACM Reference Format:
Gianluca Cima, Federico Croce, and Maurizio Lenzerini. 2021. Query Definability and Its Approximations in Ontology-based Data Management. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3482466

1 INTRODUCTION

As first introduced for relational databases [5, 6, 36, 39], query definability is the reverse engineering task that, given a set of tuples and a database, aims at finding a query whose answers over such database are exactly the tuples in the set. In other words, the goal of this task is to derive an intensional definition (the query) of an extensionally defined set. Over the years, researchers have found several interesting applications of this problem, spanning from simplifying query formulation by non-experts, to debugging facilities for data engineers. Moreover, the query definability has been studied as a useful tool for data exploration, data analysis, usability, data security and more [26, 28]. With the rise of Machine Learning (ML), we argue that this topic could be also beneficial for providing meaningful reformulations of what is called a training dataset in any typical supervised ML-based classification task. In this context, the training set used in a classification task is seen as a set of tuples in a database schema, and the query derived by solving the query definability problem results into an intensional definition of the input training set. In a sense, the expression derived can be used as an explanation of the intensional properties of the training set. The idea is that an intensional characterization of the training set can help understanding the behaviour of a classifier, a very important task for wide and safe adoption of machine learning and data mining technologies, especially in dealing with bias.

In this paper, we address the problem of query definability in the context of Ontology-based Data Management (OBDM), which is a paradigm for accessing data using a conceptual representation of the domain of interest expressed as an ontology. OBDM relies on a three-level architecture, consisting of the schema of the data layer $\mathcal{S}$ (which we assume constituted by a relational schema), the ontology $\mathcal{O}$, a declarative and explicit representation of the domain of interest for the organization, and the mapping $\mathcal{M}$ between the two. Consequently, an OBDM specification is formalized as the triple $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ which, together with an $\mathcal{S}$-database D, form a so-called OBDM system $\Sigma = \langle J, D \rangle$. In this context, we are going to tackle the problem of query definability by leveraging the notion of evaluation of a query with respect to an OBDM system, in turn based on the notion of *certain answers* to a query over an OBDM system.

Intuitively, given an OBDM system $\Sigma = \langle J, D \rangle$ and a D-dataset λ, our goal is to derive a query expression over $\mathcal{O}$ that suitably *characterizes* λ w.r.t. Σ. In other words, we aim at deriving a "good" definition of λ using a query expressed over the concepts and roles of the ontology $\mathcal{O}$ of J.

Inspired by the works in [22, 29] about query definability in Description Logics (DLs), we consider the query whose certain answers with respect to $\Sigma = \langle J, D \rangle$ is exactly λ as the perfect characterization for λ. We note that, since in this paper we tackle the query definability problem in OBDM, differently from the works in [22, 29], this work has the added complexity of considering the mapping layer of the OBDM system, which is, to the best of our knowledge, novel to this field. This work has also been inspired by the *concept learning* tools presented in [7, 20, 34], and by the notion of *query abstraction* [13, 14, 16]. We differ from the former because

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia

ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3482466

in that work the goal is to learn a concept expression capturing a given dataset, whereas our goal is to derive a full-blown query that, evaluated over the ontology, returns the dataset as answers. We differ from the latter because, although the goal is stil to derive a query expression over the ontology, in that work the input is a query over the data layer, whereas in query definability the input is a set of tuples. It follows that the two tasks are completely different and require different technical solutions: in the present work we aim at finding a query over the ontology such that the certain answers of the query w.r.t. the OBDM system are equal to the given specific dataset, whereas in [13, 14, 16], the goal is to find a query over the ontology such that the certain answers of the query are equal to the evaluation of the given query over the database schema, for all possible databases of the OBDM system. In the framework section of this paper we will better characterize the relationship between the two notions of query definability and query abstraction.

Virtually all the above-mentioned works point out that in many cases a perfect ontological characterization of a given dataset does not exist. We argue that, in these cases, reasonable and useful ontological characterizations can still be provided. In particular, we propose to resort to suitable approximations of the perfect characterizations, in terms of recall and precision. To this end, we introduce the notions of sound and complete characterizations. The former is a query whose certain answers form a subset of the D-dataset λ in input, whereas the certain answers of the latter, form a superset of the D-dataset λ. Obviously, we are interested in computing the best approximated characterizations, which we call maximally sound and minimally complete characterizations, respectively. A maximally sound (resp., minimally complete) characterization is a sound (resp., complete) characterization such that no other sound (resp., complete) characterization exists that better approximates the D-dataset λ.

This paper provides the following contributions:

- We present a general, formal framework for the various notions of ontological characterizations mentioned above. The framework includes the definition of three tasks that are relevant for reasoning about characterizations of a dataset, namely verification (verify whether a given query is a sound, complete, or perfect characterizations), computation (compute a characterization of a certain type), and existence (check whether a characterization of a certain type exists).
- We provide computational complexity results for the three reasoning tasks mentioned above in a scenario that uses the most common languages in the OBDM literature, namely where the ontology language is *DL-Lite*$_{\mathcal{R}}$, the mapping language is GLAV, and the query language to express characterizations is the one of union of conjunctive queries. As for the two decision problems of verification and existence, we provide both upper bounds and matching lower bounds. As for the computation task, we provide algorithms for computing perfect, minimally complete, and maximally sound characterizations, provided they exist.

The paper is organized as follows. After the preliminaries in Section 2, Section 3 illustrates the framework, and Sections 4, 5, and 6 present the results on the three reasoning tasks, i.e., verification, computation and existence, respectively. Finally, Section 7 concludes the paper by discussing possible future work.

2 PRELIMINARIES

We recall some notations and languages about relational databases [1], Description Logics (DLs) [4], and the Ontology-based Data Management (OBDM) paradigm [24].

Databases, Datasets, and Queries: A *relational database schema* (or simply *schema*) $\mathcal{S}$ is a finite set of predicate symbols, each with a specific arity. Given a schema $\mathcal{S}$, an $\mathcal{S}$*-database* D is a finite set of *facts* satisfying all integrity constraints in $\mathcal{S}$ whose form is $s(\vec{c})$, where s is an n-ary predicate symbol of $\mathcal{S}$, and $\vec{c} = (c_1, \ldots, c_n)$ is an n-tuple of constants, each taken from a countable infinite set of symbols denoted by Const. We denote by $dom(D)$ the finite set of constants occurring in D. Observe that $dom(D) \subseteq$ Const.

Given a schema $\mathcal{S}$ and an $\mathcal{S}$-database D, a D*-dataset* λ of arity n is a finite set of n-tuples $\vec{c}$ of constants occurring in D, i.e., $\lambda \subseteq dom(D)^n$.

A *query* $q_{\mathcal{S}}$ over a schema $\mathcal{S}$ is an expression in a certain query language $\mathcal{Q}$ using the predicate symbols of $\mathcal{S}$ and arguments of predicates are *variables*, i.e., we disallow constants to occur in queries. Each query has an associated arity. The *evaluation* of a query $q_{\mathcal{S}}$ of arity n over an $\mathcal{S}$-databases D is a set of *answers* $q_{\mathcal{S}}^D$, each answer being an n-tuple of constants occurring in $dom(D)$, i.e., $q_{\mathcal{S}}^D \subseteq dom(D)^n$. We are particularly interested in *conjunctive queries* and unions thereof.

A *conjunctive query (CQ)* over a schema $\mathcal{S}$ is an expression of the form $q_{\mathcal{S}} = \{\vec{x} \mid \exists \vec{y}.\phi(\vec{x}, \vec{y})\}$ such that (*i*) $\vec{x} = (x_1, \ldots, x_n)$, called the *target list* of $q_{\mathcal{S}}$, is an n-tuple of *distinguished variables*, where n is the arity $q_{\mathcal{S}}$ (*ii*) $\vec{y} = (y_1, \ldots, y_m)$ is an m-tuple of *existential variables*; and (*iii*) $\phi(\vec{x}, \vec{y})$, called the *body* of $q_{\mathcal{S}}$, is a finite conjunction of atoms of the form $s(v_1, \ldots, v_p)$, where s is a p-ary predicate symbol of $\mathcal{S}$ and v_i is either a distinguished or an existential variable, i.e., $v_i \in \vec{x} \cup \vec{y}$, for each $i = [1, p]$. Variables belong to a countable infinite set of symbols denoted by $\mathcal{V}$, where Const $\cap\, \mathcal{V} = \emptyset$. A *union of conjunctive queries (UCQ)* is a finite set of CQs with same arity, called its *disjuncts*.

For a conjunction of atoms $\phi(\vec{x}, \vec{y})$, we denote by $set(\phi)$ the set of all the atoms occurring in ϕ. For a set of atoms C and a tuple $\vec{c} = (c_1, \ldots, c_n)$ of constants, we denote by $query(C, \vec{c})$ the CQ $\{\vec{x} \mid \exists \vec{y}.\phi(\vec{x}, \vec{y})\}$, where (*i*) $\phi(\vec{x}, \vec{y})$ is the conjunction of all the atoms occurring in the set of atoms C', where C' is obtained from C by replacing everywhere each constant c_i occurring in $\vec{c}$ with a fresh variable x_{c_i} and each constant c not occurring in $\vec{c}$ with a fresh variable y_c, (*ii*) $\vec{x} = (x_{c_1}, \ldots, x_{c_n})$, and (*iii*) $\vec{y}$ is the tuple of all variables occurring in C' that do not occur in $\vec{x}$.

Following the terminology of [35], we say that a query $q_{\mathcal{S}}$ over a schema $\mathcal{S}$ *defines a* D*-dataset* λ *inside an* $\mathcal{S}$*-database* D if $q_{\mathcal{S}}^D = \lambda$, and say that λ is $\mathcal{Q}$*-definable inside* D, for a query language $\mathcal{Q}$, if there exists a query $q_{\mathcal{S}} \in \mathcal{Q}$ that defines λ inside D.

