Semantic Web -1 (2023) 1–36
DOI 10.3233/SW-233391
IOS Press

Separability and Its Approximations in Ontology-based Data Management

Gianluca Cima *, Federico Croce and Maurizio Lenzerini
Sapienza University of Rome, Italy
E-mails: cima@diag.uniroma1.it, croce@diag.uniroma1.it, lenzerini@diag.uniroma1.it

Editors: Roberto Confalonieri, University of Padua, Italy; Oliver Kutz, Free University of Bozen-Bolzano, Italy; Diego Calvanese, Umeå University, Sweden and Free University of Bozen-Bolzano, Italy; Jose M. Alonso, University of Santiago de Compostela, CiTIUS, Spain; Shang-Ming Zhou, University of Plymouth, UK
Solicited reviews: Jean Christoph Jung, TU Dortmund University, Germany; four anonymous reviewers

Abstract. Given two datasets, i.e., two sets of tuples of constants, representing positive and negative examples, logical separability is the reasoning task of finding a formula in a certain target query language that separates them. As already pointed out in previous works, this task turns out to be relevant in several application scenarios such as concept learning and generating referring expressions. Besides, if we think of the input datasets of positive and negative examples as composed of tuples of constants classified, respectively, positively and negatively by a black-box model, then the separating formula can be used to provide global post-hoc explanations of such a model. In this paper, we study the separability task in the context of Ontology-based Data Management (OBDM), in which a domain ontology provides a high-level, logic-based specification of a domain of interest, semantically linked through suitable mapping assertions to the data source layer of an information system. Since a formula that properly separates (proper separation) two input datasets does not always exist, our first contribution is to propose (best) approximations of the proper separation, called (minimally) complete and (maximally) sound separations. We do this by presenting a general framework for separability in OBDM. Then, in a scenario that uses by far the most popular languages for the OBDM paradigm, our second contribution is a comprehensive study of three natural computational problems associated with the framework, namely Verification (check whether a given formula is a proper, complete, or sound separation of two given datasets), Existence (check whether a proper, or best approximated separation of two given datasets exists at all), and Computation (compute any proper, or any best approximated separation of two given datasets).

Keywords: Ontology-based Data Management, Separability, Explainable Artificial Intelligence, Semantic Technologies

1. Introduction

The separability problem deals with finding an intensional representation of two datasets, i.e., sets of data items, interpreted as positive and negative examples. In this problem, one is given two sets of data items, one with positive and the other with negative examples, and is asked to provide a query so that the evaluation of such a query over the database contains all the data items in the set of positive examples, and none of the data items in the set of negative examples. We say that a solution to this problem is a query that separates the given datasets. A special case of this problem arises when only one set of positive examples is given as input, and one is interested in finding a query

*Corresponding author. E-mail: cima@diag.uniroma1.it. All authors have contributed equally.

whose evaluation over the database coincides with the data items in such a set. In this paper, we refer to the latter special case with the term characterizability, and we say that a solution to this problem is a query that characterizes the given dataset.

The separability problem has initially been studied for relational databases and is known in the community as the query-by-example problem.[1] Over the years, researchers have found several interesting applications of the separability problem, spanning from simplifying query formulation by non-experts, to debugging facilities for data engineers. Indeed, the problem has been studied as a useful tool for data exploration, concept learning, data analysis, usability, data security and more [51,54]. Moreover, as already observed in [37], the problem is studied in two special cases in which the input datasets are constituted by only one single tuple. In separability, this special case is studied for entity comparison in RDF graphs, where the goal is to find a meaningful description that separates one entity from another. Similarly, in characterizability, this special case is studied for generating referring expressions (GRE), where one is interested in describing a single data item by a logical expression that allows to separate it from all other data items. With the rise of Machine Learning (ML), we argue that this topic acquires primary importance for providing meaningful explanations to any typical supervised black-box model used for classification tasks. When applied to classification, the ultimate goal of supervised learning is to construct models that are able to predict the target output (i.e., the class) of the proposed inputs. To achieve this, the learning algorithm is provided with some training examples that demonstrate the intended relation of input and output values. Then, the learned model is supposed to be able to correctly classify instances that have not been shown during training. A crucial problem for wise and safe adoption of ML-based black-box models is that, especially in high-risk domains such as healthcare and finance, it is often very hard to understand the rationale behind a classification made by these models. This may lead to discriminatory biases in the classification that were not intended and, more surprisingly, of which the designers were unaware of.

In this paper, we assume that the classification task is performed in an organization that adopts an Ontology-based Data Management (OBDM)[2] approach [46,48]. OBDM is a paradigm for accessing data using a conceptual representation of the domain of interest expressed as an ontology. The OBDM paradigm relies on a three-level architecture, consisting of the data layer, the ontology, and the mapping between the two. Consequently, an OBDM specification is a triple $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ which, together with an $\mathcal{S}$-database D, form a so-called OBDM system $\Sigma = \langle J, D \rangle$. We are going to tackle the separability problem 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 two D-datasets λ^+ and λ^-, our goal is to derive a query expression over $\mathcal{O}$ that *separates* λ^+ and λ^- in Σ (called here *proper separation*).

Example 1. Let $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ be the OBDM specification in which $\mathcal{O} = \{\mathsf{MathStudent} \sqsubseteq \mathsf{Student}, \mathsf{ForeignStudent} \sqsubseteq \mathsf{Student}\}$ asserts that both math students and foreign students are students, $\mathcal{S} = \{s_1, s_2, s_3, s_4, s_5\}$, and the mapping $\mathcal{M}$ consists of the following assertions:

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

Consider now the OBDM system $\Sigma = \langle J, D \rangle$, where J is the OBDM specification illustrated above and D is the $\mathcal{S}$-database $D = \{s_1(c_4), s_2(c_3), s_4(b_1), s_5(d_1), s_3(c_1, b_1), s_3(c_2, d_1), s_3(c_3, e_1), s_3(c_4, e_2), s_3(c_5, e_3)\}$. Let the D-datasets of positive and negative examples be $\lambda_1^+ = \{(c_1), (c_2), (c_3)\}$ and $\lambda_1^- = \{(c_5)\}$, respectively. One can see

[1] On the other hand, the characterizability problem is known in the literature as query definability.

[2] In this paper, we prefer the usage of the acronym OBDM rather than its similar *OBDA*, which stands for *Ontology-based Data Access* [58], because data access is just one aspect, although one of the most important, of the more general notion of data management.

that the query $q^1_{\mathcal{O}} = \{(x) \mid \mathsf{Student}(x)\}$ over $\mathcal{O}$ separates λ^+_1 and λ^-_1 in Σ because its set of certain answers over Σ, $\{(c_1), (c_2), (c_3), (c_4)\}$, contains all the positive examples in λ^+ and none of the negative examples in λ^-.

An important contribution of our work is to provide approximated results for all the cases in which it is not possible to provide a separating query. We argue that, in these cases, reasonable and useful ontological characterizations can still be provided. We propose to resort to suitable approximations of the proper separating query, by introducing the notions of sound and complete separating queries. The former is a query whose certain answers have empty intersection with the D-dataset λ^-, whereas the certain answers of the latter form a superset of the D-dataset λ^+. Obviously, we are interested in computing the best approximated separating queries, which we call maximally sound and minimally complete separations, respectively. A maximally sound (resp., minimally complete) separation is a sound (resp., complete) separation such that no other sound (resp., complete) separation exists that better approximates the input datasets. Moreover, we cover the special cases in which the input datasets are constituted by only one single tuple for all our results and refer to them as the *single tuple* variants of the problems we deal with.

In this context, the training set used in the classification task, which is a collection of data items that are labeled as positive and negative examples, is seen as two sets of tuples in the database schema. The query derived by solving the separability problem results into an intensional definition of such training set, and are considered as an explanation of the intensional properties of the training set. The same principle can also be applied to a set of tuples that has not been seen during the training of the model. In this scenario, one can consider the black-box model as an oracle that assigns a class to all given tuples. Then, the query derived by solving the separability problem in this new context, is considered as an explanation of the intrinsic behaviour of the model. Traditionally, there are two different types of explanations: *global* and *local*. We refer to the former for explanations of the general behaviour of the model, and to the latter for explanations of the output of the model with respect to a specific object. The present work poses the foundational basis for providing both kind of explanations. Indeed, it deals with global explanations when the separating query is searched with respect to positive and negative examples containing and arbitrary number of tuples. It deals with local explanations when the sets of positive and negative examples for which one searches for a separating query contain only one single tuple.

Our procedure fits into the definition of *post-hoc* explanations of black-box models, i.e. a set of techniques aimed at approximating the behaviour of a black-box model with a surrogate interpretable model. We are now going to describe how in our context the role of the surrogate model is played by the query resulting from the solution of the separability problem. Suppose an organization that adopts the OBDM paradigm wants to train a model for predicting which candidates in a selection process are the most likely to perform well in a certain job. For training the model, the organization is given the curricula of current employees with a feedback on their performances that makes it possible to divide the training set in two different classes: the good and bad performers. For the sake of simplicity, suppose John, Mary, and Jane, who studied Biology, Medicine, and Math respectively, perform well. Suppose also that Matt, Angeline, and Jess, who studied Music, Linguistics, and Fashion respectively, perform badly. The ML-based black-box model is trained with this dataset and it is optimised to reach the highest possible accuracy. Now suppose some candidates apply for the job and are evaluated by the model. The latter states that Lucy, Mara, and George, who studied Math, Chemistry, and Physics respectively, have high probability to perform well in doing the job. At the same time, the model states that Lucas, and Paul, who studied Art, and Classics are believed to perform badly. The organization now wonders why the black-box model divided the candidates in this way. By instantiating a separability problem with the two sets of positive and negative candidates, the organization finds out that, no matter how sophisticated the internal details of the black-box model are, the resulting classification is so that all positive candidates are answers to the query *"return all the candidates with a scientific background"*, and none of the negative candidates are. This separating query provides an intuitive explanation of the actual behaviour of the model. Of course, there are in general many valid separating queries for a given instance of a separability problem, as it is possible that in many cases there is no valid separating query at all.

Contributions of the paper The contribution provided by this paper can be summarized as follows:

– We present a formal framework for separability in OBDM. In particular, we first cast the classical notion of separating query in the OBDM context (called here *proper separation*), and then we propose the relaxations mentioned above (*complete separation* and *sound separation*) as well as their optimal versions (*minimally*

complete separation and *maximally sound separation*). We do exactly the same for the special case of characterization, which deals only with a dataset of positive examples rather than both positive and negative examples.

- We study the Verification problem for both separability and characterizability in OBDM, i.e. check whether a given query is a proper, complete, or sound separation (resp., characterization) of two given datasets (resp., a given dataset). More specifically, we introduce three families of decision problems for both separability and characterizability, called X-*V*SEP($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$) (resp., X-*V*CHAR($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$)) for X = {Proper, Complete, Sound}, which are parametric with respect to an ontology language $\mathcal{L}_\mathcal{O}$, a mapping language $\mathcal{L}_\mathcal{M}$, and a query language $\mathcal{Q}$. We provide tight computational complexity bounds for the most common languages used in OBDM, i.e., $\mathcal{L}_\mathcal{O}$ is *DL-Lite*$_\mathcal{R}$, $\mathcal{L}_\mathcal{M}$ is GLAV, and $\mathcal{Q}$ is UCQ. The results are summarized in Table 1.
- We study the Computation problem. We provide two algorithms that, taking as input an OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$ and two D-datasets λ^+ and λ^-, where $\mathcal{O}$ is a *DL-Lite*$_\mathcal{R}$ ontology and $\mathcal{M}$ is a GLAV mapping, return, respectively, a UCQ-minimally complete Σ-separation of λ^+ and λ^- and a UCQ-maximally sound Σ-separation of λ^+ and λ^-. As a consequence, this proves that in the scenario under consideration the two best approximated versions of proper separations always exist.
- We study the Existence problem for both separability and characterizability in OBDM. Since for the scenario under consideration their two best approximated versions always exist, we only focus on the existence of a proper separation (resp., characterization), i.e., check whether a proper separation (resp., characterization) in a target query language $\mathcal{Q}$ of two given datasets (resp., a given dataset) exists at all. More specifically, we introduce a family of decision problems for both separability and characterizability, called SEP($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$) (resp., CHAR($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$)), which are parametric with respect to an ontology language $\mathcal{L}_\mathcal{O}$, a mapping language $\mathcal{L}_\mathcal{M}$, and a query language $\mathcal{Q}$. Also in this case, we provide tight computational complexity bounds for the most common languages used in OBDM. The results are summarized in Table 2.

Table 1

For X = {Proper, Complete, Sound}, the table reports the exact computational complexity of X-*V*SEP($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$) and X-*V*CHAR($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$) when $\mathcal{L}_\mathcal{O}$ is *DL-Lite*$_\mathcal{R}$, $\mathcal{L}_\mathcal{M}$ is GLAV, and $\mathcal{Q}$ is UCQ. We point out that all the lower bounds already hold for the single-tuple version of the considered problem (i.e., when all the input datasets consist of a single tuple) and when $\mathcal{L}_\mathcal{O}$ is $\emptyset$ (i.e., the ontology language allowing only for ontologies without assertions), $\mathcal{L}_\mathcal{M}$ is GAV∩LAV (i.e., the mapping language allowing only for assertions that are both GAV and LAV), and $\mathcal{Q}$ is CQ

X-*V*SEP(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ)/X-*V*CHAR(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ)	
X = Complete	NP-complete
X = Sound	coNP-complete
X = Proper	DP-complete

Table 2

For $\mathcal{L}_\mathcal{M}$ = {LAV, GAV, GLAV}, the table reports the exact computational complexity of SEP($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$) and CHAR($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$) when $\mathcal{L}_\mathcal{O}$ is *DL-Lite*$_\mathcal{R}$ and $\mathcal{Q}$ is UCQ. Again, all the lower bounds already hold for the single-tuple version of the considered problem and when $\mathcal{L}_\mathcal{O}$ is $\emptyset$. Moreover, such lower bounds hold for both $\mathcal{Q}$ = CQ and $\mathcal{Q}$ = UCQ

SEP(*DL-Lite*$_\mathcal{R}$, $\mathcal{L}_\mathcal{M}$, UCQ)/CHAR(*DL-Lite*$_\mathcal{R}$, $\mathcal{L}_\mathcal{M}$, UCQ)	
$\mathcal{L}_\mathcal{M}$ = LAV	coNP-complete
$\mathcal{L}_\mathcal{M}$ = GAV	Θ_2^p-complete
$\mathcal{L}_\mathcal{M}$ = GLAV	Π_3^p-complete

This paper is an extended version of our CIKM'21 conference paper [22]. We point out that the conference paper focused only on the characterizability reasoning task, whereas here we illustrate a formal framework and study the related computational problems both for separability and characterizability. Furthermore, in this extended version we provide all the full proofs that were only sketched in [22]. Finally, we observe that the CHAR(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ) decision problem is incorrectly claimed to be coNEXPTIME-complete in the conference version of this paper. In this extended version, we fix this error and show that this decision problem is actually Π_3^p-complete.

To the best of our knowledge, the present work is the first to introduce separability and characterizability in the OBDM context. In particular, while separability and characterizability have been studied in various ontology-enriched query answering settings, this is the first to take into account all the layers stack of the OBDM paradigm, thus including also the mapping layer, and also propose and study natural relaxations of the separability and characterizability notions. Nevertheless, we remark that both in the Computation and in the Existence problems for separability and characterizability we mainly focus only on UCQ as a query language, while other works on the subject also investigate the typically more challenging scenario of CQ as a query language (see, e.g., [7,30,34,64]).