Given a set of atoms C, we denote by $dom(C)$ the set of all constants and variables occurring in a set of atoms C. Observe that $dom(C) \subseteq$ Const $\cup\, \mathcal{V}$. Let C_1 and C_2 be two sets of atoms. We say that a function $h : dom(C_1) \rightarrow dom(C_2)$ is a *homomorphism* from C_1 to C_2 if $h(C_1) \subseteq h(C_2)$, where $h(C_1)$ is the image of C_1

under h, i.e., $h(C_1) = \{h(\alpha) \mid \alpha \in C_1\}$ with $h(s(t_1, \ldots, t_n)) = s(h(t_1), \ldots, h(t_n))$ for each atom $\alpha = s(t_1, \ldots, t_n)$. For two sets of atoms C_1 and C_2 and two tuples of terms $\vec{t_1}$ and $\vec{t_2}$, we write $(C_1, \vec{t_1}) \rightarrow (C_2, \vec{t_2})$ if there is a function h from $dom(C_1) \cup \vec{t_1}$ to $dom(C_2) \cup \vec{t_2}$ such that (*i*) h is a homomorphism from C_1 to C_2, and (*ii*) $h(\vec{t_1}) = \vec{t_2}$ (where, for a tuple of terms $\vec{t} = (t_1, \ldots, t_n)$, $h(\vec{t}) = (h(t_1), \ldots, h(t_n))$), $(C_1, \vec{t_1}) \nrightarrow (C_2, \vec{t_2})$ otherwise.

Observe that for an $\mathcal{S}$-database D and a CQ $q_{\mathcal{S}} = \{\vec{x} \mid \exists \vec{y}.\phi(\vec{x}, \vec{y})\}$ over $\mathcal{S}$ of arity n, the set of answers $q_{\mathcal{S}}^D$ corresponds to the set of n-tuples $\vec{c}$ of constants occurring in D for which $(set(\phi), \vec{x}) \rightarrow (D, \vec{c})$.

Syntax and Semantics of DL-Lite$_\mathcal{R}$: DLs are fragments of First-order logic languages using only unary and binary predicates, called *atomic concepts* and *atomic roles*, respectively. In this paper, a *DL ontology* (or simply *ontology*) $\mathcal{O}$ is a TBox ("Terminological Box") expressed in a specific DL, that is, a set of assertions stating general properties of concepts and roles built according to the syntax of the specific DL, which represents the intensional knowledge of a modeled domain.

We are interested in DL ontologies expressed in *DL-Lite$_\mathcal{R}$*, the member of the *DL-Lite* family [10] that underpins OWL 2 QL, i.e., the OWL 2 profile especially designed for efficient query answering [27]. A *DL-Lite$_\mathcal{R}$* ontology $\mathcal{O}$ is a finite set of *assertions* of the form:

$B_1 \sqsubseteq B_2$	$R_1 \sqsubseteq R_2$	(concept/role inclusion)
$B_1 \sqsubseteq \neg B_2$	$R_1 \sqsubseteq \neg R_2$	(concept/role disjointness)

where B_1, B_2 are basic concepts, i.e., expressions of the form A, $\exists P$, or $\exists P^-$, with A and P an atomic concept and an atomic role, respectively, and R_1 and R_2 basic roles, i.e., expressions of the form P, or P^-.

Given a *DL-Lite$_\mathcal{R}$* ontology $\mathcal{O}$, we denote by $V_\mathcal{O}$ the *$\mathcal{O}$-violation query*, i.e., the boolean UCQ obtained by including a disjunct of the form $\{() \mid \exists y.A_1(y) \wedge A_2(y)\}$ (respectively, $\{() \mid \exists y_1, y_2.A_1(y_1) \wedge R(y_1, y_2)\}$, $\{() \mid \exists y_1, y_2, y_3.R_1(y_1, y_2) \wedge R_2(y_1, y_3)\}$, and $\{() \mid \exists y_1, y_2.R_1(y_1, y_2) \wedge R_2(y_1, y_2)\}$) for each disjointness assertion $A_1 \sqsubseteq \neg A_2$ (respectively, $A_1 \sqsubseteq \neg \exists R$ or $\exists R \sqsubseteq \neg A_1$, $\exists R_1 \sqsubseteq \neg \exists R_2$, and $R_1 \sqsubseteq \neg R_2$) occurring in $\mathcal{O}$, where an atom of the form $R(y, y')$ stands for either $P(y, y')$ if R denotes an atomic role P, or $P(y', y)$ if R denotes the inverse of an atomic role, i.e., $R = P^-$.

The semantics of DL ontologies is specified through the notion of interpretation: an *interpretation* $\mathcal{I}$ for an ontology $\mathcal{O}$ is a pair $\mathcal{I} = \langle \Delta^\mathcal{I}, \cdot^\mathcal{I} \rangle$, where the *interpretation domain* $\Delta^\mathcal{I}$ is a non-empty, possibly infinite set of constants, and the *interpretation function* $\cdot^\mathcal{I}$ assigns to each atomic concept A a set of domain objects $A^\mathcal{I} \subseteq \Delta^\mathcal{I}$, and to each atomic role P a set of pairs of domain objects $P^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$. For the constructs of *DL-Lite$_\mathcal{R}$*, the interpretation function extends to other basic concepts and basic roles as follows: $(\exists P)^\mathcal{I} = \{o \mid \exists o'.\ (o, o') \in P^\mathcal{I}\}$ and $(P^-)^\mathcal{I} = \{(o, o') \mid (o', o) \in P^\mathcal{I}\}$. We often treat interpretations $\mathcal{I}$ for ontologies $\mathcal{O}$ as a (possibly infinite) set of facts over (the predicates in the alphabet of) $\mathcal{O}$.

We say that an interpretation $\mathcal{I}$ for an ontology $\mathcal{O}$ satisfies $\mathcal{O}$, denoted by $\mathcal{I} \models \mathcal{O}$, if $\mathcal{I}$ satisfies every assertion in $\mathcal{O}$. For the *DL-Lite$_\mathcal{R}$* assertions, an interpretation $\mathcal{I}$ satisfies a concept inclusion assertion $B_1 \sqsubseteq B_2$ (respectively, role inclusion assertion $R_1 \sqsubseteq R_2$) if $B_1^\mathcal{I} \subseteq B_2^\mathcal{I}$ (respectively, $R_1^\mathcal{I} \subseteq R_2^\mathcal{I}$), and it satisfies a concept disjointness assertion $B_1 \sqsubseteq \neg B_2$ (respectively, role disjointness assertion $R_1 \sqsubseteq \neg R_2$) if $B_1^\mathcal{I} \cap B_2^\mathcal{I} = \emptyset$ (respectively, $R_1^\mathcal{I} \cap R_2^\mathcal{I} = \emptyset$).

Whenever we speak about queries $q_\mathcal{O}$ over ontologies $\mathcal{O}$, we mean queries in a certain language $\mathcal{Q}$ using the atomic concepts and roles in the alphabet of $\mathcal{O}$ as predicates. For a UCQ $q_\mathcal{O}$ over a *DL-Lite$_\mathcal{R}$* ontology $\mathcal{O}$, we denote by $\mathsf{PerfectRef}(\mathcal{O}, q_\mathcal{O})$ the UCQ computed by executing the algorithm PerfectRef [10] on $\mathcal{O}$ and $q_\mathcal{O}$.

Ontology-based Data Management: According to [24, 32], an *Ontology-based Data Management (OBDM)* specification is a triple $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$, where $\mathcal{O}$ is a DL ontology, $\mathcal{S}$ is a relational database schema, also called *source schema*, and $\mathcal{M}$ is a *mapping*, i.e., a finite set of assertions over the signature $\mathcal{S} \cup \mathcal{O}$ relating the source schema $\mathcal{S}$ to the ontology $\mathcal{O}$. An OBDM system is a pair $\Sigma = \langle J, D \rangle$, where $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ is an OBDM specification and D is an $\mathcal{S}$-database.

The semantics of an OBDM system $\Sigma = \langle \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle, D \rangle$ is given in terms of interpretations $\mathcal{I} = \langle \Delta^\mathcal{I}, \cdot^\mathcal{I} \rangle$ for $\mathcal{O}$ in which the interpretation function $\cdot^\mathcal{I}$ further assigns to each constant $c \in dom(D)$ a domain object $c \in \Delta^\mathcal{I}$. Specifically, we say that an interpretation $\mathcal{I}$ for $\mathcal{O}$ is a *model* of an OBDM system $\Sigma = \langle \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle, D \rangle$ if (*i*) $\mathcal{I} \models \mathcal{O}$, and (*ii*) the pair $\langle \mathcal{I}, D \rangle \models \mathcal{M}$. We say that an OBDM system Σ is *consistent* if it has at least one model, *inconsistent* otherwise.

The set of *certain answers* of a query $q_\mathcal{O}$ over an ontology $\mathcal{O}$ w.r.t. an OBDM system $\Sigma = \langle J, D \rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$, denoted by $cert^D_{q_\mathcal{O}, J}$, is the set of tuples of constants $(c_1, \ldots, c_n)$ occurring in D such that $(c_1^\mathcal{I}, \ldots, c_n^\mathcal{I}) \in q_\mathcal{O}^\mathcal{I}$ for each model $\mathcal{I}$ of Σ, where $\mathcal{I}$ is seen as a set of facts over $\mathcal{O}$. If Σ is inconsistent, then the set of certain answers of any query $q_\mathcal{O}$ over $\mathcal{O}$ w.r.t. Σ is simply the set of all possible tuples of constants occurring in D whose arity is the one of the query. We say that two queries q_1 and q_2 are equivalent w.r.t. an OBDM system $\Sigma = \langle J, D \rangle$ if $cert^D_{q_1, J} = cert^D_{q_2, J}$.

As for the mapping component of an OBDM system, in this paper we are interested in *GLAV* assertions [18], which are assertions of the form $q_\mathcal{S} \rightarrow q_\mathcal{O}$, where $q_\mathcal{S}$ and $q_\mathcal{O}$ are CQs over $\mathcal{S}$ and over $\mathcal{O}$, respectively, with the same target list $\vec{x} = (x_1, \ldots, x_n)$. Special cases of GLAV assertions highly considered in the data integration literature are GAV and LAV assertions [23]: in a GAV (resp., LAV) mapping, $q_\mathcal{O}$ (resp., $q_\mathcal{S}$) is simply an atom without existential variables. A GLAV (resp., GAV, LAV, GAV∩LAV) mapping is a finite set of GLAV (resp., GAV, LAV, both GAV and LAV) assertions.

Given a GLAV mapping $\mathcal{M}$ relating $\mathcal{S}$ to $\mathcal{O}$, an interpretation $\mathcal{I}$ for $\mathcal{O}$, and an $\mathcal{S}$-database D, we have that $\langle \mathcal{I}, D \rangle \models \mathcal{M}$ if $(c_1, \ldots, c_n) \in q_\mathcal{S}^D$ implies $(c_1^\mathcal{I}, \ldots, c_n^\mathcal{I}) \in q_\mathcal{O}^\mathcal{I}$ for each mapping assertion $q_\mathcal{S} \rightarrow q_\mathcal{O}$ occurring in $\mathcal{M}$ and for each possible tuple $(c_1, \ldots, c_n)$ of constants occurring in D.

Let $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ be an OBDM specification where $\mathcal{O} = \emptyset$, i.e., $\mathcal{O}$ has no assertions, and $\mathcal{M}$ is a GLAV mapping. From results of [11, 21], it is well-known that, given a UCQ $q_\mathcal{O}$ over $\mathcal{O}$, by splitting the GLAV mapping $\mathcal{M}$ into a GAV mapping followed by a LAV mapping over an intermediate alphabet, it is always possible to compute a UCQ over $\mathcal{S}$, denoted by $\mathsf{MapRef}(\mathcal{M}, q_\mathcal{O})$, such that $\mathsf{MapRef}(\mathcal{M}, q_\mathcal{O})^D = cert^D_{q_\mathcal{O}, J}$ for each $\mathcal{S}$-database D.

Let now $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ be an OBDM specification where $\mathcal{O}$ is a *DL-Lite$_\mathcal{R}$* ontology and $\mathcal{M}$ is a GLAV mapping. For a UCQ $q_\mathcal{O}$ over $\mathcal{O}$, we denote by $\mathrm{rew}_{q_\mathcal{O}, J}$ the following UCQ over $\mathcal{S}$: $\mathrm{rew}_{q_\mathcal{O}, J} :=$

$\mathsf{MapRef}(\mathcal{M}, \mathsf{PerfectRef}(\mathcal{O}, q_\mathcal{O}))$. By combining the above observation with results of [10], we have that (*i*) $cert^D_{q_\mathcal{O},J} = \mathsf{rew}^D_{q_\mathcal{O},J}$, for each UCQ $q_\mathcal{O}$ over $\mathcal{O}$ and for each $\mathcal{S}$-database D such that $\langle J, D\rangle$ is consistent, and (*ii*) $\langle J, D\rangle$ is inconsistent if and only if $\mathsf{rew}^D_{V_\mathcal{O},J} = \{\langle\rangle\}$, for each $\mathcal{S}$-database D. We note that *DL-Lite*$_\mathcal{R}$ is insensitive to the adoption of the unique name assumption for UCQ answering [3].

Canonical Structure: Given an $\mathcal{S}$-database D and a GLAV mapping $\mathcal{M}$ relating a schema $\mathcal{S}$ to an ontology $\mathcal{O}$, the *chase* [9] of D with respect to $\mathcal{M}$, denoted by $\mathcal{M}(D)$, is the set of atoms computed as follows: (*i*) we start with $\mathcal{M}(D) := \emptyset$; then (*ii*) for every GLAV assertion $\{\vec{x} \mid \exists\vec{y}.\phi_\mathcal{S}(\vec{x},\vec{y})\} \rightarrow \{\vec{x} \mid \exists\vec{z}.\varphi_\mathcal{O}(\vec{x},\vec{z})\}$ in $\mathcal{M}$ and for every tuple of constants $\vec{c}$ such that $(set(\phi_\mathcal{S}), \vec{x}) \rightarrow (D, \vec{c})$, we add to $\mathcal{M}(D)$ the image of the set of atoms $set(\varphi_\mathcal{O})$ under h', that is, $\mathcal{M}(D) := \mathcal{M}(D) \cup h'(\varphi_\mathcal{O}(\vec{x},\vec{z}))$, where h' extends h by assigning to each variable z occurring in $\vec{z}$ a different fresh variable of $\mathcal{V}$ still not present in $dom(\mathcal{M}(D))$. Observe that $\mathcal{M}(D)$ is guaranteed to be finite and can be always computed in exponential time.

We conclude this section with the following observation used in the technical development of the next sections. Let $\Sigma = \langle\langle\mathcal{O}, \mathcal{S}, \mathcal{M}\rangle, D\rangle$ be an OBDM system where $\mathcal{O}$ is a *DL-Lite*$_\mathcal{R}$ ontology and $\mathcal{M}$ is a GLAV mapping. We call the *canonical structure* of $\mathcal{O}$ with respect to $\mathcal{M}$ and D, denoted by $C_\mathcal{O}^{\mathcal{M}(D)}$, the (possibly infinite) set of atoms obtained by first computing $\mathcal{M}(D)$ as described before, and then by chasing $\mathcal{M}(D)$ with respect to the inclusion assertions of $\mathcal{O}$ as described in [10, Definition 5] but using the alphabet $\mathcal{V}$ of variables whenever a new element is needed in the chase. Observe that this latter is a *fair* deterministic strategy, i.e., it is such that if at some point an assertion is applicable, then it will be eventually applied. By combining results of [19, Proposition 4.2] with [10, Theorem 29], it is well-known that, for a UCQ $q_\mathcal{O} = \{\vec{x}_1 \mid \exists\vec{y}_1.\phi^1_\mathcal{O}(\vec{x}_1,\vec{y}_1)\} \cup \ldots \cup \{\vec{x}_p \mid \exists\vec{y}_p.\phi^p_\mathcal{O}(\vec{x}_p,\vec{y}_p)\}$ over $\mathcal{O}$ and a tuple of constants $\vec{c}$, if $\Sigma = \langle J, D\rangle$ is consistent, then we have $\vec{c} \in cert^D_{q_\mathcal{O},J}$ if and only if $(set(\phi^i_\mathcal{O}), \vec{x}_i) \rightarrow (C_\mathcal{O}^{\mathcal{M}(D)}, \vec{c})$ for some $i \in [1, p]$.

3 FRAMEWORK

In what follows, $\Sigma = \langle J, D\rangle$ refers to an OBDM system where $J = \langle\mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ is an OBDM specification and D is an $\mathcal{S}$-database. Intuitively, given a set λ of n-tuples of constants occurring in D (i.e., λ is a D-dataset of arity n), we aim at finding a query $q_\mathcal{O}$ over $\mathcal{O}$ in a certain query language $\mathcal{Q}$ characterizing λ w.r.t. the OBDM system Σ. Since the evaluation of queries is based on certain answers, we are naturally led to the following definition.

Definition 1. *$q_\mathcal{O} \in \mathcal{Q}$ is a* perfect Σ-characterization *of λ in the query language $\mathcal{Q}$, if $cert^D_{q_\mathcal{O},J} = \lambda$.*

Clearly, if a perfect Σ-characterization of λ exists, then it is unique up to Σ-equivalence, and therefore in the following we will always refer to *the* perfect Σ-characterization of λ in the query language $\mathcal{Q}$.

Example 1. *Let $\Sigma = \langle J, D\rangle$ be as follows. $J = \langle\mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ is the OBDM specification such that $\mathcal{O} = \{\mathit{MathStudent} \sqsubseteq \mathit{Student}, \mathit{ForeignStudent} \sqsubseteq \mathit{Student}\}$, $\mathcal{S} = \{s_1, s_2, s_3, s_4, s_5\}$, and $\mathcal{M}$ contains the GAV assertions:*

$$\{(x) \mid s_1(x)\} \rightarrow \{(x) \mid \mathit{Student}(x)\}$$
$$\{(x) \mid s_2(x)\} \rightarrow \{(x) \mid \mathit{Student}(x)\}$$
$$\{(x_1, x_2) \mid s_3(x_1, x_2)\} \rightarrow \{(x_1, x_2) \mid \mathit{EnrolledIn}(x_1, x_2)\}$$
$$\{(x) \mid \exists y.s_3(x, y) \wedge s_4(y)\} \rightarrow \{(x) \mid \mathit{MathStudent}(x)\}$$
$$\{(x) \mid \exists y.s_3(x, y) \wedge s_5(y)\} \rightarrow \{(x) \mid \mathit{ForeignStudent}(x)\}$$

And the $\mathcal{S}$-database is $D = \{s_1(c_4), s_2(c_3), s_4(b_1), s_5(d_1), s_3(c_1, b_1), s_3(c_2, d_1), s_3(c_3, e_1), s_3(c_4, e_2), s_3(c_5, e_3)\}$. For the D-dataset $\lambda = \{(c_1), (c_2), (c_3)\}$, since $q^1_\mathcal{O} = \{(x) \mid \mathit{Student}(x)\}$ and $q^2_\mathcal{O} = \{(x) \mid \exists y.\mathit{EnrolledIn}(x, y)\}$ are such that $cert^D_{q^1_\mathcal{O},J} = \{(c_1), (c_2), (c_3), (c_4)\}$ and $cert^D_{q^2_\mathcal{O},J} = \{(c_1), (c_2), (c_3), (c_4), (c_5)\}$, and since $q^3_\mathcal{O} = \{(x) \mid \mathit{MathStudent}(x)\}$ and $q^4_\mathcal{O} = \{(x) \mid \mathit{ForeignStudent}(x)\}$ are such that $cert^D_{q^3_\mathcal{O},J} = \{(c_1)\}$ and $cert^D_{q^4_\mathcal{O},J} = \{(c_2)\}$, one can verify that no perfect Σ-characterization of λ in UCQ exists.

Notice the difference with the notion of *abstraction* [13, 14], introduced in [12] and studied under various scenarios [16, 17, 25]. In abstraction, we are given an OBDM specification $J = \langle\mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ and a query $q_\mathcal{S}$ over $\mathcal{S}$, and the aim is to find a query $q_\mathcal{O}$ over $\mathcal{O}$, called *the perfect J-abstraction of $q_\mathcal{S}$*, such that $cert^D_{q_\mathcal{O},J} = q^D_\mathcal{S}$ *for each $\mathcal{S}$-database D* for which $\langle J, D\rangle$ is consistent. Conversely, here we are also given an $\mathcal{S}$-database D, and instead of a query $q_\mathcal{S}$ we have a set of tuples λ of constants taken from D, and the aim is to find a query $q_\mathcal{O}$ over $\mathcal{O}$ such that $cert^D_{q_\mathcal{O},J} = \lambda$. The following proposition establishes the relationship between the notion of characterization introduced here and the notion of abstraction.