Finally, we mention that, since *DL-Lite*$_{\mathcal{R}}$ is insensitive to the adoption of the *unique name assumption (UNA)* for UCQ answering [5], all our results hold both with and without the UNA.

Outline of the paper The paper is organized as follows. After the discussion of related works in Section 2, Section 3 introduces the relevant background for our study and Section 4 illustrates a formal framework for separation in OBDM. Then, Sections 5, 6, and 7 present the results on the three computational problems Verification, Computation, and Existence, respectively. We deal first with the Computation problem and then with the Existence problem because the solutions we will provide about the Computation problem will help us to solve some tasks about the Existence problem. Finally, Section 8 concludes the paper with a perspective on future work.

2. Related work

Query definability and query-by-example have long been studied for classical databases, starting from [70] up to many recent studies [3,7,41,52,64,65,68]. We have laid the foundations for analysing the complexity of our decision problems from the results in [3,7]. We found the work in [65] inspirational since they dealt with the problem of deriving a metric for establishing how close is a non proper separating query to the optimal solution. We found the survey in [52] important for the whole research problem as they present a well described motivating example and they outline the main open problems of this topic.

Our work has also been inspired by the notion of *Abstraction* [19,20,23]. Although the goal is still to derive a query expression over the ontology, in that work the input is a query over the data layer of the OBDM architecture, whereas in query definability (resp. in query-by-example) the input is a set of tuples (or two sets of tuples). It follows that the two tasks are completely different and require different technical solutions. We postpone a detailed comparison between Abstraction and the separability notion investigated in this paper in Section 4.1 (when we will have all the technical tools in hand to provide an in-depth comparison).

The works that are closer to ours are the ones in [4,34,55]. In [4] the authors study the existence, and verification problems both for query definability and for query-by-example, and computation problem for the query definability case. In their work, the ontology is expressed as an RDF graph and they consider several fragments of SPARQL as the query language to be used for the separation of the examples. Differently from the present work, they do not aim at finding the best approximated separation. We share with [55] the expressive power of the language used for the ontology (DL-Lite), and for the separation query the input examples (UCQs). However, the work in [55] does not deal with the cases in which proper separations for the input examples do not exist, and it is based on a slightly different framework from ours, i.e. since in that work they study the problems in the context of ontology-mediated queries, they do not have the mapping layer that instead is part of our more general OBDM paradigm.

The work in [34], studies the query-by-example problem for expressive horn description logic ontologies, namely Horn-$\mathcal{ALCI}$. Apart from the clear difference in the expressive power of the ontologies, their work does not consider the case of approximated separations of the input examples, and it does not consider the mapping layer.

This work has also been inspired by [11,31,38,44]. All these papers' goal is to learn a concept expression that best captures a given set of examples (or two set of positive and negative examples for query-by-example). We differ from these works because our goal is to derive a full-blown query that separates the input examples.

The problem of checking whether there exists a formula separating positive and negative examples in the presence of an ontology has recently been studied in [37,40]. Other than verifying that the ontology entails the searched formula for all positive examples, the authors conducted an in-depth analysis of the so-called separability problem accounting for both weak separability, i.e. the one we study in this paper, and strong separability, i.e. checking

whether the negation of the separating formula is entailed by the KB. They also consider the case of enriching the separating formula by adding helper symbols that are not originally present in the ontology, and study the complexity of the decision problems for a wide range of languages both for expressing the ontology and the separating formula. In this paper, we are interested in studying the weak separability problem in the context of the OBDM by also considering the cases in which the separating formula does not exist and one wants to search for sound and complete approximations of it.

Another important line of research related to the present work is the one regarding post-hoc explanations of opaque machine learning models. As also highlighted by other works [25,26,49,60], the query-by-example problem can easily be adapted for explaining the output of a black-box machine learning classification model. For example, consider the case of a binary classifier labelling a set of examples in two classes 1 and 0. In this scenario, the solution of the query-by-example problem is considered as a surrogate of the machine learning model, so that the examples labelled by class 1 are the answers of the reverse engineered query, while none of the examples labelled by class 0 are. Therefore the query acts as an explanation because it provides a more human understandable way, especially in our framework in which the query is based on the knowledge of the ontology, for classifying the given examples in the two different classes. Although relevant for our work, we differ from all the above cited papers. In [60], they map the inputs of the machine learning classifier to the ontology and then uses a concept learning tool to find a class expression over the ontology that best describes the positive example. In [25], for explaining the behaviour of a black-box classifier, they build another black-box classifier (a neural network) and then project the output of this latter model onto a so-called rule space, where each coordinate represents the activation of a rule that is described in First Order Logic. In [26] they present the TREPAN algorithm, i.e. a way for building a decision tree in which the nodes are linked to an ontology, that is used as a means for explaining the input positive and negative examples. We consider the work in [49] to be relevant for our work, even though is rather preliminary and does not specify many important details such as the expressive power of the language of the ontology, and the language of the query they search for. The biggest differences with the present work are the fact that they do not consider the mapping layer between the ontology and the data, and that they do not focus on the concepts of maximally-sound and minimally-complete solutions, in cases where a perfect solution does not exist. On the contrary, they define a best approximated query as the one minimising the jaccard distance between the answers of the query and the set of positive examples in the input.

Inductive Logic Programming (ILP) [43] has long been considered related to the query definability and query-by-example tasks. We also considered it inspiring for our work, but we soon acknowledged that the expressive power of the languages used for representing the knowledge base are incomparable, and that in ILP they are interested in searching for explanations of a set of logical facts rather than a set of tuples.

The Active Learning task initially introduced by [2] has been studied as a possible framework for learning queries from examples in relational databases [64] and in the presence of *DL-Lite* ontologies [30]. Moreover, instances of the framework in [2] can be found in [42] and in [57]. In particular, the goal of [42] is to learn an ontology that is equivalent to a target ontology, while the goal of [57] is to learn an ontology that is *query inseparable* with respect to a target ontology $\mathcal{T}$ and a query language $\mathcal{Q}$, i.e. given a set of ground atoms (ABox) $\mathcal{A}$, the learned ontology $\mathcal{H}$ must be such that $\langle \mathcal{H}, \mathcal{A} \rangle \models q$ if and only if $\langle \mathcal{T}, \mathcal{A} \rangle \models q$, for every $q \in \mathcal{Q}$.

Finally, query inseparability has been studied even outside the learning task. For example, [9] studies query inseparability for (fragments of) Horn-$\mathcal{ALC}$ as ontology language and CQ as a query language, whereas [10] studies query inseparability for both the ontology languages $\mathcal{ALC}$ and $\mathcal{EL}$ and for (fragments of) UCQ as a query language.

3. Preliminaries

We recall some notions about relational databases [1], Description Logics (DLs) [6], and Ontology-based Data Management (OBDM) [47]. We define $\Gamma_{\mathcal{S}}$, $\Gamma_{\mathcal{O}}$, Const, and $\mathcal{V}$ to be the pairwise disjoint, countably infinite sets of symbols for *database predicates*, *ontology predicates*, *constants*, and *variables*, respectively. We further assume that $\Gamma_{\mathcal{O}}$ is partitioned into the disjoint sets Γ_A and Γ_P for *atomic concepts* and *atomic roles*, respectively.

Databases, datasets, and queries A *relational database schema* (or simply *schema*) $\mathcal{S}$ is a finite subset of $\Gamma_{\mathcal{S}}$. Given a schema $\mathcal{S}$, an *$\mathcal{S}$-database* D is a finite set of *facts* (a.k.a. *ground atoms*) of the form $s(\vec{c})$, where s is an n-ary predicate in $\mathcal{S}$, and $\vec{c} = (c_1, \ldots, c_n)$ is an n-tuple of constants from Const. We denote by $dom(D)$ the finite subset of Const of those constants occurring in D. Given a schema $\mathcal{S}$ and an $\mathcal{S}$-database D, a *D-dataset* λ of arity n is simply 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 from $\mathcal{V}$, 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 $\mathcal{S}$-database D is a set of *answers* $q_{\mathcal{S}}^D$, each answer being an n-tuple of constants occurring in $dom(D)$, i.e., $q_{\mathcal{S}}^D \subseteq dom(D)^n$. A query $q_{\mathcal{S}}$ of arity 0 over a schema $\mathcal{S}$ is called a *boolean* query, and we denote by $q_{\mathcal{S}}^D = \{\langle\rangle\}$ (resp., $q_{\mathcal{S}}^D = \emptyset$) the fact that $D \models q_{\mathcal{S}}$ (resp., $D \not\models q_{\mathcal{S}}$).

Following the terminology of [7,63], we say that a query $q_{\mathcal{S}}$ over a schema $\mathcal{S}$ *explains two D-datasets* λ^+ *and* λ^- (resp., *defines a D-dataset* λ^+) *inside an $\mathcal{S}$-database* D if $\lambda^+ \subseteq q_{\mathcal{S}}^D$ and $q_{\mathcal{S}}^D \cap \lambda^- = \emptyset$ (resp., $q_{\mathcal{S}}^D = \lambda^+$). We also say that λ^+ and λ^- are *$\mathcal{Q}$-explainable* (resp., 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 explains λ^+ and λ^- (resp., defines λ^+) inside D.

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* from $\mathcal{V}$, where n is the arity of $q_{\mathcal{S}}$ (ii) $\vec{y} = (y_1, \ldots, y_m)$ is an m-tuple of *existential variables* from $\mathcal{V}$; 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 an p-ary predicate in $\mathcal{S}$ and v_i is either a distinguished or an existential variable, i.e., $v_i \in \vec{x} \cup \vec{y}$ for each $i \in [1, p]$. A *union of conjunctive queries (UCQ)* $q_{\mathcal{S}}$ over a schema $\mathcal{S}$ is a finite set $\{\vec{x_1} \mid \exists \vec{y_1}.\phi_1(\vec{x_1}, \vec{y_1})\} \cup \cdots \cup \{\vec{x_m} \mid \exists \vec{y_m}.\phi_m(\vec{x_m}, \vec{y_m})\}$ of CQs over $\mathcal{S}$ 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 $\mathcal{C}$ and a tuple $\vec{c} = (c_1, \ldots, c_n)$ of constants, we denote by $query(\mathcal{C}, \vec{c})$ 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 $\mathcal{C}'$, where $\mathcal{C}'$ is obtained from $\mathcal{C}$ 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 $\mathcal{C}'$ that do not occur in $\vec{x}$.

Given a set of atoms $\mathcal{C}$, we denote by $dom(\mathcal{C})$ the set of all constants and variables occurring in $\mathcal{C}$. Observe that $dom(\mathcal{C}) \subseteq \mathsf{Const} \cup \mathcal{V}$. Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two sets of atoms. We say that a function $h : dom(\mathcal{C}_1) \to dom(\mathcal{C}_2)$ is a *homomorphism* from $\mathcal{C}_1$ to $\mathcal{C}_2$ if $h(\mathcal{C}_1) \subseteq \mathcal{C}_2$, where $h(\mathcal{C}_1)$ is the image of $\mathcal{C}_1$ under h, i.e., $h(\mathcal{C}_1) = \{h(\alpha) \mid \alpha \in \mathcal{C}_1\}$ 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 $\mathcal{C}_1$ and $\mathcal{C}_2$ and two tuples of terms $\vec{t_1}$ and $\vec{t_2}$, we write $(\mathcal{C}_1, \vec{t_1}) \to (\mathcal{C}_2, \vec{t_2})$ if there is a function h from $dom(\mathcal{C}_1) \cup \vec{t_1}$ to $dom(\mathcal{C}_2) \cup \vec{t_2}$ such that (i) h is a homomorphism from $\mathcal{C}_1$ to $\mathcal{C}_2$, and (ii) $h(\vec{t_1}) = \vec{t_2}$ (where, for a tuple of terms $\vec{t} = (t_1, \ldots, t_n)$, $h(\vec{t}) = (h(t_1), \ldots, h(t_n))$), $(\mathcal{C}_1, \vec{t_1}) \not\to (\mathcal{C}_2, \vec{t_2})$ otherwise.

We define the semantics of (U)CQs in terms of homomorphisms. 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, we let $q_{\mathcal{S}}^D$ to be the set of n-tuples $\vec{c}$ of constants occurring in D for which $(set(\phi), \vec{x}) \to (D, \vec{c})$. Furthermore, for an $\mathcal{S}$-database D and a UCQ $q_{\mathcal{S}} = q_1 \cup \cdots \cup q_m$ over $\mathcal{S}$, we let $q_{\mathcal{S}}^D = q_1^D \cup \cdots \cup q_m^D$.

Syntax and semantics of DL-Lite$_{\mathcal{R}}$ In this paper, a *DL ontology* (or simply *ontology*) $\mathcal{O}$ is a TBox ("Terminological Box") expressed in a specific DL, that is, a finite set of assertions stating general properties of atomic concepts and roles built according to the syntax of the specific DL, which represents the intensional knowledge of a modeled domain. The *alphabet* of an ontology $\mathcal{O}$ is the finite subset of $\Gamma_{\mathcal{O}}$ of atomic concepts and atomic roles mentioned in the assertions of $\mathcal{O}$, and we assume that every ontology $\mathcal{O}$ comprises the special *bottom concept* $\bot$ in its alphabet. In this paper, whenever we speak of a query $q_{\mathcal{O}}$ over an ontology $\mathcal{O}$, we implicitly mean a query in a certain query language $\mathcal{Q}$ that uses the atomic concepts and the atomic roles in the alphabet of $\mathcal{O}$ as predicate symbols.

The semantics of DL ontologies is specified through the notion of first-order interpretations: an *interpretation* $\mathcal{I}$ 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 objects, and the *interpretation function* $\cdot^{\mathcal{I}}$ assigns to each constant $c \in \mathsf{Const}$ an object $c^{\mathcal{I}} \in \Delta^{\mathcal{I}}$, to each atomic concept $A \in \Gamma_A$ a subset $A^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$ (with the requirement $\bot^{\mathcal{I}} = \emptyset$), and to each atomic role $P \in \Gamma_P$ a subset $P^{\mathcal{I}}$ of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. We say that an interpretation $\mathcal{I}$ satisfies an ontology $\mathcal{O}$, denoted by $\mathcal{I} \models \mathcal{O}$, if $\mathcal{I}$ satisfies every assertion in $\mathcal{O}$. When

convenient, with a slight abuse of notation, we treat an interpretation $\mathcal{I}$ as the (potentially infinite) set of facts of the form $A(o)$ and $P(o_1, o_2)$ *that are true in* $\mathcal{I}$, i.e., that are such that $o \in A^{\mathcal{I}}$ and $(o_1, o_2) \in P^{\mathcal{I}}$.

We are particularly interested in DL ontologies expressed in *DL-Lite*$_{\mathcal{R}}$, the member of the *DL-Lite* family [15] that underpins `OWL 2 QL`, i.e., the `OWL 2` profile especially designed for efficient query answering [53]. A *DL-Lite*$_{\mathcal{R}}$ ontology $\mathcal{O}$ is a finite set of *assertions* of the form:

$$B_1 \sqsubseteq B_2 \quad R_1 \sqsubseteq R_2 \quad \text{(concept/role inclusion)}$$

$$B_1 \sqsubseteq \neg B_2 \quad R_1 \sqsubseteq \neg R_2 \quad \text{(concept/role disjointness)}$$

where B_1, B_2 are *basic concepts*, i.e., expressions of the form A, $\exists P$, or $\exists P^-$, with $A \in \Gamma_A$ and $P \in \Gamma_P$, and R_1 and R_2 *basic roles*, i.e., expressions of the form P, or P^- with $P \in \Gamma_P$. We assume that the special bottom concept $\bot$ never occurs in the right-hand side of inclusion assertions. This is without loss of generality, since each inclusion assertion of the form $B \sqsubseteq \bot$ is logically equivalent to the disjointness assertion $B \sqsubseteq \neg B$.

For the constructs of *DL-Lite*$_{\mathcal{R}}$, the interpretation function $\cdot^{\mathcal{I}}$ of an interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ extends to basic concepts and basic roles as follows: $(\exists P)^{\mathcal{I}} = \{o \mid \exists o'.(o, o') \in P^{\mathcal{I}}\}$ and $(P^-)^{\mathcal{I}} = \{(o, o') \mid (o', o) \in P^{\mathcal{I}}\}$. Finally, 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, $R_1^{\mathcal{I}} \cap R_2^{\mathcal{I}} = \emptyset$).

Syntax and semantics of ontology-based data management According to [47,58], 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 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. For readability purposes, with a slight abuse of notation, we sometimes denote an OBDM system as a quadruple $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$, where $\langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ is an OBDM specification and D is an $\mathcal{S}$-database.

In this paper, each assertion in the mapping component of an OBDM system is a *Global-And-Local-As-View (GLAV)* assertion [27], that is, an assertion 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 *Global-As-View (GAV)* and *Local-As-View (LAV)* assertions [45]: in a GAV (resp., LAV) assertion, $q_{\mathcal{O}}$ (resp., $q_{\mathcal{S}}$) is simply an atom without existential variables. Finally, a GLAV (resp., GAV, LAV, GAV∩LAV) mapping $\mathcal{M}$ consists in a finite set of GLAV (resp., GAV, LAV, both GAV and LAV) assertions.

Given a GLAV mapping $\mathcal{M}$ relating a source schema $\mathcal{S}$ to an ontology $\mathcal{O}$, an interpretation $\mathcal{I}$, and an $\mathcal{S}$-database D, we denote by $\langle \mathcal{I}, D \rangle \models \mathcal{M}$ the fact that $(c_1, \ldots, c_n) \in q_{\mathcal{S}}^D$ implies $(c_1^{\mathcal{I}}, \ldots, c_n^{\mathcal{I}}) \in q_{\mathcal{O}}^{\mathcal{I}}$ for each possible mapping assertion $q_{\mathcal{S}} \rightarrow q_{\mathcal{O}}$ in $\mathcal{M}$ and for each possible tuple $(c_1, \ldots, c_n)$ of constants occurring in D. Here, $q_{\mathcal{O}}^{\mathcal{I}}$ denotes the evaluation of $q_{\mathcal{O}}$ over the interpretation $\mathcal{I}$ seen as a (potentially infinite) set of facts.

As customary in OBDM, we define the semantics of OBDM systems $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$ by specifying which are the first-order models of the OBDM specification $\langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ relative to the $\mathcal{S}$-database D. Specifically, we say that an interpretation $\mathcal{I}$ is a *model* of an OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$ if (i) $\mathcal{I} \models \mathcal{O}$, and (ii) $\langle \mathcal{I}, D \rangle \models \mathcal{M}$. Finally, we say that an OBDM system Σ is *consistent* if it has at least one model, *inconsistent* otherwise.

For an OBDM system $\Sigma = \langle J, D \rangle$, with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$, and a query $q_{\mathcal{O}}$ over $\mathcal{O}$, we denote by $cert^D_{q_{\mathcal{O}}, J}$ the set of *certain answers* of $q_{\mathcal{O}}$ w.r.t. Σ, i.e., 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. 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. Finally, we say that two queries q_1 and q_2 are equivalent w.r.t. an OBDM system $\Sigma = \langle J, D \rangle$ if $cert^D_{q_1, J} = cert^D_{q_2, J}$.

Query rewriting Given 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 [15] on $\mathcal{O}$ and $q_{\mathcal{O}}$ (slightly extended to deal also with the bottom concept $\bot$). In a nutshell, the PerfectRef algorithm uses the concept/role inclusions of the input *DL-Lite*$_{\mathcal{R}}$ ontology $\mathcal{O}$ as rewriting rules applied to the input UCQ $q_{\mathcal{O}}$. In this way, it compiles into the query $q_{\mathcal{O}}$ all the knowledge provided by $\mathcal{O}$ that is relevant to answering the query. PerfectRef applies repeatedly the following two steps until a fixpoint is reached: step (i) uses concept/role inclusions as rewriting rules to rewrite query atoms one by one,

each time producing a new CQ to be added to the final output; step (ii) unifies the query atoms to enable further executions of step (i). For additional details, we refer the reader to [15].

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 [17,29], 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.

Now, let $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ be an OBDM specification where $\mathcal{O}$ is a *DL-Lite*$_\mathcal{R}$ ontology and $\mathcal{M}$ is a GLAV mapping. Given a UCQ $q_\mathcal{O}$ over $\mathcal{O}$, in what follows, we denote by $\mathsf{rew}_{q_\mathcal{O}, J}$ the following UCQ over $\mathcal{S}$: $\mathsf{rew}_{q_\mathcal{O}, J} := \mathsf{MapRef}(\mathcal{M}, \mathsf{PerfectRef}(\mathcal{O}, q_\mathcal{O}))$. By combining the above observation regarding MapRef with the correctness of the PerfectRef algorithm [15] and the fact that *DL-Lite*$_\mathcal{R}$ is insensitive to the adoption of the UNA for UCQ answering [5], we derive that $cert^D_{q_\mathcal{O}, J} = \mathsf{rew}^D_{q_\mathcal{O}, J}$ holds for each UCQ $q_\mathcal{O}$ over $\mathcal{O}$ and for each $\mathcal{S}$-database D such that $\langle J, D \rangle$ is consistent. Furthermore, by combining the above observation regarding MapRef with results of [58], we derive that, given an $\mathcal{S}$-database D, the OBDM system $\langle J, D \rangle$ is inconsistent if and only if $\mathsf{rew}^D_{V_\mathcal{O}, J} = \{\langle\rangle\}$. Here, $V_\mathcal{O}$ is the *$\mathcal{O}$-violation query*, i.e., the boolean UCQ over $\mathcal{O}$ constituted by the disjunct $\{() \mid \exists y.\bot(y)\}$ and 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., $R = P^-$.

Canonical structure Given an $\mathcal{S}$-database D and a GLAV mapping $\mathcal{M}$ relating a source schema $\mathcal{S}$ to an ontology $\mathcal{O}$, the *chase* [13] of D with respect to $\mathcal{M}$, denoted by $\mathcal{M}(D)$, is the set of atoms computed as follows: (i) $\mathcal{M}(D)$ is initially set to the empty set of atoms; 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 $\vec{c}$ of constants occurring in D 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, we set $\mathcal{M}(D) := \mathcal{M}(D) \cup h'(\varphi_\mathcal{O}(\vec{x}, \vec{z}))$, 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 for the technical development of the next sections. Let $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$ be an OBDM system where $\mathcal{O}$ is a *DL-Lite*$_\mathcal{R}$ ontology and $\mathcal{M}$ is a GLAV mapping. The *canonical structure* of $\mathcal{O}$ with respect to $\mathcal{M}$ and D, denoted by $\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}$, is the (potentially 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 [15, 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 [28, Proposition 4.2] with [15, Theorem 29], it is well-known that, for a UCQ $q_\mathcal{O} = \{\vec{x_1} \mid \exists \vec{y_1}.\phi^1_\mathcal{O}(\vec{x_1}, \vec{y_1})\} \cup \cdots \cup \{\vec{x_p} \mid \exists \vec{y_p}.\phi^p_\mathcal{O}(\vec{x_p}, \vec{y_p})\}$ over $\mathcal{O}$ and a tuple $\vec{c}$ of constants occurring in D, if $\Sigma = \langle J, D \rangle$ is consistent, then the following holds: $\vec{c} \in cert^D_{q_\mathcal{O}, J}$ if and only if $(set(\phi^i_\mathcal{O}), \vec{x_i}) \rightarrow (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{c})$ for some $i \in [1, p]$.

4. Formal framework

In what follows, we implicitly use $\Sigma = \langle J, D \rangle$ to denote 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 two sets λ^+ and λ^- of tuples of constants occurring in D (i.e., λ^+ and λ^- are D-datasets) of positive and negative examples, respectively, we aim at finding a query $q_\mathcal{O}$ over the ontology $\mathcal{O}$ in a certain target query language $\mathcal{Q}$ that logically separates λ^+ and λ^- w.r.t. Σ. Since the evaluation of queries in OBDM systems is based on certain answers, we are naturally led to the following definition.

Definition 1. $q_\mathcal{O} \in \mathcal{Q}$ is a *proper* Σ*-separation* of λ^+ and λ^- in the query language $\mathcal{Q}$, if both conditions (i) $\lambda^+ \subseteq cert^D_{q_\mathcal{O}, J}$ and (ii) $cert^D_{q_\mathcal{O}, J} \cap \lambda^- = \emptyset$ hold.

The next example illustrates the above definition.

Example 2. Recall the OBDM system $\Sigma = \langle J, D\rangle$, the D-datasets $\lambda_1^+ = \{(c_1), (c_2), (c_3)\}$ and $\lambda_1^- = \{(c_5)\}$, and the CQ $q_{\mathcal{O}}^1$ over the ontology $\mathcal{O}$ illustrated in Example 1. As already observed, we have that $cert^D_{q_{\mathcal{O}}^1, J} = \{(c_1), (c_2), (c_3), (c_4)\}$, and therefore $q_{\mathcal{O}}^1$ is a proper Σ-separation of λ_1^+ and λ_1^- in CQ.

Consider now a slight variation of the negative examples, i.e., consider $\lambda^+ = \lambda_1^+$ and $\lambda^- = \{(c_4), (c_5)\}$. Since $q_{\mathcal{O}}^1$ and $q_{\mathcal{O}}^2 = \{(x) \mid \exists y.\mathsf{EnrolledIn}(x, y)\}$ are such that $cert^D_{q_{\mathcal{O}}^1, J} = \{(c_1), (c_2), (c_3), (c_4)\}$ and $cert^D_{q_{\mathcal{O}}^2, J} = \{(c_1), (c_2), (c_3), (c_4), (c_5)\}$, and since $q_{\mathcal{O}}^3 = \{(x) \mid \mathsf{MathStudent}(x)\}$ and $q_{\mathcal{O}}^4 = \{(x) \mid \mathsf{ForeignStudent}(x)\}$ are such that $cert^D_{q_{\mathcal{O}}^3, J} = \{(c_1)\}$ and $cert^D_{q_{\mathcal{O}}^4, J} = \{(c_2)\}$, one can verify that no proper Σ-separation of λ^+ and λ^- in UCQ exists.

Clearly, the more expressive the target query language $\mathcal{Q}$, the more likely it is possible to distinguish (w.r.t. the OBDM system) the properties between the tuples in λ^+ and λ^- by means of the operators in $\mathcal{Q}$, and therefore the more likely the proper separation in $\mathcal{Q}$ exists. Unfortunately, the next example shows that, even in trivial cases and without any restriction on the target query language $\mathcal{Q}$, proper separations are not guaranteed to exist.

Example 3. Let $\Sigma = \langle J, D\rangle$ be the following OBDM system: (i) $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ is the OBDM specification in which $\mathcal{O} = \emptyset$, i.e., $\mathcal{O}$ contains no assertions, $\mathcal{S} = \{s_1, s_2\}$, 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 is the $\mathcal{S}$-database $D = \{s_1(c_1), s_2(c_2)\}$.

For the D-datasets $\lambda^+ = \{(c_1)\}$ and $\lambda^- = \{(c_2)\}$, one can trivially verify that, whatever is the query language $\mathcal{Q}$, there can be no query $q_{\mathcal{O}} \in \mathcal{Q}$ over $\mathcal{O}$ for which $cert^D_{q_{\mathcal{O}}, J}$ include the tuple (c_1) but does not include the tuple (c_2). To see this, note that $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)} = \{A(c_1), A(c_2)\}$. It follows that, whatever is the query language $\mathcal{Q}$, there can be no query $q_{\mathcal{O}} \in \mathcal{Q}$ over $\mathcal{O}$ which is a proper Σ-separation of λ^+ and λ^- in $\mathcal{Q}$.

Notice 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 a proper Σ-separation of λ^+ and λ^- in UCQ would simply be the CQ $\{(x) \mid A(x)\}$ over the ontology $\mathcal{O}$.

Borrowing similar ideas from [23], to remedy situations where proper separations do not exist, we now introduce approximations of proper separations in terms of completeness and soundness. More specifically, a complete separation is a query that captures all the positive examples in its set of certain answers w.r.t. the OBDM system (i.e., satisfies condition (i) of Definition 1), whereas a sound separation is a query that contains none of the negative examples in its set certain answers w.r.t. the OBDM system (i.e., satisfies condition (ii) of Definition 1).

Definition 2. $q_{\mathcal{O}} \in \mathcal{Q}$ is a *complete* (resp., *sound*) Σ-*separation* of λ^+ and λ^- in the query language $\mathcal{Q}$, if $\lambda^+ \subseteq cert^D_{q_{\mathcal{O}}, J}$ (resp., $cert^D_{q_{\mathcal{O}}, J} \cap \lambda^- = \emptyset$).

We observe that the condition for being a complete (resp., sound) separation does not involve λ^- (resp., λ^+).

Example 4. Refer to Example 2. We have that $q_{\mathcal{O}}^1$ and $q_{\mathcal{O}}^2$ are complete Σ-separations of λ^+ and λ^- in UCQ, whereas $q_{\mathcal{O}}^3$ and $q_{\mathcal{O}}^4$ are sound Σ-separations of λ^+ and λ^- in UCQ.

As the above example manifests, there may be several complete and sound separations. In those cases, the interest is unquestionably in those queries that approximate at *best* the proper one, at least relative to a target query language $\mathcal{Q}$. Informally, a $\mathcal{Q}$-minimally complete separation is a complete separation in $\mathcal{Q}$ containing a minimal (w.r.t. set containment) possible set of negative examples in its set of certain answers, whereas a $\mathcal{Q}$-maximally sound separation is a sound separation in $\mathcal{Q}$ capturing a maximal (w.r.t. set containment) possible set of positive examples in its set of certain answers.

Definition 3. $q_{\mathcal{O}}$ is a $\mathcal{Q}$-*minimally complete* (resp., $\mathcal{Q}$-*maximally sound*) Σ-*separation* of λ^+ and λ^-, if $q_{\mathcal{O}}$ is a complete (resp., sound) Σ-separation of λ^+ and λ^- in $\mathcal{Q}$ and there is no $q'_{\mathcal{O}} \in \mathcal{Q}$ satisfying both the following two conditions:

(i) $q'_{\mathcal{O}}$ is a complete (resp., sound) Σ-separation of λ^+ and λ^- in $\mathcal{Q}$;
(ii) $cert^D_{q'_{\mathcal{O}}, J} \cap \lambda^- \subset cert^D_{q_{\mathcal{O}}, J} \cap \lambda^-$ (resp., $cert^D_{q_{\mathcal{O}}, J} \cap \lambda^+ \subset cert^D_{q'_{\mathcal{O}}, J} \cap \lambda^+$).