Proposition 1. *Let $\Sigma = \langle J, D\rangle$ be a consistent OBDM system, λ be a D-dataset, and $q_\mathcal{S}$ be a query that defines λ inside D. If a query $q_\mathcal{O} \in \mathcal{Q}$ is the perfect J-abstraction of $q_\mathcal{S}$, then $q_\mathcal{O}$ is the perfect Σ-characterization of λ in $\mathcal{Q}$.*

The next example shows that the converse of the above proposition does not necessarily hold, thus stressing the fact that the two problems are indeed different.

Example 2. *Let $\Sigma = \langle J, D\rangle$ be as follows:* (i) *$J = \langle\mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ is such that $\mathcal{O} = \emptyset$, $\mathcal{S} = \{s_1, s_2\}$, and $\mathcal{M} = \{m_1, m_2\}$ with $m_1 = \{(x) \mid s_1(x)\} \rightarrow \{(x) \mid A(x)\}$ and $m_2 = \{(x) \mid s_2(x)\} \rightarrow \{(x) \mid A(x)\}$; and* (ii) *$D = \{s_1(c)\}$.*

For the D-dataset $\lambda = \{(c)\}$, one can verify that $q_\mathcal{S} = \{(x) \mid s_1(x)\}$ is such that $q^D_\mathcal{S} = \lambda$ and that $q_\mathcal{O} = \{(x) \mid A(x)\}$ is such that $cert^D_{q_\mathcal{O},J} = \lambda$, i.e., $q_\mathcal{O}$ is the perfect Σ-characterization of λ in CQ. However, the query $q_\mathcal{O}$ is not a perfect J-abstraction of $q_\mathcal{S}$, since for the $\mathcal{S}$-database $D' = \{s_2(c)\}$ we have $cert^{D'}_{q_\mathcal{O},J} = \{(c)\}$ whereas $q^{D'}_\mathcal{S} = \emptyset$.

Clearly, the more expressive the query language $\mathcal{Q}$, the more likely we can express the implicit relationship between the tuples in λ by means of the operators in $\mathcal{Q}$, and therefore the more likely the perfect characterization in $\mathcal{Q}$ exists. Unfortunately, the next example shows that, even without any restriction on the query language, perfect characterizations are not guaranteed to exist even in trivial cases.

Example 3. *Recall the OBDM specification J of the previous example, and let $\Sigma = \langle J, D\rangle$ be the OBDM system with $D = \{s_1(c_1), s_2(c_2)\}$. For the D-dataset $\lambda = \{(c_1)\}$, one can trivially verify that, whatever is the query language $\mathcal{Q}$, there is no query $q_O \in \mathcal{Q}$ for which $cert^D_{q_O,J} = \lambda$.*

Note the importance of the role played by the mapping $\mathcal{M}$ in order to reach this conclusion. Indeed, if we replace m_2 with $\{(x) \mid s_2(x)\} \rightarrow \{(x) \mid B(x)\}$, then the perfect Σ-characterization of λ would be the CQ $\{(x) \mid A(x)\}$.

Borrowing the ideas from [16] to remedy situations where perfect abstractions do not exist, we now introduce approximations of perfect characterizations in terms of recall (complete) and precision (sound).

Definition 2. *$q_O \in \mathcal{Q}$ is a* complete *(resp.,* sound*) Σ-characterization of λ in the query language $\mathcal{Q}$, if $\lambda \subseteq cert^D_{q_O,J}$ (resp., $cert^D_{q_O,J} \subseteq \lambda$).*

Example 4. *Refer to Example 1. We have that q^1_O and q^2_O are complete Σ-characterization of λ, whereas q^3_O and q^4_O are sound Σ-characterization of λ.*

As the above example manifests, there may be several complete and sound characterizations relative to a query language $\mathcal{Q}$. In those cases, the interest is unquestionably in those that best approximate the perfect one.

Definition 3. *q_O is a $\mathcal{Q}$-*minimally complete *(resp., $\mathcal{Q}$-*maximally sound*) Σ-characterization of λ, if q_O is a complete (resp., sound) Σ-characterization of λ in $\mathcal{Q}$ and there is no $q'_O \in \mathcal{Q}$ such that (*i*) q'_O is a* complete *(resp., sound) Σ-characterization of λ and (*ii*) $cert^D_{q'_O,J} \subset cert^D_{q_O,J}$ (resp., $cert^D_{q_O,J} \subset cert^D_{q'_O,J}$).*

Example 5. *Refer again to Example 1. The CQ q^1_O is a UCQ-minimally complete Σ-characterization of λ, whereas q^2_O is not. Both q^3_O and q^4_O are CQ-maximally sound Σ-characterizations of λ, but neither of them is a UCQ-maximally sound Σ-characterization of λ. Indeed, a UCQ-maximally sound Σ-characterization of λ is $q^5_O = q^3_O \cup q^4_O$.*

Given this general framework, there are (at least) three computational problems to consider, with respect to an ontology language $\mathcal{L}_O$, a mapping language $\mathcal{L}_\mathcal{M}$, and a query language $\mathcal{Q}$. Given an OBDM system $\Sigma = \langle\langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle, D\rangle$ and a D-dataset λ, where $\mathcal{O} \in \mathcal{L}_O$ and $\mathcal{M} \in \mathcal{L}_\mathcal{M}$:

- *Verification:* given $q_O \in \mathcal{Q}$, check whether q_O is a perfect (resp., complete, sound) Σ-characterization of λ.
- *Computation:* compute the perfect in $\mathcal{Q}$ (resp., $\mathcal{Q}$-minimally complete, or $\mathcal{Q}$-maximally sound) Σ-characterization of λ, provided it exists.
- *Existence:* check whether there exists a perfect in $\mathcal{Q}$ (resp., $\mathcal{Q}$-minimally complete, or $\mathcal{Q}$-maximally sound) Σ-characterization of λ.

In what follows, if not otherwise stated, we refer to the following scenario which considers by far the most popular languages for the OBDM paradigm: (*i*) $\mathcal{L}_O$ is *DL-Lite*$_\mathcal{R}$, (*ii*) $\mathcal{L}_\mathcal{M}$ is GLAV, and (*iii*) $\mathcal{Q}$ is UCQ.

In this scenario, there are two interesting properties that are worth mentioning. First, since the UCQ language allows for the conjunction (resp., union) operator, if an UCQ-minimally complete (resp., UCQ-maximally sound) Σ-characterization of λ exists, then it is unique up to Σ-equivalence.

Proposition 2. *If q_1 and q_2 are UCQ-minimally complete (resp., UCQ-maximally sound) Σ-characterizations of λ, then they are equivalent w.r.t. Σ.*

Due to the above property, in what follows we simply refer to *the* UCQ-minimally complete (resp., UCQ-maximally sound) Σ-characterization of λ.

Second, as expected, in this scenario perfect characterizations are less likely to exist than in the plain relational database case.

Proposition 3. *Let $\Sigma = \langle J, D\rangle$ be a consistent OBDM system, and λ be a D-dataset. If there exists a perfect Σ-characterization of λ in UCQ, then λ is UCQ-definable inside D.*

In general, the converse of the above proposition does not hold. Indeed, in Example 3, while there is no perfect Σ-characterization of λ in any query language $\mathcal{Q}$, the CQ $q_\mathcal{S} = \{(x) \mid s_1(x)\}$ witnesses that λ is CQ-definable inside D.

4 VERIFICATION

We now define the verification problems for X-query definability (X-VQDEF), where X={Perfect, Complete, Sound}. These decision problems are parametric with respect to the ontology language $\mathcal{L}_O$ to express $\mathcal{O}$, the mapping language $\mathcal{L}_\mathcal{M}$ to express $\mathcal{M}$, and the query language $\mathcal{Q}$ to express q_O.

Problem:	**X-VQDEF($\mathcal{L}_O$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$)**
Input:	An OBDM system $\Sigma = \langle\langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle, D\rangle$, a D-dataset λ, and a query $q_O \in \mathcal{Q}$ over $\mathcal{O}$, where $\mathcal{O} \in \mathcal{L}_O$ and $\mathcal{M} \in \mathcal{L}_\mathcal{M}$.
Question:	Is q_O a **X** Σ-characterization of λ?

In what follows, given a syntactic object x such as a query, an ontology, or a mapping, we denote by $\sigma(x)$ its size.

Theorem 1. *Complete-VQDEF(DL-Lite*$_\mathcal{R}$*, GLAV, UCQ) is in NP.*

Proof. We now show how to check whether q_O is a complete Σ-characterization of λ (i.e., $\lambda \subseteq cert^D_{q_O,J}$) in NP, where $\Sigma = \langle J, D\rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$.

Let n be the arity of the tuples in the D-dataset λ. For each n-tuple of constants $\vec{c} \in \lambda$, we first guess (*i*) a CQ q'_O over $\mathcal{O}$ which is either of arity n and size at most $\sigma(q_O)$, or a boolean one capturing a disjointness assertion d (e.g., $\{() \mid \exists y.A_1(y) \wedge A_2(y)\}$ capturing $d = A_1 \sqsubseteq \neg A_2$); (*ii*) a sequence ρ_O of ontology assertions; (*iii*) a CQ $q_\mathcal{S}$ over $\mathcal{S}$ of size at most $\sigma(\mathcal{M}) \cdot \sigma(q'_O)$ which is either of arity n and of the form $\{\vec{x} \mid \exists \vec{y}.\phi_\mathcal{S}(\vec{x}, \vec{y})\}$, or a boolean one of the form $\{() \mid \exists \vec{y}.\phi_\mathcal{S}(\vec{y})\}$; (*iv*) a sequence $\rho_\mathcal{M}$ of mapping assertions; and (*v*) a function f from the variables occurring in $q_\mathcal{S}$ to $dom(D)$.