By definition, it is immediate to verify that a Σ-proper separation of λ^+ and λ^- in a query language $\mathcal{Q}$ is both a $\mathcal{Q}$-minimally complete Σ-separation of λ^+ and λ^- and a $\mathcal{Q}$-maximally sound Σ-separation of λ^+ and λ^-.

Example 5. Refer again to Example 2. One can verify that the CQ $q^1_{\mathcal{O}}$ is a UCQ-minimally complete Σ-separation of λ^+ and λ^-, whereas $q^2_{\mathcal{O}}$ is not. Moreover, both $q^3_{\mathcal{O}}$ and $q^4_{\mathcal{O}}$ are CQ-maximally sound Σ-separations of λ^+ and λ^-, but neither of them is a UCQ-maximally sound Σ-separation of λ^+ and λ^-. Indeed, a UCQ-maximally sound Σ-separation of λ^+ and λ^- is $q^5_{\mathcal{O}} = q^3_{\mathcal{O}} \cup q^4_{\mathcal{O}}$.

We point out that all the above definitions are a generalization of the definitions illustrated in [22] which deal only with datasets of positive examples, rather than datasets of both positive and negative examples as done here. More specifically, in [22], as well as in all the works addressing the separability task, when only a D-dataset λ^+ of positive examples is provided, to λ^- it is implicitly associated the D-dataset $\lambda^- = dom(D)^n \setminus \lambda^+$, where n is the arity of the tuples in λ^+. With this remark in mind, we can now specialize the above definitions for the only dataset of positive examples case and report the definitions given in [22].

Definition 4. $q_{\mathcal{O}} \in \mathcal{Q}$ is a *proper* (resp., *complete*, *sound*) Σ-*characterization* of λ^+ in the query language $\mathcal{Q}$, if $cert^D_{q_{\mathcal{O}},J} = \lambda^+$ (resp., $\lambda^+ \subseteq cert^D_{q_{\mathcal{O}},J}$, $cert^D_{q_{\mathcal{O}},J} \subseteq \lambda^+$).

$q_{\mathcal{O}}$ is a $\mathcal{Q}$-*minimally complete* (resp., $\mathcal{Q}$-*maximally sound*) Σ-*characterization* of λ^+, if $q_{\mathcal{O}}$ is a complete (resp., sound) Σ-characterization of λ^+ in $\mathcal{Q}$ and there is no $q'_{\mathcal{O}} \in \mathcal{Q}$ satisfying both the following two conditions:

(i) $q'_{\mathcal{O}}$ is a complete (resp., sound) Σ-characterization of λ^+ in $\mathcal{Q}$;
(ii) $cert^D_{q'_{\mathcal{O}},J} \subset cert^D_{q_{\mathcal{O}},J}$ (resp., $cert^D_{q_{\mathcal{O}},J} \subset cert^D_{q'_{\mathcal{O}},J}$).

In other words, given an OBDM system $\Sigma = \langle J, D\rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$, a D-dataset λ^+ of arity n, and a query $q_{\mathcal{O}} \in \mathcal{Q}$, we have that $q_{\mathcal{O}}$ is a proper in $\mathcal{Q}$ (resp., complete in $\mathcal{Q}$, sound in $\mathcal{Q}$, $\mathcal{Q}$-minimally complete, $\mathcal{Q}$-maximally sound) Σ-characterization of λ^+ if and only if $q_{\mathcal{O}}$ is a proper in $\mathcal{Q}$ (resp., complete in $\mathcal{Q}$, sound in $\mathcal{Q}$, $\mathcal{Q}$-minimally complete, $\mathcal{Q}$-maximally sound) Σ-separation of λ^+ and λ^-, where $\lambda^- = dom(D)^n \setminus \lambda^+$.

4.1. *Relation with the* abstraction *reasoning task*

We now discuss the relation between the notion of separation in OBDM introduced here with the notion of *Abstraction* [19,20], recently introduced in [18] and studied under various scenarios [21,23,24,50] for addressing several reverse engineering tasks in OBDM. 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 *a 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, in the separation task also the $\mathcal{S}$-database D is given, and instead of a query $q_{\mathcal{S}}$ we have two set of tuples λ^+ and λ^- of constants taken from D, and the aim is to find a query $q_{\mathcal{O}}$ over $\mathcal{O}$ such that both $\lambda^+ \subseteq cert^D_{q_{\mathcal{O}},J}$ and $cert^D_{q_{\mathcal{O}},J} \cap \lambda^- = \emptyset$ hold.

Following [23], we also say that a query $q_{\mathcal{O}}$ over $\mathcal{O}$ is *a complete (resp., sound) J-abstraction of $q_{\mathcal{S}}$* if $q^D_{\mathcal{S}} \subseteq cert^D_{q_{\mathcal{O}},J}$ (resp., $cert^D_{q_{\mathcal{O}},J} \subseteq q^D_{\mathcal{S}}$) for each $\mathcal{S}$-database D for which $\langle J, D\rangle$ is consistent. The next proposition establishes a precise relationship between the notion of separation in OBDM introduced here and the notion of abstraction.

Proposition 1. *Let $\Sigma = \langle J, D\rangle$ be a consistent OBDM system, λ^+ and λ^- be two D-datasets, and $q_{\mathcal{S}}$ be a query that explains λ^+ and λ^- inside D. If a query $q_{\mathcal{O}} \in \mathcal{Q}$ is a perfect (resp., complete, sound) J-abstraction of $q_{\mathcal{S}}$, then $q_{\mathcal{O}}$ is a proper (resp., complete, sound) Σ-separation of λ^+ and λ^- in $\mathcal{Q}$.*

Proof. Suppose $q_{\mathcal{O}} \in \mathcal{Q}$ is a perfect (resp., complete, sound) J-abstraction of $q_{\mathcal{S}}$, i.e., $cert^{D'}_{q_{\mathcal{O}},J} = q^{D'}_{\mathcal{S}}$ (resp., $q^{D'}_{\mathcal{S}} \subseteq cert^{D'}_{q_{\mathcal{O}},J}$, $cert^{D'}_{q_{\mathcal{O}},J} \subseteq q^{D'}_{\mathcal{S}}$) for each $\mathcal{S}$-database D' for which $\langle J, D'\rangle$ is consistent. Since by assumption $\Sigma = \langle J, D\rangle$ is consistent, we have that $cert^D_{q_{\mathcal{O}},J} = q^D_{\mathcal{S}}$ (resp., $q^D_{\mathcal{S}} \subseteq cert^D_{q_{\mathcal{O}},J}$, $cert^D_{q_{\mathcal{O}},J} \subseteq q^D_{\mathcal{S}}$). Now, since both $\lambda^+ \subseteq q^D_{\mathcal{S}}$ and $q^D_{\mathcal{S}} \cap \lambda^- = \emptyset$ hold by the assumption that $q_{\mathcal{S}}$ explains λ^+ and λ^- inside D, we derive that both $\lambda^+ \subseteq cert^D_{q_{\mathcal{O}},J}$ and $cert^D_{q_{\mathcal{O}},J} \cap \lambda^- = \emptyset$ (resp., only $\lambda^+ \subseteq cert^D_{q_{\mathcal{O}},J}$, only $cert^D_{q_{\mathcal{O}},J} \cap \lambda^- = \emptyset$) hold, which means that $q_{\mathcal{O}}$ is a proper (resp., complete, sound) Σ-separation of λ^+ and 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 6. Let $\Sigma = \langle J, D\rangle$ be as follows. $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ is the OBDM specification in which $\mathcal{O} = \emptyset$, i.e., $\mathcal{O}$ contains no assertions, $\mathcal{S} = \{s_1, s_2, s_3\}$, and $\mathcal{M}$ consists of the following two GAV assertions:

$$\{(x) \mid s_1(x) \wedge s_2(x)\} \rightarrow \{(x) \mid \mathsf{Student}(x)\}$$
$$\{(x) \mid s_3(x)\} \rightarrow \{(x) \mid \mathsf{Student}(x)\}$$

Let also the $\mathcal{S}$-database be $D = \{s_1(a), s_2(a), s_2(b)\}$ and the D-datasets λ^+ and λ^- be $\lambda^+ = \{(a)\}$ and $\lambda^- = \{(b)\}$. Consider, moreover, the CQ $q_{\mathcal{S}} = \{(x) \mid s_1(x)\}$. One can see that the query $q_{\mathcal{S}}$ explains λ^+ and λ^- inside D because $q_{\mathcal{S}}^D = \{(a)\}$. Consider now the CQ $q_{\mathcal{O}} = \{(x) \mid \mathsf{Student}(x)\}$ over $\mathcal{O}$. One can see that $q_{\mathcal{O}}$ is a proper (and therefore, also a complete and a sound) Σ-separation of λ^+ and λ^- in CQ because $cert^D_{q_{\mathcal{O}},J} = \{(a)\}$.

Notice, however, that for the $\mathcal{S}$-database $D' = \{s_1(a), s_3(b)\}$ we have that (i) $\langle J, D'\rangle$ is a consistent OBDM system, (ii) $q_{\mathcal{S}}^{D'} = \{(a)\}$, and (iii) $cert^{D'}_{q_{\mathcal{O}},J} = \{(b)\}$. Thus, $q_{\mathcal{O}}$ is neither a complete nor a sound (and therefore, nor a perfect) J-abstraction of $q_{\mathcal{S}}$. Indeed both $q_{\mathcal{S}}^{D'} \not\subseteq cert^{D'}_{q_{\mathcal{O}},J}$ (witnessing that $q_{\mathcal{O}}$ is not a complete J-abstraction of $q_{\mathcal{S}}$) and $cert^{D'}_{q_{\mathcal{O}},J} \not\subseteq q_{\mathcal{S}}^{D'}$ (witnessing that $q_{\mathcal{O}}$ is not a sound J-abstraction of $q_{\mathcal{S}}$) hold.

4.2. Computational problems associated with the framework

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

- *Verification:* given also a query $q_{\mathcal{O}} \in \mathcal{Q}$, check whether $q_{\mathcal{O}}$ is a proper (resp., complete, sound) Σ-separation of λ^+ and λ^- in $\mathcal{Q}$.
- *Computation:* compute any proper in $\mathcal{Q}$ (resp., any $\mathcal{Q}$-minimally complete, any $\mathcal{Q}$-maximally sound) Σ-separation of λ^+ and λ^-, provided it exists.
- *Existence:* check whether there exists a query $q_{\mathcal{O}} \in \mathcal{Q}$ that is a proper in $\mathcal{Q}$ (resp., a $\mathcal{Q}$-minimally complete, a $\mathcal{Q}$-maximally sound) Σ-separation of λ^+ and λ^-.

Analogous computational problems can be defined when in the input we have only one D-dataset λ^+ of arity n, rather than two D-datasets λ^+ and λ^-, and we implicitly think of λ^- as $dom(D)^n \setminus \lambda^+$.

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) $\mathcal{L}_{\mathcal{O}}$ 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.

Proposition 2. *Let $\Sigma = \langle J, D\rangle$ be an OBDM system, and λ^+ and λ^- be two D-datasets of arity n such that $\lambda^+ \cup \lambda^- = dom(D)^n$. If q_1 and q_2 are UCQ-minimally complete (resp., UCQ-maximally sound) Σ-separations of λ^+ and λ^-, then they are equivalent w.r.t. Σ.*

Proof. We first address the case of UCQ-maximally sound, and then the case of UCQ-minimally complete.

Assume that q_1 and q_2 are UCQ-maximally sound Σ-separations of λ^+ and λ^- and suppose, for the sake of contradiction, that they are not equivalent w.r.t. Σ. This implies the existence of a tuple $\vec{c}$ such that $\vec{c} \notin cert^D_{q_1,J}$ and $\vec{c} \in cert^D_{q_2,J}$. Observe that, since $\lambda^+ \cup \lambda^- = dom(D)^n$, $\vec{c}$ must be such that $\vec{c} \in \lambda^+$, otherwise q_2 would not be a sound Σ-separation of λ^+ and λ^- in UCQ, thus reaching a contradiction. But then, the UCQ $Q = q_1 \cup q_2$ is such that (i) since both q_1 and q_2 are sound Σ-separations of λ^+ and λ^- in UCQ, Q is a sound Σ-separation of λ^+ and λ^- in UCQ as well, and (ii) $cert^D_{q_1,J} \cap \lambda^+ \subset cert^D_{Q,J} \cap \lambda^+$ holds because of tuple $\vec{c}$. Obviously, this contradicts the fact that q_1 is a UCQ-maximally sound Σ-separation of λ^+ and λ^-.

Assume now that q_1 and q_2 are UCQ-minimally complete Σ-separations of λ^+ and λ^- and suppose, for the sake of contradiction, that they are not equivalent w.r.t. Σ. This implies the existence of a tuple $\vec{c}$ such that $\vec{c} \in cert^D_{q_1,J}$

and $\vec{c} \notin cert^D_{q_2,J}$. Observe that, since $\lambda^+ \cup \lambda^- = dom(D)^n$, $\vec{c}$ must be such that $\vec{c} \in \lambda^-$, otherwise q_2 would not be a complete Σ-separation of λ^+ and λ^- in UCQ, thus reaching a contradiction. But then, consider the query Q such that $cert^D_{Q,J} = cert^D_{q_1,J} \cap cert^D_{q_2,J}$. Obviously, since q_1 and q_2 are UCQs, Q exists and can be expressed as a UCQ. Moreover, (i) since q_1 and q_2 are complete Σ-separations of λ^+ and λ^- in UCQ, Q is a complete Σ-separation of λ^+ and λ^- in UCQ as well, and (ii) $cert^D_{Q,J} \cap \lambda^- \subset cert^D_{q_1,J} \cap \lambda^-$ holds because of tuple $\vec{c}$. Obviously, this contradicts the fact that q_1 is a UCQ-minimally complete Σ-separation of λ^+ and λ^-. □

This also means that, in our scenario, for an OBDM system $\Sigma = \langle J, D\rangle$ and two D-datasets λ^+ and λ^- of arity n such that $\lambda^+ \cup \lambda^- = dom(D)^n$, if a proper Σ-separation of λ^+ and λ^- in UCQ exists, then it is unique up to equivalence w.r.t. Σ. Furthermore, for the characterizability case where λ^- is implicitly set to be $dom(D)^n \setminus \lambda^+$, proper in UCQ, UCQ-minimally complete, and UCQ-maximally sound Σ-characterizations of λ^+ are always unique up to equivalence w.r.t. Σ, provided they exist.

Secondly, in this scenario, as one may expect, proper separations are less likely to exist than explanations in the plain relational database case.

Proposition 3. *Let $\Sigma = \langle J, D\rangle$ be a consistent OBDM system, and λ^+ and λ^- be two D-datasets. If there exists a proper Σ-separation of λ^+ and λ^- in UCQ, then λ^+ and λ^- are UCQ-explainable inside D.*

Proof. Suppose there exists a proper Σ-separation of λ^+ and λ^- in UCQ, i.e., there is a UCQ $q_{\mathcal{O}}$ over $\mathcal{O}$ for which $\lambda^+ \subseteq cert^D_{q_{\mathcal{O}},J}$ and $cert^D_{q_{\mathcal{O}},J} \cap \lambda^- = \emptyset$. Recall from Section 3 that the UCQ $\mathsf{rew}_{q_{\mathcal{O}},J}$ over $\mathcal{S}$ is such that $cert^{D'}_{q_{\mathcal{O}},J} = \mathsf{rew}^{D'}_{q_{\mathcal{O}},J}$ for each $\mathcal{S}$-database D' for which $\langle J, D'\rangle$ is consistent. Since $\Sigma = \langle J, D\rangle$ is consistent by assumption, we have that $cert^D_{q_{\mathcal{O}},J} = \mathsf{rew}^D_{q_{\mathcal{O}},J}$. Thus, $\mathsf{rew}_{q_{\mathcal{O}},J}$ is such that both $\lambda^+ \subseteq \mathsf{rew}^D_{q_{\mathcal{O}},J}$ and $\mathsf{rew}^D_{q_{\mathcal{O}},J} \cap \lambda^- = \emptyset$ hold, from which immediately follows that $\mathsf{rew}_{q_{\mathcal{O}},J}$ explains λ^+ and λ^- inside D by definition, and thus λ^+ and λ^- are UCQ-explainable inside D. □

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

Note that Definition 1 can be seen as a generalization of the classical notion of explanation in the plain relational database case [7], when one also has to deal with mapping assertions and ontology assertions. In the specific case of OBDM systems $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D\rangle$ such that $\mathcal{O} = \emptyset$ and $\mathcal{M}$ contains assertions of the form $\{(x) \mid s(x)\} \rightarrow \{(x) \mid A(x)\}$ and $\{(x) \mid s'(x_1, x_2)\} \rightarrow \{(x) \mid P(x_1, x_2)\}$ providing a one-to-one correspondence between the alphabet of the ontology and the predicate symbols of the source schema, we observe that, given two D-datasets λ^+ and λ^- (resp., a D-dataset λ^+), there exists a Σ-proper separation of λ^+ and λ^- (resp., a Σ-proper characterization of λ^+) in UCQ if and only if λ^+ and λ^- are UCQ-explainable (resp., λ^+ is UCQ-definable) inside D.

We conclude this section by observing that, in the case of plain relational databases, two D-datasets λ^+ and λ^- (resp., a D-dataset λ^+) are UCQ-explainable (resp., is UCQ-definable) inside D if and only if the union of the CQs describing the tuples in λ^+ achieves the separation (resp., the definition). More formally, λ^+ and λ^- (resp., λ^+) are UCQ-explainable (resp., is UCQ-definable) inside D if and only if $query(D, \vec{c_1}) \cup \cdots \cup query(D, \vec{c_n})$ explains λ^+ and λ^- inside D (resp., defines λ^+ inside D), where $\lambda^+ = \{\vec{c_1}, \ldots, \vec{c_n}\}$. When casting this problem to the OBDM setting, one of the key challenges is that the separating query needs to be formulated over the ontology level, while data reside at the source level, with a mapping layer mediating the two. One of the contributions of this paper is to provide a similar characterization as in the relational database case for the OBDM setting (cf. Corollary 3).

5. The verification problem

We now define the verification problems for X-separability (X-VSEP) and X-characterization (X-VCHAR), where $X = \{\text{Proper, Complete, Sound}\}$. These decision problems are parametric with respect to the ontology language $\mathcal{L}_{\mathcal{O}}$

to express the ontology $\mathcal{O}$, the mapping language $\mathcal{L}_\mathcal{M}$ to express the mapping $\mathcal{M}$, and the target query language $\mathcal{Q}$ to express the query $q_\mathcal{O}$ over $\mathcal{O}$.

X-VSEP($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$)

Input: An OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$, two D-datasets λ^+ and λ^-, and a query $q_\mathcal{O} \in \mathcal{Q}$ over $\mathcal{O}$, where $\mathcal{O} \in \mathcal{L}_\mathcal{O}$ and $\mathcal{M} \in \mathcal{L}_\mathcal{M}$.
Question: Is $q_\mathcal{O}$ a **X** Σ-separation of λ^+ and λ^- in $\mathcal{Q}$?

X-VCHAR($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$)

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

We also introduce two important special cases of the above decision problems, namely: X-VSTSEP($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$) and X-VSTCHAR($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$), which are special cases of X-VSEP($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$) and X-VCHAR($\mathcal{L}_\mathcal{O}$, $\mathcal{L}_\mathcal{M}$, $\mathcal{Q}$), respectively, in which the all the input D-datasets are singleton sets (i.e., they consist of just a single tuple).

In what follows, given a syntactic object x such as a query, an ontology, or a mapping, we denote by $|x|$ its size, which is the number of symbols needed to write it, with names of predicates, variables, etc. counting as one.

We start by analyzing the upper bounds for the case $X =$ Complete. The proof of the next theorems rely on the fact that, given an OBDM specification $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ of our considered scenario and a UCQ $q_\mathcal{O}$ over $\mathcal{O}$, each disjunct in $\mathsf{PerfectRef}(\mathcal{O}, q_\mathcal{O})$ can be obtained after a polynomial number of transformations of an initial disjunct in $q_\mathcal{O}$ [15], and, analogously, each disjunct in $\mathsf{MapRef}(\mathcal{M}, q_\mathcal{O})$ can be obtained after a polynomial number of transformations of an initial disjunct in $q_\mathcal{O}$ [17]. This clearly implies that, although there may be an exponential number of disjuncts in the UCQ $\mathsf{rew}_{q_\mathcal{O},J}$, the size of each such disjunct is always bounded by a polynomial in the size of $q_\mathcal{O}$, $\mathcal{O}$, and $\mathcal{M}$. More precisely, the size of each disjunct in $\mathsf{PerfectRef}(\mathcal{O}, q_\mathcal{O})$ is at most $|q_\mathcal{O}|$ and the size of each disjunct in $\mathsf{rew}_{q_\mathcal{O},J}$ is at most $|\mathcal{M}| \cdot |q_\mathcal{O}|$.

Theorem 1. *Complete-V*SEP*(DL-Lite$_\mathcal{R}$, GLAV, UCQ) and Complete-V*CHAR*(DL-Lite$_\mathcal{R}$, GLAV, UCQ) are in NP.*

Proof. We address Complete-VSEP(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ). The case of Complete-VCHAR(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ) can be addressed in exactly the same way (recall that λ^- is immaterial for the complete case). In particular, we now show how to check in NP whether $q_\mathcal{O}$ is a complete Σ-separation of λ^+ and λ^- in UCQ (i.e., $\lambda^+ \subseteq cert^D_{q_\mathcal{O},J}$), where $\Sigma = \langle J, D \rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ in which $\mathcal{O}$ is a *DL-Lite*$_\mathcal{R}$ ontology and $\mathcal{M}$ is a GLAV mapping.

Let n be the arity of the tuple(s) in the D-datasets λ^+ and λ^-. For each n-tuple of constants $\vec{c} \in \lambda^+$, we first guess (i) a CQ $q'_\mathcal{O}$ over $\mathcal{O}$ which is either of arity n and size at most $|q_\mathcal{O}|$, 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 $\rho_\mathcal{O}$ of ontology assertions; (iii) a CQ $q_\mathcal{S}$ over $\mathcal{S}$ of size at most $|\mathcal{M}| \cdot |q'_\mathcal{O}|$ 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 $\rho_\mathcal{O}$, either we can rewrite a disjunct of $q_\mathcal{O}$ into $q'_\mathcal{O}$ through $\mathcal{O}$ (i.e., $q'_\mathcal{O} \in \mathsf{PerfectRef}(\mathcal{O}, q_\mathcal{O})$), or we can rewrite a disjunct of $V_\mathcal{O}$ into $q'_\mathcal{O}$ through $\mathcal{O}$ (i.e., $q'_\mathcal{O} \in \mathsf{PerfectRef}(\mathcal{O}, V_\mathcal{O})$); (ii) by means of $\rho_\mathcal{M}$, we can rewrite $q'_\mathcal{O}$ into $q_\mathcal{S}$ through $\mathcal{M}$ (i.e., $q_\mathcal{S} \in \mathsf{MapRef}(\mathcal{M}, q'_\mathcal{O})$, and thus either $q_\mathcal{S} \in \mathsf{rew}_{q_\mathcal{O},J}$ or $q_\mathcal{S} \in \mathsf{rew}_{V_\mathcal{O},J}$); and finally (iii) f consists in a homomorphism witnessing either $(set(\phi_\mathcal{S}), \vec{x}) \rightarrow (D, \vec{c})$, i.e., $\vec{c} \in q^D_\mathcal{S}$ (and therefore $\vec{c} \in \mathsf{rew}^D_{q_\mathcal{O},J}$, which means $\vec{c} \in cert^D_{q_\mathcal{O},J}$), or $(set(\phi_\mathcal{S}), ()) \rightarrow (D, ())$, i.e., $q^D_\mathcal{S} = \{\langle\rangle\}$ (and therefore $\mathsf{rew}^D_{V_\mathcal{O},J} = \{\langle\rangle\}$, which means that Σ is inconsistent and thus $\vec{c} \in cert^D_{q_\mathcal{O},J}$ by definition of certain answers). □

By exploiting the above result, we now address the upper bounds for the case $X =$ Sound.

Theorem 2. *Sound-V*SEP*(DL-Lite$_\mathcal{R}$, GLAV, UCQ) and Sound-V*CHAR*(DL-Lite$_\mathcal{R}$, GLAV, UCQ) are in coNP.*

Proof. We start with Sound-VSEP(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ), and then we consider Sound-VCHAR(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ). In particular, we now show how to check in NP whether $q_\mathcal{O}$ is not a sound Σ-separation of λ^+ and λ^- in UCQ (i.e., $cert^D_{q_\mathcal{O},J} \cap \lambda^- \neq \emptyset$), where $\Sigma = \langle J, D\rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ in which $\mathcal{O}$ is a *DL-Lite*$_\mathcal{R}$ ontology and $\mathcal{M}$ is a GLAV mapping.

We first guess (i) a tuple of constants $\vec{c}$, and, exactly as in the proof of Theorem 1, we also guess (ii) $q'_\mathcal{O}$, $\rho_\mathcal{O}$, $q_\mathcal{S}$, $\rho_\mathcal{M}$, and f. Then, we check in polynomial time whether (i) $\vec{c} \in \lambda^-$, and (ii) using $q'_\mathcal{O}$, $\rho_\mathcal{O}$, $q_\mathcal{S}$, $\rho_\mathcal{M}$, and f, we follow exactly the same polynomial time procedure described in the proof of Theorem 1 to check whether $\vec{c} \in cert^D_{q_\mathcal{O},J}$.

As for the Sound-VCHAR(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ) case, it is sufficient to replace the check (i) $\vec{c} \in \lambda^-$ with the check $\vec{c} \in dom(D)^n \setminus \lambda^+$, where n is the arity of the tuple(s) in the D-dataset λ^+. Clearly this latter check can be done in polynomial time as well, by first checking that every constant of $\vec{c}$ effectively occurs in $dom(D)$ and then simply checking that $\vec{c} \notin \lambda^+$. □

We recall that the complexity class DP, introduced in [56], resides at the second level of the polynomial time hierarchy [61], and it is composed of all those decision problems that are the *intersection* of a decision problem in NP and a decision problem in coNP, that is, DP = NP ∧ coNP = $\{P_1 \cap P_2 \mid P_1 \in \text{NP} \wedge P_2 \in \text{coNP}\}$.

Since $q_\mathcal{O}$ is a proper Σ-separation of λ^+ and λ^- in $\mathcal{Q}$ if and only if it is both a sound, and a complete Σ-separation of λ^+ and λ^- in $\mathcal{Q}$, we immediately derive the following upper bounds for X = Proper.

Corollary 1. *Proper-V*SEP*(DL-Lite*$_\mathcal{R}$*, GLAV, UCQ) and Proper-V*CHAR*(DL-Lite*$_\mathcal{R}$*, GLAV, UCQ) are in DP.*

We now provide matching lower bounds, proving that all of them already hold for the singleton datasets special cases. More specifically, we show that they already hold for the same fixed OBDM system $\Sigma = \langle J, D\rangle$, same fixed D-datasets λ^+ and λ^- (resp., D-dataset λ^+) containing only a single unary tuple, and for CQs as queries. Furthermore, the fixed OBDM specification $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ is such that $\mathcal{O} = \emptyset$ (i.e., $\mathcal{O}$ contains no ontology assertions) and $\mathcal{M}$ is a GAV∩LAV mapping (i.e., $\mathcal{M}$ is both a GAV and a LAV mapping). To simplify the presentation, with a slight abuse of notation, from now on we denote by $\mathcal{L}_\mathcal{O} = \emptyset$ the ontology language allowing only for ontologies $\mathcal{O} = \emptyset$, i.e., ontologies $\mathcal{O}$ without assertions.

Theorem 3. *There is an OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D\rangle$ such that $\mathcal{O} = \emptyset$ and $\mathcal{M}$ is a GAV∩LAV mapping, and two D-datasets λ^+ and λ^- (resp., a D-dataset λ^+) containing only one unary tuple for which:*

- *Complete-VST*SEP*(∅, GAV∩LAV, CQ) and Complete-VST*CHAR*(∅, GAV∩LAV, CQ) are NP-hard;*
- *Sound-VST*SEP*(∅, GAV∩LAV, CQ) and Sound-VST*CHAR*(∅, GAV∩LAV, CQ) are coNP-hard;*
- *Proper-VST*SEP*(∅, GAV∩LAV, CQ) and Proper-VST*CHAR*(∅, GAV∩LAV, CQ)) are DP-hard.*

Proof. 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 $\mathcal{O}$ contains no assertions and whose alphabet consists of two atomic roles P_1 and P_2, $\mathcal{S} = \{s_1, s_2\}$, and $\mathcal{M}$ consists of the following two GAV∩LAV assertions: $\{(x_1, x_2) \mid s_1(x_1, x_2)\} \rightarrow \{(x_1, x_2) \mid P_1(x_1, x_2)\}$ and $\{(x_1, x_2) \mid s_2(x_1, x_2)\} \rightarrow \{(x_1, x_2) \mid P_2(x_1, x_2)\}$, which simply mirrors source predicate s_i to atomic role P_i, for $i \in [1, 2]$, and (ii) D is the $\mathcal{S}$-database composed of the following facts:

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

Let, moreover, λ^+ and λ^- be the fixed D-datasets $\lambda^+ = \{(c_4)\}$ and $\lambda^- = \{(c_3)\}$. Note that λ^- is needed only for the decision problems related to separability but not for the decision problems related to characterizability.

Let $G = (V, E)$ be a finite and undirected graph without loops[3] or isolated nodes, where $V = \{y_1, \ldots, y_n\}$. We define a CQ $q_G = \{(x) \mid \exists \vec{y}.\phi_{\mathcal{O}}(x, \vec{y})\}$ over $\mathcal{O}$ as follows:

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

Notice that q_G can be constructed in LogSpace from an input graph G as above.