Then, we check in polynomial time whether (*i*) by means of ρ_O, either we can rewrite a disjunct of q_O into q'_O through $\mathcal{O}$ (i.e., $q'_O \in \mathsf{PerfectRef}(\mathcal{O}, q_O)$), or we can rewrite a disjunct of V_O into

q'_O through $\mathcal{O}$ (i.e., $q'_O \in \text{PerfectRef}(\mathcal{O}, V_O)$); (*ii*) by means of $\rho_\mathcal{M}$ we can rewrite q'_O into $q_\mathcal{S}$ through $\mathcal{M}$ (i.e., $q_\mathcal{S} \in \text{MapRef}(\mathcal{M}, q'_O)$, and thus either $q'_O \in \text{rew}_{q_O,J}$ or $q'_O \in \text{rew}_{V_O,J}$); and finally (*iii*) f consists in a homomorphism witnessing either $(set(\phi_\mathcal{S}), \vec{x}) \to (D, \vec{c})$, i.e., $\vec{c} \in q^D_\mathcal{S}$ (and therefore $\vec{c} \in \text{rew}^D_{q_O,J}$, which means $\vec{c} \in cert^D_{q_O,J}$), or $(set(\phi_\mathcal{S}), ()) \to (D, ())$, i.e., $D \models q_\mathcal{S}$ (and therefore $\text{rew}^D_{V_O,J} = \{\langle\rangle\}$, which means that Σ is inconsistent and thus $\vec{c} \in cert^D_{q_O,J}$ by definition). □

Theorem 2. *Sound-VQDEF(DL-Lite$_\mathcal{R}$, GLAV, UCQ) is in coNP.*

Proof. We now show how to check whether q_O is not a sound Σ-characterization of λ (i.e., $cert^D_{q_O,J} \not\subseteq \lambda$) in NP, where $\Sigma = \langle J, D\rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$.

We first guess (*i*) a tuple of constants $\vec{c}$, and, exactly as in the proof of Theorem 1, (*ii*) q'_O, ρ_O, $q_\mathcal{S}$, $\rho_\mathcal{M}$, and f. Then, we check in polynomial time whether (*i*) $\vec{c}$ contains only constants from $dom(D)$ and $\vec{c} \notin \lambda$ (i.e., $\vec{c} \in dom(D)^n \setminus \lambda$), and (*ii*) using q'_O, ρ_O, $q_\mathcal{S}$, $\rho_\mathcal{M}$, and f, we follow exactly the same polynomial time procedure in the proof of Theorem 1 to check whether $\vec{c} \in cert^D_{q_O,J}$. □

Recall that a decision problem is in DP if and only if it is the conjunction of a decision problem in NP and a decision problem in coNP [30]. Since q_O is a perfect Σ-characterization of λ if and only if it is both a sound, and a complete Σ-characterization of λ, we immediately derive the following upper bound.

Corollary 3. *Perfect-VQDEF(DL-Lite$_\mathcal{R}$, GLAV, UCQ) is in DP.*

We now provide matching lower bounds. We show that they already hold for the same, very simple, fixed OBDM system Σ and dataset λ, and for single, unary CQs as queries.

Theorem 4. *There is an OBDM system $\Sigma = \langle\langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle, D\rangle$ such that $\mathcal{O} = \emptyset$ and $\mathcal{M}$ is a GAV∩LAV mapping, and a D-dataset λ containing only a unary tuple for which the problem Complete-VQDEF(∅, GAV∩LAV, CQ) (resp., Sound-VQDEF(∅, GAV∩LAV, CQ), Perfect-VQDEF(∅, GAV∩LAV, CQ)) is NP-hard (resp., coNP-hard, DP-hard).*

Proof (Sketch.) Let $\Sigma = \langle J, D\rangle$ be the fixed OBDM system such that (*i*) $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ is an OBDM specification in which $\mathcal{O} = \emptyset$ is an empty ontology whose alphabet contains two atomic roles P_1 and P_2, $\mathcal{S} = \{s_1, s_2\}$, and $\mathcal{M}$ contains the following two GAV∩LAV assertions: $\{(x_1, x_2) \mid s_1(x_1, x_2)\} \to \{(x_1, x_2) \mid P_1(x_1, x_2)\}$, and $\{(x_1, x_2) \mid s_2(x_1, x_2)\} \to \{(x_1, x_2) \mid P_2(x_1, x_2)\}$, which simply mirrors source predicate s_i to atomic role P_i, for $i = [1, 2]$, and (*ii*) D is the $\mathcal{S}$-database composed of the following facts:

$$\{s_1(x, y) \mid x = \{r', g', b'\} \text{ and } y = \{r', g', b'\} \text{ and } x \neq y\} \cup$$
$$\{s_1(x, y) \mid x = \{r, g, b, y\} \text{ and } y = \{r, g, b, y\} \text{ and } x \neq y\} \cup$$
$$\{s_2(x, c_3) \mid x = \{r', g', b'\}\} \cup \{s_2(x, c_4) \mid x = \{r, g, b, y\}\}.$$

Let, moreover, λ be the fixed D-dataset $\lambda = \{(c_4)\}$.

Let $G = (V, E)$ be a finite and undirected graph without loops or isolated nodes, where $V = \{y_1, \ldots, y_n\}$. We define a CQ $q_G = \{(x) \mid \exists \vec{y}.\phi_O(x, \vec{y})\}$ over $\mathcal{O}$ as follows:

$$\{(x) \mid \exists y_1, \ldots, y_n. \bigwedge_{(y_i, y_j) \in E} (P_1(y_i, y_j)) \wedge \bigwedge_{y_i \in V} (P_2(y_i, x))\}$$

Notice that q_G can be constructed in LogSpace from an input graph G. Furthermore, for both $i = 3$ and $i = 4$ and for any graph $G = (V, E)$ as above, it can be shown that G is i-colourable if and only if $(c_i) \in cert^D_{q_G,J}$. With this property at hand, it is not hard to prove the claimed lower bounds. Here, we only address the more interesting perfect case.

The DP-hardness is by a LogSpace reduction from *exact-4-colourability*, a well-known DP-complete problem [33]. In particular, a graph G is exact-4-colourable (i.e., 4-colourable and not 3-colourable) if and only if $cert^D_{q_G,J} = \{(c_4)\}$. □

Corollary 5. *Complete-VQDEF(DL-Lite$_\mathcal{R}$, GLAV, UCQ), Sound-VQDEF(DL-Lite$_\mathcal{R}$, GLAV, UCQ), and Perfect-VQDEF(DL-Lite$_\mathcal{R}$, GLAV, UCQ) are NP-complete, coNP-complete, and DP-complete, respectively.*

Finally, the lower bound proof of Theorem 4 can be easily adapted for the plain relational database case. Thus, given a schema $\mathcal{S}$, an $\mathcal{S}$-database D, a D-dataset λ, and a UCQ $q_\mathcal{S}$ over $\mathcal{S}$, it is DP-complete the problem of deciding whether $q_\mathcal{S}$ defines λ inside D (the DP membership of this problem directly follows from Corollary 3).

5 COMPUTATION

In this section, we address the computation problem. We start by considering the case when the OBDM system Σ at hand is inconsistent as a separate case. Given an inconsistent OBDM system $\Sigma = \langle J, D\rangle$ and a D-dataset λ of arity n, we point out that any query q_O over the ontology $\mathcal{O}$ of the OBDM specification J is the UCQ-minimally complete Σ-characterization of λ (recall that the certain answers of any query q_O of arity n w.r.t. an inconsistent OBDM system Σ is the set of all possible n-tuples of constants occurring in D). Furthermore, if $\lambda = dom(D)^n$, then any query q_O is also the UCQ-maximally sound (and therefore the perfect) Σ-characterization of λ; otherwise, i.e., $\lambda \subsetneq dom(D)^n$, no sound (and therefore, no UCQ-maximally sound and no perfect) Σ-characterization of λ exists.

Having thoroughly covered the case of inconsistent OBDM systems, in what follows in this section, unless otherwise stated, we implicitly assume to deal only with consistent OBDM systems.

Specifically, given a consistent OBDM system $\Sigma = \langle J, D\rangle$ and a D-dataset λ, we provide exponential time algorithms for computing UCQ-minimally complete and UCQ-maximally sound Σ-characterizations of λ, thus proving that, in this case, they always exist. As already observed in Proposition 2, in our scenario all UCQ-minimally complete (resp., UCQ-maximally sound) characterizations of λ are unique up to logical equivalence w.r.t. Σ, and therefore we refer to *the* UCQ-minimally complete (resp., UCQ-maximally sound) Σ-characterization of λ.

Before illustrating the main techniques to compute such best characterizations, we provide two crucial properties about the canonical structure that we will use to establish the correctness of our algorithms.

Proposition 4. *Let $\Sigma = \langle\langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle, D\rangle$ be an OBDM system, q_O be a UCQ over $\mathcal{O}$, and $\vec{c}$ and $\vec{b}$ be two tuples of constants such that $(C_O^{\mathcal{M}(D)}, \vec{c}) \to (C_O^{\mathcal{M}(D)}, \vec{b})$. If $\vec{c} \in cert^D_{q_O,J}$, then $\vec{b} \in cert^D_{q_O,J}$.*

Proof. If Σ is inconsistent, the claim is trivial. If Σ is consistent, from Section 2 we know that $\vec{c} \in cert^D_{q_O,J}$ implies the existence of a disjunct $q = \{\vec{x} \mid \exists \vec{y}.\phi(\vec{x}, \vec{y})\}$ in q_O for which

$(set(\phi), \vec{x}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{c})$. Let h be the homomorphism witnessing that $(set(\phi), \vec{x}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{c})$, and let h' be the homomorphism witnessing that $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{c}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$, which holds by the premises of the proposition. The composition function $h'' = h' \circ h$ is then a homomorphism witnessing that $(set(\phi), \vec{x}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$. It follows that $\vec{b} \in cert^D_{q_{\mathcal{O}},J}$, as required. □

Proposition 5. *Let $\Sigma = \langle\langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle, D\rangle$ be a consistent OBDM system, $\vec{b}$ and $\vec{c}$ be two tuples of constants, and $q_{\vec{c}}$ be the CQ $q_{\vec{c}} = query(\mathcal{M}(D), \vec{c})$. We have that $\vec{b} \in cert^D_{q_{\vec{c}},J}$ if and only if $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{c}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$.*

Proof. Suppose that $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{c}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$, and let h be the homomorphism witnessing it. Consider the query $q_{\vec{c}} = query(\mathcal{M}(D), \vec{c}) = \{\vec{x} \mid \exists \vec{y}.\phi(\vec{x}, \vec{y})\}$. Observe that $set(\phi)$ is obtained from $\mathcal{M}(D)$ by appropriately replacing each occurrence of each constant $c \in dom(\mathcal{M}(D))$ either with a distinguished variable $x_c \in \vec{x}$ or with an existential variable $y_c \in \vec{y}$. This means that h can be immediately transformed into a homomorphism witnessing that $(set(\phi), \vec{x}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$, thus implying that $\vec{b} \in cert^D_{q_{\vec{c}},J}$.