By inspecting the OBDM system $\Sigma = \langle J, D\rangle$, for any graph G as above, the set of certain answers $cert^D_{q_G,J}$ must necessarily be an element of the power set of $\{(c_3), (c_4)\}$. More specifically, the following property holds:

Claim 1. *For both $i = 3$ and $i = 4$, we have that a graph $G = (V, E)$ is i-colourable if and only if $(c_i) \in cert^D_{q_G,J}$.*

Proof. First of all, notice that $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}$ is composed of the following facts:

$$\begin{aligned}&\big\{P_1(x, y) \mid x = \{r', g', b'\} \text{ and } y = \{r', g', b'\} \text{ and } x \neq y\big\}\\ &\quad \cup \big\{P_1(x, y) \mid x = \{r, g, b, o\} \text{ and } y = \{r, g, b, o\} \text{ and } x \neq y\big\}\\ &\quad \cup \big\{P_2(x, c_3) \mid x = \{r', g', b'\}\big\} \cup \big\{P_2(x, c_4) \mid x = \{r, g, b, o\}\big\}.\end{aligned}$$

"**Only-if part:**" Suppose $G = (V, E)$ is 3-colourable (resp., 4-colourable), that is, there exists a function $f : V \to \{r', g', b'\}$ (resp., $f : V \to \{r, g, b, o\}$) such that $f(y_i) \neq f(y_j)$ for each $(y_i, y_j) \in E$. Let $\phi_{\mathcal{O}}$ be the body of q_G, and consider the extension of f which assigns to the distinguished variable x of q_G the constant c_3 (resp., c_4). It can be readily seen that f consists in a homomorphism from $set(\phi_{\mathcal{O}})$ to $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}$ such that $f(x) = c_3$ (resp., $f(x) = c_4$). In other words, f witnesses that $(set(\phi_{\mathcal{O}}), (x)) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, (c_3))$ (resp., $(set(\phi_{\mathcal{O}}), (x)) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, (c_4))$). Thus, $(c_3) \in cert^D_{q_G,J}$ (resp., $(c_4) \in cert^D_{q_G,J}$), as required.

"**If part:**" Suppose $G = (V, E)$ is not 3-colourable (resp., not 4-colourable), that is, each possible function $f : V \to \{r', g', b'\}$ (resp., $f : V \to \{r, g, b, o\}$) is such that $f(y_i) = f(y_j)$ for some $(y_i, y_j) \in E$. Clearly, this implies that $(set(\phi_{\mathcal{O}}), (x)) \nrightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, (c_3))$ (resp., $(set(\phi_{\mathcal{O}}), (x)) \nrightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, (c_4))$). Thus, $(c_3) \notin cert^D_{q_G,J}$ (resp., $(c_4) \notin cert^D_{q_G,J}$), as required. □

With the above property in hand, and the fact that $cert^D_{q_G,J}$ is an element of the power set of $\{(c_3), (c_4)\}$ for each possible graph $G = (V, E)$, we are now ready to prove the claimed lower bounds.

As for the complete case, the proof of NP-hardness is by a LogSpace reduction from the *4-colourability problem*, which is NP-complete [32]. In particular, from the above claim a graph G is 4-colourable if and only if $(c_4) \in cert^D_{q_G,J}$, i.e., if and only if $\lambda^+ \subseteq cert^D_{q_G,J}$, which is the condition for q_G to be a complete Σ-separation of λ^+ and λ^- in CQ (resp., a complete Σ-characterization of λ^+ in CQ).

As for the sound case, the proof of coNP-hardness is by a LogSpace reduction from the *complement of the 3-colourability problem*, which is coNP-complete [32]. In particular, from the above claim a graph G is not 3-colourable if and only if $(c_3) \notin cert^D_{q_G,J}$, i.e., if and only if $cert^D_{q_G,J} \cap \lambda^- = \emptyset$ (resp., $cert^D_{q_G,J} \subseteq \lambda^+$), which is the condition for q_G to be a sound Σ-separation of λ^+ and λ^- in CQ (resp., a sound Σ-characterization of λ^+ in CQ).

Finally, as for the proper case, the proof of DP-hardness is by a LogSpace reduction from the *exact-4-colourability problem*, which is DP-complete [59]. In particular, a graph G is exact-4-colourable (i.e., 4-colourable and not 3-colourable) if and only if $cert^D_{q_G,J} = \{(c_4)\}$, i.e., if and only if $cert^D_{q_G,J} \subseteq \lambda^+$ and $cert^D_{q_G,J} \cap \lambda^- = \emptyset$ (resp., $cert^D_{q_G,J} = \lambda^+$), which is the condition for q_G to be a proper Σ-separation of λ^+ and λ^- in CQ (resp., a proper Σ-characterization of λ^+ in CQ). □

[3] In graph theory, a loop is an edge that connects a vertex with itself [8].

For the scenario under investigation in this paper, we can now establish the precise computational complexity of all the Verification decision problems introduced at the beginning of this section.

Corollary 2. *The following holds:*

- *Complete-V*SEP*(DL-Lite$_\mathcal{R}$, GLAV, UCQ) and Complete-V*CHAR*(DL-Lite$_\mathcal{R}$, GLAV, UCQ) are NP-complete. The lower bounds already hold for Complete-VST*SEP*(∅, GAV∩LAV, CQ) and Complete-VST*CHAR*(∅, GAV∩LAV, CQ);*
- *Sound-V*SEP*(DL-Lite$_\mathcal{R}$, GLAV, UCQ) and Sound-V*CHAR*(DL-Lite$_\mathcal{R}$, GLAV, UCQ) are coNP-complete. The lower bounds already hold for Sound-VST*SEP*(∅, GAV∩LAV, CQ) and Sound-VST*CHAR*(∅, GAV∩LAV, CQ);*
- *Proper-V*SEP*(DL-Lite$_\mathcal{R}$, GLAV, UCQ) and Proper-V*CHAR*(DL-Lite$_\mathcal{R}$, GLAV, UCQ) are DP-complete. The lower bounds already hold for Proper-VST*SEP*(∅, GAV∩LAV, CQ) and Proper-VST*CHAR*(∅, GAV∩LAV, CQ).*

Finally, from the lower bounds given in Theorem 3, we can derive two interesting novel results also in the context of explainability and definability in the plain relational database case. More specifically, since the fixed OBDM specification $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ used in the proof that theorem is such that $\mathcal{O} = \emptyset$ and $\mathcal{M}$ is both a GAV and a LAV mapping, the proof can be straightforwardly adapted also for the plain relational database case. Thus, given a schema $\mathcal{S}$, an $\mathcal{S}$-database D, two D-datasets λ^+ and λ^- (resp., a D-dataset λ^+), and a UCQ $q_\mathcal{S}$ over $\mathcal{S}$, it is DP-complete the problem of deciding whether $q_\mathcal{S}$ explains λ^+ and λ^- (resp., defines λ^+) inside D (the DP membership of these decision problems directly follows from Corollary 1).

6. The computation problem

In this section, we address the Computation problem. We start by considering the case when the given OBDM system Σ at hand is inconsistent. Given an inconsistent OBDM system $\Sigma = \langle J, D \rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ and two D-datasets λ^+ and λ^- (resp., a D-dataset λ^+) of arity n, we point out that any query $q_\mathcal{O} \in \mathcal{Q}$ of arity n over $\mathcal{O}$ is a $\mathcal{Q}$-minimally complete Σ-separation (resp., Σ-characterization) of λ^+ and λ^- (resp., of λ^+). This is so because, by definition, the certain answers of any query $q_\mathcal{O}$ of arity n w.r.t. an inconsistent OBDM system Σ is the set of all possible n-tuples of constants occurring in D, i.e., $dom(D)^n$. Furthermore, if $\lambda^+ = dom(D)^n$ and $\lambda^- = \emptyset$, then any query $q_\mathcal{O} \in \mathcal{Q}$ of arity n over $\mathcal{O}$ is also a $\mathcal{Q}$-maximally sound (and therefore a proper in $\mathcal{Q}$) Σ-separation (resp., Σ-characterization) of λ^+ and λ^- (resp., of λ^+); otherwise, no sound (and therefore, no $\mathcal{Q}$-maximally sound and no proper in $\mathcal{Q}$) Σ-separation (resp., Σ-characterization) of λ^+ and λ^- (resp., of λ^+) exists. Since, however, for OBDM systems of our scenario it is always possible to check whether they are inconsistent or not (cf. Section 3), from the above observations one can trivially derive suitable algorithms for the Computation problem in all the cases in which the input OBDM system Σ is inconsistent.

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

Specifically, we now provide two algorithms that, given a consistent OBDM system $\Sigma = \langle J, D \rangle$ and two D-datasets λ^+ and λ^- (resp., a D-dataset λ^+), always terminate and return, respectively, a UCQ-minimally complete and a UCQ-maximally sound Σ-separation (resp., Σ-characterization) of λ^+ and λ^- (resp., of λ^+). This proves that, in our investigated scenario, they always exist and can be computed. In fact, the algorithms we provide focus only on the Separability case. Algorithms for the Characterizability case can be immediately derived from the ones we provide by simply computing $\lambda^- = dom(D)^n \setminus \lambda^+$, where n is the arity of the tuple(s) in λ^+.

Before delving into the details of the two algorithms, we provide some crucial properties about the canonical structure that will be used to establish the correctness of such algorithms.

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

Proof. If Σ is inconsistent, then the claim is trivial. If Σ is consistent, from Section 3 we know that $\vec{c} \in cert^D_{q_\mathcal{O},J}$ implies the existence of a disjunct $q = \{\vec{x} \mid \exists \vec{y}.\phi(\vec{x}, \vec{y})\}$ in $q_\mathcal{O}$ for which $(set(\phi), \vec{x}) \rightarrow (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{c})$. Let h be

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

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

Proof. Suppose that $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{c}) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$, and let h be the homomorphism witnessing this. 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 $(set(\phi), \vec{x}) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$, 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 $(set(\phi), \vec{x}) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$, 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 $(\mathcal{M}(D), \vec{c}) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$. By construction of the canonical structure $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}$, it is now trivial to verify that h' can be properly extended into a homomorphism h'' witnessing that $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{c}) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$. □

We are now ready to present our algorithms for the Computation problem. We start with the complete case, and provide the Algorithm 1 (MinCompSeparation) for computing UCQ-minimally complete separations.

Informally, for each positive example $\vec{c}_i \in \lambda^+$, the algorithm obtains from the set of atoms $\mathcal{M}(D)$ the CQ $query(\mathcal{M}(D), \vec{c}_i)$. Then, the output query $q_{\mathcal{O}}$ is the union of all the CQs obtained in such a way.

Example 7. Let $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ be the same OBDM specification of Example 2. 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-datasets $\lambda^+ = \{(c_1), (c_2)\}$ and $\lambda^- = \{(c_3)\}$, MinCompSeparation($\langle J, D\rangle$, λ^+,λ^-) returns the UCQ $q_{\mathcal{O}} = query(\mathcal{M}(D), (c_1)) \cup query(\mathcal{M}(D), (c_2))$, where $query(\mathcal{M}(D), (c_1)) = \{(x_{c_1}) \mid \exists y_{c_2}, y_{c_3}, y_b.\mathsf{Student}(x_{c_1}) \wedge \mathsf{EnrolledIn}(y_{c_2}, y_b) \wedge \mathsf{EnrolledIn}(y_{c_3}, y_b)\}$ and $query(\mathcal{M}(D), (c_2)) = \{(x_{c_2}) \mid \exists y_{c_1}, y_{c_3}, y_b.\mathsf{EnrolledIn}(x_{c_2}, y_b) \wedge \mathsf{EnrolledIn}(y_{c_3}, y_b) \wedge \mathsf{Student}(y_{c_1})\}$. Note that the query $q_{\mathcal{O}}$ returned by the algorithm is a UCQ-minimally complete Σ-separation of λ^+ and λ^-, where $\Sigma = \langle J, D\rangle$.

The following theorem establishes termination and correctness of the Algorithm 1 (MinCompSeparation).

Theorem 4. *Let $\Sigma = \langle J, D\rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ be a consistent OBDM system and λ^+ and λ^- be two D-datasets. We have that MinCompSeparation(Σ, λ^+,λ^-) terminates and returns a UCQ-minimally complete Σ-separation of λ^+ and λ^-.*

Proof. Termination of the algorithm as well as completeness of the UCQ $q_{\mathcal{O}}$ returned are straightforward. In particular, due to Proposition 5, it is obvious that each $\vec{c}_i \in \lambda^+$ is such that $\vec{c}_i \in cert^{D}_{q_{\vec{c}_i}, J}$, where $q_{\vec{c}_i} = query(\mathcal{M}(D), \vec{c}_i)$.

To prove that $q_{\mathcal{O}}$ is also a UCQ-minimally complete Σ-separation of λ^+ and λ^-, note that it is enough to show that any query q over $\mathcal{O}$ that is a complete Σ-separation of λ^+ and λ^- in UCQ is such that $cert^{D}_{q_{\mathcal{O}}, J} \subseteq cert^{D}_{q, J}$.

Algorithm 1 MinCompSeparation

Input: Consistent OBDM system $\Sigma = \langle J, D\rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$; D-dataset $\lambda^+ = \{\vec{c}_1, \ldots, \vec{c}_m\}$; D-dataset λ^-
Output: UCQ $q_{\mathcal{O}}$ over $\mathcal{O}$

1: Compute $\mathcal{M}(D)$
2: $q_{\mathcal{O}} \leftarrow query(\mathcal{M}(D), \vec{c}_1) \cup \cdots \cup query(\mathcal{M}(D), \vec{c}_m)$
3: **return** $q_{\mathcal{O}}$

Algorithm 2 MaxSoundSeparation

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; D-dataset λ^- of arity n

Output: UCQ $q_{\mathcal{O}}$ over $\mathcal{O}$

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

We do this by contraposition. Let q be a UCQ over $\mathcal{O}$ 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 $\vec{b} \in cert^D_{q_{\vec{c_i}},J}$ for some $q_{\vec{c_i}} = query(\mathcal{M}(D), \vec{c_i})$ with $\vec{c_i} \in \lambda^+$ occurring in $q_{\mathcal{O}}$. By Proposition 5, one can see that $\vec{b} \in cert^D_{q_{\vec{c_i}},J}$ implies $(\mathcal{C}^{\mathcal{M}(D)}_{\mathcal{O}}, \vec{c_i}) \rightarrow (\mathcal{C}^{\mathcal{M}(D)}_{\mathcal{O}}, \vec{b})$. By Proposition 4, it follows that each UCQ q' over $\mathcal{O}$ containing the tuple $\vec{c_i}$ in its set of certain answers w.r.t. Σ must contain also tuple $\vec{b}$ in such a set, i.e., $\vec{c_i} \in cert^D_{q',J}$ implies $\vec{b} \in cert^D_{q',J}$ for any UCQ q' over $\mathcal{O}$. Thus, since $\vec{b} \notin cert^D_{q,J}$, we derive that $\vec{c_i} \notin cert^D_{q,J}$ as well. Now, since $\vec{c_i} \in \lambda^+$ and $\vec{c_i} \notin cert^D_{q,J}$, this immediately implies that q is not a complete Σ-separation of λ^+ and λ^- in UCQ, as required. □

We now turn to the sound case, and provide the Algorithm 2 (MaxSoundSeparation) for computing UCQ-maximally sound Σ-separations.

Intuitively, starting from the query $query(\mathcal{M}(D), \vec{c_1}) \cup \cdots \cup query(\mathcal{M}(D), \vec{c_m})$, which is a UCQ-minimally complete Σ-separation of λ^+ and λ^-, the algorithm simply discards all those disjuncts whose set of certain answers w.r.t. Σ contain at least a tuple $\vec{b} \in \lambda^-$. We recall from Section 3 that the set of certain answers of a CQ $q_{\vec{c_i}}$ w.r.t. a consistent OBDM system $\Sigma = \langle J, D\rangle$ can be always computed by first obtaining its reformulation $\mathsf{rew}_{q_{\vec{c_i}},J}$ over the source schema $\mathcal{S}$, and then by evaluating this latter query directly over the $\mathcal{S}$-database D. In other words, the if-condition of line 5 can be equivalently reformulated as: $\mathsf{rew}^D_{q_{\vec{c_i}},J} \cap \lambda^- = \emptyset$.