Suppose now that $\vec{b} \in cert^D_{q_{\vec{c}},J}$. Since Σ is consistent, it follows that there is a homomorphism h witnessing that $(set(\phi), \vec{x}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$, where $q_{\vec{c}} = query(\mathcal{M}(D), \vec{c}) = \{\vec{x} \mid \exists \vec{y}.\phi(\vec{x}, \vec{y})\}$. By considering again the relationship between $set(\phi)$ and $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}$, the homomorphism h can be immediately transformed into a homomorphism h' that witnesses $(\mathcal{M}(D), \vec{c}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$. It is now not hard to verify that h' can be extended into a homomorphism h'' witnessing that $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{c}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$. □

We are now ready to present our techniques. We start with the complete case, and provide the algorithm MinCompCharacterization for computing UCQ-minimally complete characterizations.

Algorithm MinCompCharacterization

Input:
OBDM system $\Sigma = \langle J, D\rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$;
D-dataset $\lambda = \{\vec{c_1}, \ldots, \vec{c_n}\}$
Output:
UCQ $q_{\mathcal{O}}$ over $\mathcal{O}$

1: Compute $\mathcal{M}(D)$
2: $q_{\mathcal{O}} \leftarrow query(\mathcal{M}(D), \vec{c_1}) \cup \ldots \cup query(\mathcal{M}(D), \vec{c_n})$
3: **return** $q_{\mathcal{O}}$

Informally, for each $\vec{c_i} \in \lambda$, the algorithm obtains from the set of atoms $\mathcal{M}(D)$ the CQ $query(\mathcal{M}(D), \vec{c_i})$. Finally, the output is the union of all such CQs.

Example 6. *Let $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ be the same OBDM specification of Example 1. One can verify that for the $\mathcal{S}$-database $D = \{s_1(c_1), s_3(c_2, b), s_3(c_3, b)\}$ and the D-dataset $\lambda = \{(c_1), (c_2)\}$, MinCompCharacterization$(\langle J, D\rangle, \lambda)$ returns the UCQ $q_{\mathcal{O}} = query(\mathcal{M}(D), (c_1)) \cup query(\mathcal{M}(D), (c_2))$, where $query(\mathcal{M}(D), (c_1)) = \{(x_{c_1}) \mid \exists y_{c_2}, y_{c_3}, y_b.Student(x_{c_1}) \wedge EnrolledIn(y_{c_2}, y_b) \wedge EnrolledIn(y_{c_3}, y_b)\}$ and $query(\mathcal{M}(D), (c_2)) = \{(x_{c_2}) \mid \exists y_{c_1}, y_{c_3}, y_b.EnrolledIn(x_{c_2}, y_b) \wedge EnrolledIn(y_{c_3}, y_b) \wedge Student(y_{c_1})\}$. Furthermore, one can see that $q_{\mathcal{O}}$ is the UCQ-minimally complete Σ-characterization of λ, where $\Sigma = \langle J, D\rangle$.*

The following theorem establishes termination and correctness of the MinCompCharacterization algorithm.

Theorem 6. *MinCompCharacterization(Σ, λ) terminates and returns the UCQ-minimally complete Σ-characterization of λ.*

Proof. Termination of the algorithm as well as completeness of the UCQ $q_{\mathcal{O}}$ returned are straightforward.

To prove that $q_{\mathcal{O}}$ is also the UCQ-minimally complete Σ-characterization of λ, it is enough to show that any query q over $\mathcal{O}$ that is a complete Σ-characterization of λ is such that $cert^D_{q_{\mathcal{O}},J} \subseteq cert^D_{q,J}$, where $\Sigma = \langle J, D\rangle$. We do this by contraposition. Let q be a UCQ for which $cert^D_{q_{\mathcal{O}},J} \not\subseteq cert^D_{q,J}$, i.e., for a tuple of constants $\vec{b}$ we have $\vec{b} \notin cert^D_{q,J}$ but $\vec{b} \in cert^D_{q_{\mathcal{O}},J}$. This latter means that $\vec{b} \in cert^D_{q_{\vec{c}},J}$ for some $q_{\vec{c}} = query(\mathcal{M}(D), \vec{c})$ with $\vec{c} \in \lambda$. By Proposition 5, one can see that $\vec{b} \in cert^D_{q_{\vec{c}},J}$ implies $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{c}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$. By Proposition 4, it follows that each UCQ q' over $\mathcal{O}$ containing tuple $\vec{c}$ in its set of certain answers w.r.t. Σ must contain also tuple $\vec{b}$ in such a set. Thus, since $\vec{b} \notin cert^D_{q,J}$, we derive that $\vec{c} \notin cert^D_{q,J}$ as well. Since $\vec{c} \in \lambda$, this latter clearly implies that q is not a complete Σ-characterization of λ, as required. □

We now turn to the sound case, and provide the algorithm MaxSoundCharacterization for computing UCQ-maximally sound Σ-characterizations.

Algorithm MaxSoundCharacterization

Input:
Consistent OBDM system $\Sigma = \langle J, D\rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$;
D-dataset $\lambda = \{\vec{c_1}, \ldots, \vec{c_m}\}$ of arity n
Output:
UCQ $q_{\mathcal{O}}$ over $\mathcal{O}$

1: $\lambda^- \leftarrow dom(D)^n \setminus \lambda$
2: $q_{\mathcal{O}} \leftarrow \{\vec{x} \mid \bot(\vec{x})\}$, where $\vec{x} = (x_1, \ldots, x_n)$
3: Compute $\mathcal{M}(D)$
4: **for** each $i \leftarrow 1, \ldots, m$ **do**
5: $\quad q_i \leftarrow query(\mathcal{M}(D), \vec{c_i})$
6: $\quad$ **if** $cert^D_{q_i,J} \cap \lambda^- = \emptyset$ **then**
7: $\quad\quad q_{\mathcal{O}} \leftarrow q_{\mathcal{O}} \cup q_i$
8: $\quad$ **end if**
9: **end for**
10: **return** $q_{\mathcal{O}}$

Intuitively, starting from the UCQ $query(\mathcal{M}(D), \vec{c_1}) \cup \ldots \cup query(\mathcal{M}(D), \vec{c_m})$, the algorithm simply discards all those disjuncts whose set of certain answers w.r.t. Σ contain a tuple $\vec{b} \notin \lambda$. We recall from Section 2 that the set of certain answers of a CQ q_i w.r.t. a consistent OBDM system $\Sigma = \langle J, D\rangle$ can be computed by first

obtaining its reformulation $\text{rew}_{q_i,J}$ over the source schema $\mathcal{S}$, and then by evaluating this latter query directly over the $\mathcal{S}$-database D.

Example 7. *Refer to Example 6. Since the certain answers of* $query(\mathcal{M}(D), (c_2))$ *w.r.t.* $\Sigma = \langle J, D\rangle$ *include also* $(c_3) \notin \lambda$, *MaxSoundCharacterization*(Σ, λ) *returns the CQ* $q_O = query(\mathcal{M}(D), (c_1))$, *which is the UCQ-maximally sound* Σ*-characterization of* λ.

The following theorem establishes termination and correctness of the MaxSoundCharacterization algorithm.

Theorem 7. *MaxSoundCharacterization*(Σ, λ) *terminates and returns the UCQ-maximally sound* Σ*-characterization of* λ.

Proof. Termination of the algorithm as well as soundness of the UCQ q_O returned are straightforward.

To prove that q_O is also the UCQ-maximally sound Σ-characterization of λ, it is enough to show that any query q over O that is a sound Σ-characterization of λ is such that $cert^D_{q,J} \subseteq cert^D_{q_O,J}$, where $\Sigma = \langle J, D\rangle$. We do this by contraposition. Let q be a UCQ for which $cert^D_{q,J} \not\subseteq cert^D_{q_O,J}$, i.e., for a tuple of constants $\vec{b}$ we have $\vec{b} \in cert^D_{q,J}$ but $\vec{b} \notin cert^D_{q_O,J}$. If $\vec{b} \notin \lambda$, then we immediately get that q is not a sound Σ-characterization of λ, and we are done. So, assume that $\vec{b} \in \lambda$. Since $\vec{b} \notin cert^D_{q_O,J}$ and $\vec{b} \in \lambda$, it is easy to see that the algorithm discarded the disjunct $q_{\vec{b}} = query(\mathcal{M}(D), \vec{b})$ (otherwise, we would trivially derive that $\vec{b} \in cert^D_{q_{\vec{b}},J}$, and thus $\vec{b} \in cert^D_{q_O,J}$, which is a contradiction to the fact that $\vec{b} \notin cert^D_{q_O,J}$). From the algorithm, one can see that the only reason $q_{\vec{b}}$ was discarded is because $\vec{g} \in cert^D_{q_{\vec{b}},J}$ for at least a tuple $\vec{g} \notin \lambda$ (i.e., $\vec{g} \in dom(D)^n \setminus \lambda$). By Proposition 5, one can see that $\vec{g} \in cert^D_{q_{\vec{b}},J}$ implies $(C_O^{\mathcal{M}(D)}, \vec{b}) \rightarrow (C_O^{\mathcal{M}(D)}, \vec{g})$. By Proposition 4, it follows that each UCQ q' over O containing tuple $\vec{b}$ in its set of certain answers w.r.t. Σ must contain also tuple $\vec{g}$ in such a set. Thus, since $\vec{b} \in cert^D_{q,J}$, we derive that $\vec{g} \in cert^D_{q,J}$ as well. Since $\vec{g} \notin \lambda$, this latter clearly implies that q is not a sound Σ-characterization of λ, as required. □

Notice that, in all the cases in which a perfect characterization exists, it is clear that both the above algorithms return the same query $query(\mathcal{M}(D), \vec{c_1}) \cup \ldots \cup query(\mathcal{M}(D), \vec{c_n})$. As a direct consequence of both Theorem 6 and Theorem 7, we get the following result.