Example 8. Refer to Example 7. Since the certain answers of the CQ $query(\mathcal{M}(D), (c_2))$ w.r.t. $\Sigma = \langle J, D\rangle$ include also $(c_3) \in \lambda^-$, MaxSoundSeparation(Σ, λ^+,λ^-) returns the CQ $q_{\mathcal{O}} = query(\mathcal{M}(D), (c_1))$. Note that $q_{\mathcal{O}}$ is a UCQ-maximally sound Σ-separation of λ^+ and λ^-.

The following theorem establishes termination and correctness of the Algorithm 2 (MaxSoundSeparation).

Theorem 5. *Let $\Sigma = \langle J, D\rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ be a consistent OBDM system, and λ^+ and λ^- be two D-datasets. We have that MaxSoundSeparation(Σ, λ^+, λ^-) terminates and returns a UCQ-maximally sound Σ-separation of λ^+ and λ^-.*

Proof. Termination of the algorithm as well as soundness of the UCQ $q_{\mathcal{O}}$ returned are straightforward. In particular, by construction, all the disjuncts $q_{\vec{c_i}}$ of $q_{\mathcal{O}}$ are such that there is no tuple $\vec{b} \in \lambda^-$ for which $\vec{b} \in cert^D_{q_{\vec{c_i}},J}$.

To prove that $q_{\mathcal{O}}$ is also a UCQ-maximally sound Σ-separation of λ^+ and λ^-, note that it is enough to show that any query q over $\mathcal{O}$ that is a sound Σ-separation of λ^+ and λ^- in UCQ is such that $cert^D_{q,J} \cap \lambda^+ \subseteq cert^D_{q_{\mathcal{O}},J} \cap \lambda^+$. We do this by contraposition. Let q be a UCQ over $\mathcal{O}$ for which $cert^D_{q,J} \cap \lambda^+ \not\subseteq cert^D_{q_{\mathcal{O}},J} \cap \lambda^+$, i.e., for a tuple of constants $\vec{b} \in \lambda^+$ we have $\vec{b} \in cert^D_{q,J}$ but $\vec{b} \notin cert^D_{q_{\mathcal{O}},J}$. Since $\vec{b} \notin cert^D_{q_{\mathcal{O}},J}$ 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 $\vec{b} \in cert^D_{q_{\mathcal{O}},J}$, which is a contradiction to the fact that $\vec{b} \notin cert^D_{q_{\mathcal{O}},J}$). By construction of 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 $\vec{g} \in \lambda^-$. By Proposition 5, one can see that $\vec{g} \in cert^D_{q_{\vec{b}},J}$ implies $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b}) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{g})$. By Proposition 4, it follows that each UCQ q' over $\mathcal{O}$ containing tuple $\vec{b}$ in its set of certain answers w.r.t. Σ must contain also tuple $\vec{g}$ in such a set, i.e., $\vec{b} \in cert^D_{q',J}$ implies $\vec{g} \in cert^D_{q',J}$ for any UCQ q' over $\mathcal{O}$. Thus, since $\vec{b} \in cert^D_{q,J}$, we derive that $\vec{g} \in cert^D_{q,J}$ as well. Now, since $\vec{g} \in \lambda^-$ and $\vec{g} \in cert^D_{q,J}$, this immediately implies that q is not a sound Σ-separation of λ^+ and λ^- in UCQ, as required. □

We now briefly discuss the running time of the two above algorithms. First, we observe that the running time of the presented algorithms differs depending on which mapping language the input mapping $\mathcal{M}$ is formulated. A key difference between the mapping languages GLAV and the special cases GAV and LAV, is in the size of the computed $\mathcal{M}(D)$. In GLAV mappings, due to the simultaneous presence of queries involving multiple source predicates in the left-hand side of the mapping assertions, and join existential variables in the right-hand side of the assertions, the set of atoms $\mathcal{M}(D)$ can be exponentially large. For example, considering a database $D = \{s_i(0), s_i(1) \mid 1 \leqslant i \leqslant n\}$ and the mappings $\mathcal{M}$ containing the GLAV assertion: $\{(x_1, \ldots, x_n) \mid s_1(x_1) \wedge \cdots \wedge s_n(x_n)\} \rightarrow \{(x_1, \ldots, x_n) \mid \exists y.P(x_1, y) \wedge \cdots \wedge P(x_n, y)\}$, the number of atoms occurring in the set $\mathcal{M}(D)$ is 2^n. Conversely, both in LAV and GAV mappings, $\mathcal{M}(D)$ is always polynomially bounded, since the former does not allow for multiple source predicates in the left-hand side of mapping assertions, whereas the latter does not allow for existential variables in the right-hand side of mapping assertions and the arity of ontology predicates is fixed to at most 2.

Furthermore, the running time required to compute $\mathcal{M}(D)$ is different in the LAV and GAV cases. Indeed $\mathcal{M}(D)$ can always be computed in polynomial time whenever $\mathcal{M}$ is a LAV mapping, whereas, in general, already when $\mathcal{M}$ is a GAV mapping, since there are CQs in the left-hand side of mapping assertions that need to be evaluated over the database D, it takes exponential time to deterministically compute $\mathcal{M}(D)$ (unless PTIME = NP).

As immediate consequences of the above observations, we derive that, in general, the Algorithm 1 (MinCompleteSeparation) runs in exponential time with respect to the size of the input, while it runs in polynomial time whenever the mapping $\mathcal{M}$ of the input OBDM system is a LAV mapping. As for the case of the Algorithm 2 (MaxSoundSeparation), since after computing $\mathcal{M}(D)$ one needs also to compute the certain answers of each $query(\mathcal{M}(D), \vec{c}_i)$ for all $\vec{c}_i$ in the input dataset λ^+, and deterministically computing the certain answers of CQs with respect to an OBDM system takes exponential time (unless PTIME = NP), we derive that, in general, the algorithm runs in double exponential time with respect to the size of the input, while it runs in exponential time whenever the mapping $\mathcal{M}$ of the input OBDM system is either a GAV or a LAV mapping. This is because $\mathcal{M}(D)$ is guaranteed to be of polynomial size whenever $\mathcal{M}$ is either a GAV or a LAV mapping (and therefore, also the various CQs $query(\mathcal{M}(D), \vec{c}_i)$ are of polynomial size with respect to the size of the input).

We conclude this section by providing a semantic test for the existence of proper separations and characterizations in UCQ in the OBDM settings. We observe that, in all the cases in which a proper separation exists, it is clear that both algorithms return the same query $query(\mathcal{M}(D), \vec{c_1}) \cup \cdots \cup query(\mathcal{M}(D), \vec{c_m})$. Therefore, as a direct consequence of both Theorem 4 and Theorem 5, we get the following result.

Corollary 3. *Let $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$ be an OBDM specification, and $\lambda^+ = \{\vec{c_1}, \ldots, \vec{c_m}\}$ and λ^- two D-datasets. Either the UCQ $q_{\mathcal{O}} = query(\mathcal{M}(D), \vec{c_1}) \cup \cdots \cup query(\mathcal{M}(D), \vec{c_m})$ is a proper Σ-separation of λ^+ and λ^- in UCQ, or no proper Σ-separation of λ^+ and λ^- in UCQ exists.*

The combination of Corollary 3 and Proposition 5 allows us to provide the following semantic tests for the existence of proper separations and proper characterizations in UCQ in the OBDM setting, which can be seen as the analogous of the semantic tests given in [7] and [55] for the plain relational database setting and the ontology-enriched query answering setting, respectively.

- SEP *test for UCQs in OBDM*: given a consistent OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$ and two D-datasets λ^+ and λ^-, there exists a proper Σ-separation of λ^+ and λ^- in UCQ if and only if it is the case that $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{c}) \nrightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$ for all $\vec{c} \in \lambda^+$ and all $\vec{b} \in \lambda^-$.

– CHAR *test for UCQs in OBDM*: given a consistent OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$ and a D-dataset λ^+ of arity n, there exists a proper Σ-characterization of λ^+ in UCQ if and only if it is the case that $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{c}) \not\rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$ for all $\vec{c} \in \lambda^+$ and all $\vec{b} \in dom(D)^n \setminus \lambda^+$.

7. The existence problem

We now address the existence problem. First of all, for the scenario under consideration in this paper, the existence problem for both UCQ-minimally complete and UCQ-maximally sound separations (and also characterizations) is trivial, since by Theorems 4 and 5 they always exist. Thus, in this section we only consider the remaining proper case, by defining a variant of the decision problems as defined in [55], where also a mapping in some mapping language $\mathcal{L}_{\mathcal{M}}$ is given as input.

SEP($\mathcal{L}_{\mathcal{O}}$, $\mathcal{L}_{\mathcal{M}}$, $\mathcal{Q}$)

Input: An OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$ and two D-datasets λ^+ and λ^-, where $\mathcal{O} \in \mathcal{L}_{\mathcal{O}}$ and $\mathcal{M} \in \mathcal{L}_{\mathcal{M}}$.
Question: Does there exist a proper Σ-separation of λ^+ and λ^- in $\mathcal{Q}$?

CHAR($\mathcal{L}_{\mathcal{O}}$, $\mathcal{L}_{\mathcal{M}}$, $\mathcal{Q}$)

Input: An OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$ and a D-dataset λ^+, where $\mathcal{O} \in \mathcal{L}_{\mathcal{O}}$ and $\mathcal{M} \in \mathcal{L}_{\mathcal{M}}$.
Question: Does there exist a proper Σ-characterization of λ^+ in $\mathcal{Q}$?

We also introduce two important special cases of the above decision problems, namely: STSEP($\mathcal{L}_{\mathcal{O}}$, $\mathcal{L}_{\mathcal{M}}$, $\mathcal{Q}$) and STCHAR($\mathcal{L}_{\mathcal{O}}$, $\mathcal{L}_{\mathcal{M}}$, $\mathcal{Q}$), which are special cases of SEP($\mathcal{L}_{\mathcal{O}}$, $\mathcal{L}_{\mathcal{M}}$, $\mathcal{Q}$) and CHAR($\mathcal{L}_{\mathcal{O}}$, $\mathcal{L}_{\mathcal{M}}$, $\mathcal{Q}$), respectively, in which the all the input D-datasets are singleton sets (i.e., they consist of just a single tuple).

In what follows, we show that the computational complexity of the above decision problems differ only depending on the mapping language $\mathcal{L}_{\mathcal{M}}$ adopted. As already discussed in Section 6, a key difference between GLAV and the special cases GAV and LAV is both in the size of the computed $\mathcal{M}(D)$, i.e. exponential for GLAV mapping and polynomial for both GAV and LAV mapping, and in the effort required to compute $\mathcal{M}(D)$, i.e. polynomial time for LAV mapping and exponential time (unless PTIME = NP) in both GAV and GLAV mapping.

We start by characterizing the computational complexity of SEP and CHAR (and their respective subproblems) in the simplest LAV case, then we address the GAV case, and finally we focus on the most general GLAV case. Interestingly, all the provided matching lower bounds hold even for fixed ontologies $\mathcal{O} = \emptyset$, i.e., ontologies without assertions, and fixed D-datasets containing only a single unary tuple.

Importantly, for the scenario under consideration, from the results of the previous section, the questions in SEP and CHAR can be reformulated equivalently as follows: "is $q_{\mathcal{O}} = query(\mathcal{M}(D), \vec{c_1}) \cup \cdots \cup query(\mathcal{M}(D), \vec{c_m})$ also a sound (and so, a proper) Σ-separation of $\lambda^+ = \{\vec{c_1}, \ldots, \vec{c_m}\}$ and λ^- in UCQ?" and "is $q_{\mathcal{O}} = query(\mathcal{M}(D), \vec{c_1}) \cup \cdots \cup query(\mathcal{M}(D), \vec{c_m})$ also a sound (and so, a proper) Σ-characterization of $\lambda^+ = \{\vec{c_1}, \ldots, \vec{c_m}\}$ in UCQ?", respectively.

Theorem 6. *SEP(DL-Lite$_{\mathcal{R}}$, LAV, UCQ) and CHAR(DL-Lite$_{\mathcal{R}}$, LAV, UCQ) are coNP-complete. The lower bounds already hold for STSEP(∅, GAV∩LAV, UCQ) and STCHAR(∅, GAV∩LAV, UCQ).*

Proof. **Upper bound:** We only mention SEP(*DL-Lite*$_{\mathcal{R}}$, LAV, UCQ). The CHAR(*DL-Lite*$_{\mathcal{R}}$, LAV, UCQ) case is similar and therefore not discussed. Given an OBDM system $\Sigma = \langle J, D \rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ in which $\mathcal{O}$ is a *DL-Lite*$_{\mathcal{R}}$ ontology and $\mathcal{M}$ is a LAV mapping, and two D-datasets $\lambda^+ = \{\vec{c_1}, \ldots, \vec{c_m}\}$ and λ^-, we first compute $\mathcal{M}(D)$ in polynomial time (recall that $\mathcal{M}$ is a LAV mapping), and therefore also the query $q_{\mathcal{O}} = query(\mathcal{M}(D), \vec{c_1}) \cup \cdots \cup query(\mathcal{M}(D), \vec{c_m})$ which is a UCQ-minimally complete Σ-separation of λ^+ and λ^-. Then, exactly as illustrated

in Theorem 2, we can check in coNP whether $query(\mathcal{M}(D), \vec{c_1}) \cup \cdots \cup query(\mathcal{M}(D), \vec{c_m})$ is also a sound (and so, a proper) Σ-separation of λ^+ and λ^- in UCQ.

Lower bound: We start with STSEP($\emptyset$, GAV∩LAV, UCQ), and then we address the STCHAR($\emptyset$, GAV∩LAV, UCQ) case. The proof of coNP-hardness is by a LOGSPACE reduction from the complement of the *3-colourability problem*. We define a fixed OBDM specification $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M} \rangle$ in which $\mathcal{O}$ contains no assertions and whose alphabet consists of an atomic role e and an atomic concept A, $\mathcal{S} = \{s_e, s\}$, and $\mathcal{M}$ consists of the following two GAV∩LAV mapping assertions: $\{(x_1, x_2) \mid s_e(x_1, x_2)\} \rightarrow \{(x_1, x_2) \mid e(x_1, x_2)\}$ and $\{(x) \mid s(x)\} \rightarrow \{(x) \mid A(x)\}$, which simply mirrors source predicates s_e and s to e and A, respectively.

Let $G = (V, E)$ be a finite and undirected graph without loops or isolated nodes, where $V = \{c_1, \ldots, c_n\}$. Without loss of generality, we may assume that $V \neq \emptyset$ and that G is *connected*, i.e., there is a path from c to c' for any pair of nodes $(c, c') \in V^2$. Then, we define an $\mathcal{S}$-database D_G composed of the following facts:

$$\big\{s_e\big(c, c'\big) \mid \big(c, c'\big) \in E\big\} \cup \big\{s(c_1)\big\}$$
$$\cup \big\{s_e(x, y) \mid x = \{r, g, b\} \text{ and } y = \{r, g, b\} \text{ and } x \neq y\big\} \cup \big\{s(r)\big\}.$$

Let, moreover, λ^+ and λ^- be the fixed D_G-datasets $\lambda^+ = \{(c_1)\}$ and $\lambda^- = \{(r)\}$.