Corollary 8. *Either the UCQ* $q_O = query(\mathcal{M}(D), \vec{c_1}) \cup \ldots \cup query(\mathcal{M}(D), \vec{c_n})$ *is a perfect* Σ*-characterization of* $\lambda = \{\vec{c_1}, \ldots, \vec{c_n}\}$, *or a perfect* Σ*-characterization of* λ *in UCQ does not exist.*

Furthermore, the combination of Corollary 8 and Proposition 5 allow us to provide a semantic test for the existence of perfect characterizations in UCQ in the OBDM case, which can be seen as the analogous of the semantic tests given in [5] and [29] for the plain relational database case and the ontology-mediated query answering case, respectively. More specifically, given a consistent OBDM system $\Sigma = \langle\langle O, \mathcal{S}, \mathcal{M}\rangle, D\rangle$ and a D-dataset λ of arity n, there exists a perfect Σ-characterization of λ in UCQ if and only if it is the case that $(C_O^{\mathcal{M}(D)}, \vec{c}) \nrightarrow (C_O^{\mathcal{M}(D)}, \vec{b})$ for each $\vec{c} \in \lambda$ and each $\vec{b} \in dom(D)^n \setminus \lambda$.

In the next section, we study the computational complexity of the problem of deciding, given $\Sigma = \langle\langle O, \mathcal{S}, \mathcal{M}\rangle, D\rangle$ and λ, whether a perfect Σ-characterization of λ exists.

6 EXISTENCE

We now address the existence problem. For the scenario under consideration in this paper, the existence problem for both UCQ-minimally complete and UCQ-maximally sound characterizations is trivial, since by Theorems 6 and 7 they always exist. So, we only consider the perfect case, by defining a variant of the QDEF problem as defined in [29], where also a mapping in some mapping language is given as input.

Problem:	**QDEF**$(\mathcal{L}_O, \mathcal{L}_\mathcal{M}, \mathcal{Q})$
Input:	An OBDM system $\Sigma = \langle\langle O, \mathcal{S}, \mathcal{M}\rangle, D\rangle$ and a D-dataset λ, where $O \in \mathcal{L}_O$ and $\mathcal{M} \in \mathcal{L}_\mathcal{M}$.
Question:	Is there a query $q_O \in \mathcal{Q}$ over O such that q_O is the perfect Σ-characterization of λ?

In what follows, we show that the computational complexity of the above QDEF decision problem differs depending on the mapping language $\mathcal{L}_\mathcal{M}$ adopted. A key difference between GLAV and the special cases GAV and LAV is in the size of $\mathcal{M}(D)$. In GLAV mappings, $\mathcal{M}(D)$ can be exponentially large due to the simultaneous presence of joins in the left-hand side, and existential variables in the right-hand side, of assertions (e.g., take $D = \{s_i(0), s_i(1) \mid 1 \leq i \leq n\}$ and $\mathcal{M}$ containing the GLAV assertion: $\{(x_1, \ldots, x_n) \mid s_1(x_1) \wedge \ldots \wedge s_n(x_n)\} \rightarrow \{(x_1, \ldots, x_n) \mid \exists y.P(x_1, y) \wedge \ldots \wedge P(x_n, y)\}$). Conversely, in both LAV and GAV mappings, $\mathcal{M}(D)$ is always polynomially bounded since the former do not allow for joins in the left-hand side of assertions, whereas the latter do not allow for existential variables in the right-hand side of assertions and the arity of ontology predicates is fixed to at most 2.

GAV and LAV mappings, however, differ for the effort in computing $\mathcal{M}(D)$. While in LAV mappings $\mathcal{M}(D)$ can be always computed in polynomial time, in GAV mappings there are CQs on the left-hand side of assertions, and so $\mathcal{M}(D)$ can not be computed in polynomial time (unless P=NP).

We start by characterizing the computational complexity of the simplest LAV case, then the GAV case, and finally the most general GLAV case. Interestingly, all the provided matching lower bounds hold even for fixed ontologies $O = \emptyset$, i.e., ontologies without assertions, fixed D-dataset λ containing a single unary tuple, and for both CQs and UCQs as query languages.

Importantly, for the scenario under consideration, due to Corollary 8, the question in QDEF can be reformulated equivalently as follows: "is $q_O = query(\mathcal{M}(D), \vec{c_1}) \cup \ldots \cup query(\mathcal{M}(D), \vec{c_n})$ also a sound (and so, a perfect) Σ-characterization of $\lambda = \{\vec{c_1}, \ldots, \vec{c_n}\}$?".

Theorem 9. *QDEF(DL-Lite*$_\mathcal{R}$*, LAV, UCQ) is coNP-complete.*

Proof. As for the membership in coNP, we can first compute $\mathcal{M}(D)$ in polynomial time, and then, exactly as illustrated in

Theorem 2, we can check in coNP whether $query(\mathcal{M}(D), \vec{c_1}) \cup \ldots \cup query(\mathcal{M}(D), \vec{c_n})$ is also a sound (and so, a perfect) Σ-characterization of $\lambda = \{\vec{c_1}, \ldots, \vec{c_n}\}$.

coNP-hardness directly follows from the plain relational database case [2]. □

Recall that the complexity class Θ_2^p has many characterizations: $\Theta_2^p = \mathrm{P}^{\mathrm{NP}[\mathrm{O}(\log n)]} = \mathrm{P}$ *with a constant number of rounds of parallel queries to an* NP *oracle* [8] (see also [38] for further characterizations).

Theorem 10. *QDEF(DL-Lite$_\mathcal{R}$, GAV, UCQ) is Θ_2^p-complete.*

Proof (Sketch.) As for the upper bound, for each pair of constants $(c_1, c_2) \in dom(D)^2$ (resp., constant $c \in dom(D)$) and for each atomic role P (resp., concept A) in the alphabet of $\mathcal{O}$ we ask, all together with a single round of parallel queries to an NP oracle, whether $P(c_1, c_2) \in \mathcal{M}(D)$ (resp., $A(c) \in \mathcal{M}(D)$). Then, with a second and final round, due to Theorem 2, we can ask with a single query to an NP oracle whether $query(\mathcal{M}(D), \vec{c_1}) \cup \ldots \cup query(\mathcal{M}(D), \vec{c_n})$ is also a sound (and so, a perfect) Σ-characterization of $\lambda = \{\vec{c_1}, \ldots, \vec{c_n}\}$.

As for the lower bound, the proof of Θ_2^p-hardness is by a LogSpace reduction from *odd clique*, which is Θ_2^p-complete [37]. *Odd clique* is the problem of deciding, given an undirected graph $G = (V, E)$ without loops, whether the maximum clique size of G is an odd number. Without loss of generality, we may assume that E contains at least an edge and that the cardinality of V is an even number (indeed, it is always possible to add fresh isolated nodes to the graph G without changing its maximum clique size).

Let $V = \{v_1, \ldots, v_n\}$, we define an OBDM system $\Sigma_G = \langle J_G, D_G \rangle$ as follows: $J_G = \langle \mathcal{O}, \mathcal{S}_G, \mathcal{M}_G \rangle$ is an OBDM specification such that $\mathcal{O} = \emptyset$, $\mathcal{S}_G = \{e, s_1, \ldots, s_n\}$, and $\mathcal{M}_G$ has the following GAV assertions, for each odd $i \in [1, n]$:

$$\{(x) \mid \exists y_1, \ldots, y_i . s_i(x) \wedge cl_i\} \rightarrow \{(x) \mid A_i(x)\}$$
$$\{(x) \mid \exists y_1, \ldots, y_{i+1} . s_{i+1}(x) \wedge cl_{i+1}\} \rightarrow \{(x) \mid A_i(x)\}$$

where A_i is an atomic concept in the alphabet of $\mathcal{O}$ and, for each $p \in [1, n]$, $cl_p = \bigwedge_{\{(k,j) \mid 1 \leq k < j \leq p\}} e(y_k, y_j)$. Intuitively, cl_p asks whether G contains a clique of size p. Finally, $D_G = \{e(x_1, x_2) \mid (x_1, x_2) \in E\} \cup \{e(x_2, x_1) \mid (x_1, x_2) \in E\} \cup \{s_i(c) \mid 1 \leq i \leq n \text{ and } i \text{ is odd}\} \cup \{s_i(c') \mid 2 \leq i \leq n \text{ and } i \text{ is even}\}$. Let, moreover, λ be the fixed D_G-dataset $\lambda = \{(c)\}$.

Notice that λ is fixed, whereas the OBDM system Σ_G can be constructed in LogSpace from an input graph G.

It can be shown that, for any odd $i \in [1, n]$, $A_i(c) \in \mathrm{cert}_\mathcal{O}^{\mathcal{M}_G(D_G)}$ (resp., $A_i(c') \in \mathrm{cert}_\mathcal{O}^{\mathcal{M}_G(D_G)}$) if and only if the graph G contains a clique of size i (resp., a clique of size $i + 1$).

With this property at hand, it is not hard to prove that the maximum clique size of a graph G is odd if and only if the CQ $q_\mathcal{O} = query(\mathcal{M}_G(D_G), c)$ is also a sound (and so, a perfect) Σ_G-characterization of λ. □

Theorem 11. *QDEF(DL-Lite$_\mathcal{R}$, GLAV, UCQ) is* coNExpTime*-complete.*

Proof (Sketch.) We start by discussing the upper bound. We show how to check whether $q_\mathcal{O} = query(\mathcal{M}(D), \vec{c_1}) \cup \ldots \cup query(\mathcal{M}(D), \vec{c_m})$ is not a sound (and so, not a perfect) Σ-characterization of $\lambda = \{\vec{c_1}, \ldots, \vec{c_m}\}$ in NExpTime.

As a first step, we compute $q_\mathcal{O} = query(\mathcal{M}(D), \vec{c_1}) \cup \ldots \cup query(\mathcal{M}(D), \vec{c_m})$ in exponential time (note that $\mathcal{M}(D)$ can be exponentially large, and so also the UCQ $q_\mathcal{O}$). Then, we can proceed similarly as in the proof of Theorem 2. We guess (*i*) a tuple of constants $\vec{c}$, and (*ii*) $q'_\mathcal{O}$, $\rho_\mathcal{O}$, $q_\mathcal{S}$, $\rho_\mathcal{M}$, and f (which now can be objects of exponential size). Finally, we check in exponential time whether (*i*) $\vec{c} \in dom(D)^n \setminus \lambda$, where n is the arity of the tuples in λ, and (*ii*) the following condition holds: $\vec{c} \in \mathrm{cert}^D_{q_\mathcal{O}, J}$ or Σ is inconsistent.

As for the lower bound, the proof of coNExpTime-hardness is by a polynomial time reduction from the *complement of the succinct clique problem*. The *succinct clique problem* is known to be NExpTime-complete [31]. Due to space limitations, we do not provide such proof here but refer the reader to [15] for details. □

7 CONCLUSIONS

We have addressed the problem of UCQ-definability in the OBDM context. To semantically characterize datasets through ontologies even in cases where perfect characterizations do not exist, we have relaxed the notion of perfecteness in terms of recall and precision. Finally, in a scenario that uses the languages commonly adopted in OBDM, we have provided a thorough complexity analysis of three natural, interesting problems associated with the framework.

There are many interesting avenues for future work. Some of them are: (*i*) extending the framework for dealing also with the *query-by-example* problem, in which two distinct λ^+ and λ^- datasets are given, and one is interested in finding perfect (resp., complete and sound, with their possible corresponding approximations) characterizations queries over the ontology, so that the certain answers of such queries capture all tuples in λ^+ and no tuple in λ^-; (*ii*) investigating the existence and the computation problems when we adopt CQ as a query language instead of UCQ; (*iii*) seeking for techniques that allow to obtain, from end users' perspectives, more intelligible queries as characterizations; and (*iv*) evaluating the techniques presented in this paper to real world settings.

ACKNOWLEDGEMENTS

This work has been partially supported by the ANR AI Chair INTENDED (ANR-19-CHIA-0014), by MIUR under the PRIN 2017 project "HOPE" (prot. 2017MMJJRE), by the EU under the H2020-EU.2.1.1 project TAILOR, grant id. 952215, and by European Research Council under the European Union's Horizon 2020 Programme through the ERC Advanced Grant WhiteMech (No. 834228).

REFERENCES

[1] Serge Abiteboul, Richard Hull, and Victor Vianu. 1995. *Foundations of Databases*. Addison Wesley Publ. Co.

[2] Timos Antonopoulos, Frank Neven, and Frédéric Servais. 2013. Definability problems for graph query languages. In *Proceedings of the Sixteenth International Conference on Database Theory (ICDT 2013)*. 141–152.

[3] Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. 2009. The *DL-Lite* Family and Relations. *Journal of Artificial Intelligence Research* 36 (2009), 1–69.

[4] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider (Eds.). 2003. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press.

[5] Pablo Barceló and Miguel Romero. 2017. The Complexity of Reverse Engineering Problems for Conjunctive Queries. In *20th International Conference on Database Theory (ICDT 2017) (Leibniz International Proceedings in Informatics (LIPIcs))*, Michael Benedikt and Giorgio Orsi (Eds.), Vol. 68. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 7:1–7:17. https://doi.org/10.4230/LIPIcs.ICDT.2017.7

[6] Angela Bonifati, Radu Ciucanu, and Slawek Staworko. 2016. Learning Join Queries from User Examples. *ACM Trans. Database Syst.* 40, 4, Article 24 (Jan. 2016), 38 pages. https://doi.org/10.1145/2818637

[7] Lorenz Bühmann, Jens Lehmann, Patrick Westphal, and Simon Bin. 2018. DL-Learner Structured Machine Learning on Semantic Web Data *(WWW '18)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 467–471. https://doi.org/10.1145/3184558.3186235

[8] Samuel R. Buss and Louise Hay. 1991. On Truth-Table Reducibility to SAT. *Information and Computation* 91, 1 (1991), 86–102.

[9] Andrea Calì, Georg Gottlob, and Michael Kifer. 2013. Taming the Infinite Chase: Query Answering under Expressive Relational Constraints. *Journal of Artificial Intelligence Research* 48 (2013), 115–174.

[10] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. 2007. Tractable Reasoning and Efficient Query Answering in Description Logics: The *DL-Lite* Family. *Journal of Automated Reasoning* 39, 3 (2007), 385–429.

[11] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. 2012. Query Processing under GLAV Mappings for Relational and Graph Databases. *Proceedings of the Very Large Database Endowment* 6, 2 (2012), 61–72.

[12] Gianluca Cima. 2017. Preliminary Results on Ontology-based Open Data Publishing. In *Proceedings of the Thirtieth International Workshop on Description Logics (DL 2017) (CEUR Electronic Workshop Proceedings, http://ceur-ws.org/)*, Vol. 1879.

[13] Gianluca Cima. 2020. *Abstraction in Ontology-based Data Management*. Ph.D. Dissertation. Sapienza University of Rome.

[14] Gianluca Cima, Marco Console, Maurizio Lenzerini, and Antonella Poggi. 2021. Abstraction in Data Integration. In *Proceedings of the Thirty-Sixth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2021)*. IEEE, 1–11.

[15] Gianluca Cima, Federico Croce, and Maurizio Lenzerini. 2021. *QDEF and Its Approximations in OBDM*. CoRR. arXiv.org e-Print archive.

[16] Gianluca Cima, Maurizio Lenzerini, and Antonella Poggi. 2019. Semantic Characterization of Data Services through Ontologies. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI 2019)*. 1647–1653.

[17] Gianluca Cima, Maurizio Lenzerini, and Antonella Poggi. 2020. Non-Monotonic Ontology-based Abstractions of Data Services. In *Proceedings of the Seventeenth International Conference on Principles of Knowledge Representation and Reasoning (KR 2020)*. 243–252.

[18] AnHai Doan, Alon Y. Halevy, and Zachary G. Ives. 2012. *Principles of Data Integration*. Morgan Kaufmann.

[19] Ronald Fagin, Phokion G. Kolaitis, Renée J. Miller, and Lucian Popa. 2005. Data Exchange: Semantics and Query Answering. *Theoretical Computer Science* 336, 1 (2005), 89–124.

[20] Nicola Fanizzi, Giuseppe Rizzo, Claudia d'Amato, and Francesca Esposito. 2018. DLFoil: Class Expression Learning Revisited. In *Proceedings of the Twenty-First International Conference on Knowledge Engineering and Knowledge Management (EKAW 2018)*.

[21] Marc Friedman, Alon Levy, and Todd Millstein. 1999. Navigational Plans for Data Integration. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI 1999)*. AAAI Press, 67–73.

[22] Víctor Gutiérrez-Basulto, Jean Christoph Jung, and Leif Sabellek. 2018. Reverse Engineering Queries in Ontology-Enriched Systems: The Case of Expressive Horn Description Logic Ontologies. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, 1847–1853. https://doi.org/10.24963/ijcai.2018/255

[23] Maurizio Lenzerini. 2002. Data Integration: A Theoretical Perspective.. In *Proceedings of the Twentyfirst ACM SIGACT SIGMOD SIGART Symposium on Principles of Database Systems (PODS 2002)*. 233–246.

[24] Maurizio Lenzerini. 2011. Ontology-based Data Management. In *Proceedings of the Twentieth International Conference on Information and Knowledge Management (CIKM 2011)*. 5–6. https://doi.org/10.1145/2063576.2063582

[25] Carsten Lutz, Johannes Marti, and Leif Sabellek. 2018. Query Expressibility and Verification in Ontology-based Data Access. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference (KR 2018)*. 389–398.

[26] Denis Mayr Lima Martins. 2019. Reverse engineering database queries from examples: State-of-the-art, challenges, and research opportunities. *Information Systems* 83 (2019), 89–100. https://doi.org/10.1016/j.is.2019.03.002

[27] Boris Motik, Bernardo Cuenca Grau, Ian Horrocks, Zhe Wu, Achille Fokoue, and Carsten Lutz. 2012. *OWL 2 Web Ontology Language Profiles (Second Edition)*. W3C Recommendation. World Wide Web Consortium. Available at http://www.w3.org/TR/owl2-profiles/.

[28] Davide Mottin, Matteo Lissandrini, Yannis Velegrakis, and Themis Palpanas. 2017. New Trends on Exploratory Methods for Data Analytics. *Proc. VLDB Endow.* 10, 12 (Aug. 2017), 1977–1980.

[29] Magdalena Ortiz. 2019. Ontology-Mediated Queries from Examples: a Glimpse at the DL-Lite Case. In *Proceedings of the Fifth Global Conference on Artificial Intelligence (EPiC Series in Computing)*, Vol. 65. 1–14.

[30] Christos H. Papadimitriou and Mihalis Yannakakis. 1984. The Complexity of Facets (and Some Facets of Complexity). *J. Comput. System Sci.* 28, 2 (1984), 244–259.

[31] Christos H. Papadimitriou and Mihalis Yannakakis. 1986. A Note on Succinct Representations of Graphs. *Information and Computation* 71, 3 (1986), 181–185.

[32] Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. 2008. Linking Data to Ontologies. *Journal on Data Semantics* X (2008), 133–173. https://doi.org/10.1007/978-3-540-77688-8_5

[33] Jörg Rothe. 2003. Exact complexity of Exact-Four-Colorability. *Inform. Process. Lett.* 87, 1 (2003), 7–12.

[34] Umberto Straccia and Matteo Mucci. 2015. pFOIL-DL: Learning (Fuzzy) EL Concept Descriptions from Crisp OWL Data Using a Probabilistic Ensemble Estimation *(SAC '15)*. ACM, New York, NY, USA, 345–352. https://doi.org/10.1145/2695664.2695707

[35] Balder ten Cate and Víctor Dalmau. 2015. The Product Homomorphism Problem and Applications. In *Proceedings of the Eigthteenth International Conference on Database Theory (ICDT 2015) (LIPIcs)*, Vol. 31. 161–176.

[36] Quoc Trung Tran, Chee-Yong Chan, and Srinivasan Parthasarathy. 2014. Query Reverse Engineering. *The VLDB Journal* 23, 5 (Oct. 2014), 721–746. https://doi.org/10.1007/s00778-013-0349-3

[37] Klaus W. Wagner. 1987. More Complicated Questions About Maxima and Minima, and Some Closures of NP. *Theoretical Computer Science* 51 (1987), 53–80.

[38] Klaus W. Wagner. 1990. Bounded Query Classes. *SIAM J. Comput.* 19, 5 (1990), 833–846.

[39] Moshé M. Zloof. 1975. Query-by-example: The Invocation and Definition of Tables and Forms *(VLDB '75)*. ACM, New York, NY, USA, 1–24. https://doi.org/10.1145/1282480.1282482