Notice that the $\mathcal{S}$-database D_G can be constructed in LOGSPACE from an input graph G as above. Let the OBDM system be $\Sigma_G = \langle J, D_G \rangle$. We now show that a graph G is not 3-colourable if and only if there exists a proper Σ_G-separation of λ^+ and λ^- in UCQ, thus proving the claimed lower bound.

Claim 2. *Given a graph G, there exists a proper Σ_G-separation of λ^+ and λ^- in UCQ if and only if G is not 3-colourable.*

Proof. First of all, note that $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D_G)} = \{e(c, c') \mid (c, c') \in E\} \cup \{A(c_1)\} \cup \{e(x, y) \mid x = \{r, g, b\} \text{ and } y = \{r, g, b\} \text{ and } x \neq y\} \cup \{A(r)\}$. We recall that a proper Σ_G-separation of λ^+ and λ^- in UCQ exists if and only if the CQ $q_G = query(\mathcal{M}(D_G), (c_1))$ is also a sound (and so, a proper) Σ_G-separation of λ^+ and λ^- in UCQ.

"**Only-if part:**" Suppose $G = (V, E)$ is 3-colourable, that is, there exists a function $f : V \rightarrow \{r, g, b\}$ such that $f(c) \neq f(c')$ for each $(c, c') \in E$. Without loss of generality, we may assume that $f(c_1) = r$ (indeed, the existence of f clearly implies the existence of a function f' with $f'(c_1) = r$ and such that $f'(c) \neq f'(c')$ for each $(c, c') \in E$ holds as well). Consider the extension h of f assigning $h(x) = x$ to each $x \in \{r, g, b\}$. It can be readily seen that h consists in a homomorphism witnessing that $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D_G)}, (c_1)) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D_G)}, (r))$, which directly implies that $(r) \in cert_{q_G, J}^{D_G}$. It follows that q_G is not a sound Σ_G-separation of λ^+ and λ^- in UCQ, and therefore no proper Σ_G-separation of λ^+ and λ^- in UCQ exists.

"**If part:**" Suppose there exists no proper Σ_G-separation of λ^+ and λ^- in UCQ, i.e., the CQ q_G is such that $(r) \in cert_{q_G, J}^{D_G}$. It follows that there exists a homomorphism h witnessing that $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D_G)}, (c_1)) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D_G)}, (r))$. Now, since the graph G is connected and since $h(c_1) = h(r)$, by construction of $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D_G)}$ the homomorphism h must necessarily be such that $h(c) \in \{r, g, b\}$ for each constant c representing a node $c \in V$. But then, due to the fact that none of $e(r, r)$, $e(g, g)$, and $e(b, b)$ occur in $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D_G)}$, we derive that h is such that $h(c) \neq h(c')$ for each $e(c, c') \in \mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D_G)}$, and therefore for each $(c, c') \in E$ as well. Thus, we can conclude that G is 3-colourable. □

As for the coNP-hardness of STCHAR($\emptyset$, GAV∩LAV, UCQ), it is possible to use exactly the same reduction provided above by discarding λ^- and considering only $\lambda^+ = \{(c_1)\}$. In particular, due to the fact that $A(c)$ and $A(r)$ are the only A-facts occurring in $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D_G)}$, the set of certain answers of the query q_G w.r.t. Σ_G is always either $\{(c_1)\}$ or $\{(c_1), (r)\}$, and which one of the two depends on the 3-colourability of G. □

We now turn to consider GAV mappings. We 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 oracle for a decision problem in* NP [12] (we refer the reader to [67] for further characterizations of such complexity class). By a round of parallel queries, it is intended that the Turing machine can ask for polynomially many *non-adaptive queries* to the NP oracle.

Theorem 7. SEP(*DL-Lite*$_\mathcal{R}$, GAV, UCQ) *and* CHAR(*DL-Lite*$_\mathcal{R}$, GAV, UCQ) *are* Θ_2^p*-complete. The lower bounds already hold for ST*SEP(∅, GAV, UCQ) *and ST*CHAR(∅, GAV, UCQ).

Proof. **Upper bound:** We only mention SEP(*DL-Lite*$_\mathcal{R}$, GAV, UCQ). The CHAR(*DL-Lite*$_\mathcal{R}$, GAV, UCQ) case is similar and therefore not discussed. Given an OBDM system $\Sigma = \langle J, D\rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$ in which $\mathcal{O}$ is a *DL-Lite*$_\mathcal{R}$ ontology and $\mathcal{M}$ is a GAV mapping, and two D-datasets $\lambda^+ = \{\vec{c_1}, \ldots, \vec{c_m}\}$ and λ^-, as a first step we compute $\mathcal{M}(D)$ in polynomial time with a single round of parallel queries to an NP oracle. More specifically, 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., $A(c) \in \mathcal{M}(D)$). It is clear that deciding whether $P(c_1, c_2) \in \mathcal{M}(D)$ (resp., $A(c) \in \mathcal{M}(D)$) for a given pair of constants $(c_1, c_2) \in dom(D)^2$ (resp., constant $c \in dom(D)$), a GAV mapping $\mathcal{M}$, and a database D is an NP-complete problem because the left-hand side of mapping assertions are CQs [1].

Once obtained $\mathcal{M}(D)$ as described above, we construct in polynomial time the UCQ $q_\mathcal{O} = query(\mathcal{M}(D), \vec{c_1}) \cup \cdots \cup query(\mathcal{M}(D), \vec{c_m})$. Then, due to Theorem 2, with a second and final round of parallel queries, we can ask with a single query to an NP oracle whether $q_\mathcal{O}$ is also a sound (and so, a proper) Σ-separation of $\lambda^+ = \{\vec{c_1}, \ldots, \vec{c_m}\}$ and λ^- in UCQ.

Lower bound: We start with STSEP(∅, GAV, UCQ), and then we address the STCHAR(∅, GAV, UCQ) case. The proof of Θ_2^p-hardness is by a LOGSPACE reduction from the *odd clique problem*, which is a Θ_2^p-complete problem [66]. *Odd clique* is the problem of deciding, given a finite and 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).

Given a graph $G = (V, E)$ as above with $V = \{v_1, \ldots, v_n\}$, 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 in which $\mathcal{O}$ contains no assertions, $\mathcal{S}_G = \{e, s_1, \ldots, s_n\}$, and, for each odd $i \in [1, n]$, the mapping $\mathcal{M}_G$ has the following two GAV assertions:

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

where A_i is an atomic concept in the alphabet of $\mathcal{O}$ and, for each natural number p, cl_p is the conjunction of atoms:

$$cl_p = \bigwedge_{\{(k,j) \mid 1 \leqslant k < j \leqslant p\}} e(y_k, y_j)$$

.

Intuitively, cl_p asks whether G contains a clique of size p. Finally, D_G is the $\mathcal{S}_G$-database $D_G = \{e(x_1, x_2) \mid (x_1, x_2) \in E\} \cup \{e(x_2, x_1) \mid (x_1, x_2) \in E\} \cup \{s_i(c) \mid 1 \leqslant i \leqslant n \text{ and } i \text{ is odd}\} \cup \{s_i(c') \mid 2 \leqslant i \leqslant n \text{ and } i \text{ is even}\}$. Let, moreover, λ^+ and λ^- be the fixed D_G-datasets $\lambda^+ = \{(c)\}$ and $\lambda^- = \{(c')\}$.

Notice that λ^+ and λ^- are fixed, whereas the OBDM system Σ_G can be constructed in LOGSPACE from an input graph G as above.

The correctness of the reduction is mainly based on the next property:

Claim 3. *Let* $i \in [1, n]$ *be an odd number. We have that:*

1. $A_i(c) \in \mathcal{C}_\mathcal{O}^{\mathcal{M}_G(D_G)}$ *if and only if* G *contains a clique of size* i.
2. $A_i(c') \in \mathcal{C}_\mathcal{O}^{\mathcal{M}_G(D_G)}$ *if and only if* G *contains a clique of size* $i + 1$.

Proof. As for 1, since $s_i(c) \in D_G$, it is easy to see that the query $q_i = \{(x) \mid \exists y_1, \ldots, y_i . s_i(x) \wedge cl_i\}$ is such that $(c) \in q_i^{D_G}$ if and only if G has a clique of size i. Thus, due to the GAV assertion $q_i \rightarrow \{(x) \mid A_i(x)\}$ occurring in $\mathcal{M}_G$, we have $A_i(c) \in \mathcal{C}_\mathcal{O}^{\mathcal{M}_G(D_G)}$ if and only if G has a clique of size i.

As for 2, since $s_{i+1}(c') \in D_G$, it is easy to see that the query $q_{i+1} = \{(x) \mid \exists y_1, \ldots, y_{i+1} . s_{i+1}(x) \wedge cl_{i+1}\}$ is such that $(c') \in q_{i+1}^{D_G}$ if and only if G has a clique of size $i + 1$. Thus, due to the GAV assertion $q_{i+1} \rightarrow \{(x) \mid A_i(x)\}$

occurring in $\mathcal{M}_G$, if G has a clique of size $i+1$, then $A_i(c') \in \mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)}$. Conversely, suppose that G has not a clique of size $i+1$. On the one hand, the assertion $q_{i+1} \rightarrow \{(x) \mid A_i(x)\}$ does not make $A_i(c')$ true in $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)}$. On the other hand, since $s_i(c') \notin D_G$, not even the assertion $\{(x) \mid \exists y_1, \ldots, y_i.s_i(x) \wedge cl_i\} \rightarrow \{(x) \mid A_i(x)\}$ makes $A_i(c')$ true in $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)}$. Thus, $A_i(c') \notin \mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)}$. □

With the above property in hand, we can now prove that the maximum clique size of a graph G is an odd number if and only if the CQ $q_{\mathcal{O}} = query(\mathcal{M}_G(D_G), (c))$ is also a sound (and so, a proper) Σ_G-separation of λ^+ and λ^- in UCQ, thus showing the claimed lower bound.

Claim 4. *The maximum clique size of a graph G is an odd number if and only if the CQ $q_{\mathcal{O}} = query(\mathcal{M}_G(D_G), (c))$ is also a sound (and so, a proper) Σ_G-separation of λ^+ and λ^- in UCQ.*

Proof. "**Only-if part:**" Suppose that the maximum clique size of G is p, where p is an odd number. Due to Claim 3, we have that $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)} = \{A_1(c), A_1(c'), A_3(c), A_3(c'), \ldots, A_p(c)\}$. Observe that $A_p(c') \notin \mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)}$ because G has not a clique of size $p+1$ by assumption. Thus, $q_{\mathcal{O}} = query(\mathcal{M}_G(D_G), (c)) = \{(x_c) \mid \exists y_{c'}.\phi_{\mathcal{O}}(x_c, y_{c'})\}$, where $\phi_{\mathcal{O}}(x_c, y_{c'}) = A_1(x_c) \wedge A_1(y_{c'}) \wedge A_3(x_c) \wedge A_3(y'_c) \wedge \cdots \wedge A_p(x_c)$. It is straightforward to verify that $(set(\phi_{\mathcal{O}}), (x_c)) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)}, (c))$ but $(set(\phi_{\mathcal{O}}), (x_c)) \nrightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)}, (c'))$. It follows that $cert_{q_{\mathcal{O}}, J_G}^{D_G} = \{(c)\}$, i.e., $q_{\mathcal{O}}$ is a proper Σ_G-separation of λ^+ and λ^- in UCQ.

"**If part:**" Suppose that the maximum clique size of G is r, where r is an even number. Due to Claim 3, we have that $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)} = \{A_1(c), A_1(c'), A_3(c), A_3(c'), \ldots, A_{r-1}(c), A_{r-1}(c')\}$. Observe that $A_{r-1}(c') \in \mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)}$ and $A_{r+1}(c) \notin \mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)}$ because by assumption G has a clique of size r but not of size $r+1$. Thus, $q_{\mathcal{O}} = query(\mathcal{M}_G(D_G), (c)) = \{(x_c) \mid \exists y_{c'}.\phi_{\mathcal{O}}(x_c, y_{c'})\}$, where $\phi_{\mathcal{O}}(x_c, y_{c'}) = A_1(x_c) \wedge A_1(y_{c'}) \wedge A_3(x_c) \wedge A_3(y'_c) \wedge \cdots \wedge A_{r-1}(x_c) \wedge A_{r-1}(y_{c'})$. It is straightforward to verify that $(set(\phi_{\mathcal{O}}), (x_c)) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}_G(D_G)}, (c'))$. It follows that $(c') \in cert_{q_{\mathcal{O}}, J_G}^{D_G}$, and therefore $q_{\mathcal{O}}$ is not a sound (and so, not a proper) Σ_G-separation of λ^+ and λ^- in UCQ. □

As for the Θ_2^p-hardness of STCHAR(∅, GAV, UCQ), it is possible to use exactly the same reduction provided above by discarding λ^- and considering only $\lambda^+ = \{(c)\}$. In particular, the set of certain answers of the query $q_{\mathcal{O}} = query(\mathcal{M}_G(D_G), (c))$ w.r.t. Σ_G is always either $\{(c)\}$ or $\{(c), (c')\}$, and which one of the two depends on the parity of the maximum clique size of G. □

We now consider the remaining more general case of GLAV mappings.

Theorem 8. SEP*(DL-Lite$_{\mathcal{R}}$, GLAV, UCQ) and* CHAR*(DL-Lite$_{\mathcal{R}}$, GLAV, UCQ) are Π_3^p-complete. The lower bounds already hold for ST*SEP*(∅, GLAV, UCQ) and ST*CHAR*(∅, GLAV, UCQ).*

The remainder of this section is dedicated to the proof of the above theorem and is organized as follows. We first provide the lower bound proof in Section 7.1, and then we provide a matching upper bound in Section 7.2.

7.1. Proof of Theorem 8: Lower bound

To simplify the presentation of the lower bound proof, we first assume that ontologies may contain also ternary predicates in their alphabet. Then, by simply applying *reification* we will provide the proof by removing such an assumption. We start with STSEP(∅, GLAV, UCQ), and then we address the STCHAR(∅, GLAV, UCQ) case.

With ternary predicates: The proof of Π_3^p-hardness is by a LOGSPACE reduction from the *complement of the* ∃∀∃*-3CNF problem*. Given a formula ϕ of the form $\phi = \exists\vec{x}\forall\vec{y}\exists\vec{z}.c_1 \wedge \cdots \wedge c_k$, such that c_i is a disjunction of exactly three literals for $i \in [1, k]$, ∃∀∃-3CNF is the problem of deciding whether ϕ is satisfiable. It is the prototypical Σ_3^p-complete [62] problem.

The reduction is as follows. Let $\phi = \exists\vec{x}\forall\vec{y}\exists\vec{z}.c_1 \wedge \cdots \wedge c_k$ be an input formula for the ∃∀∃-3CNF problem, where $\vec{x} = (x_1, \ldots, x_n)$, $\vec{y} = (y_1, \ldots, y_m)$, and $\vec{z} = (z_1, \ldots, z_p)$. In what follows, for each $i \in [1, k]$, we denote by $v_{i,1}$, $v_{i,2}$, and $v_{i,3}$ the variable in $\vec{x} \cup \vec{y} \cup \vec{z}$ appearing, respectively, in the first, second, and third literal of the clause c_i. We build an OBDM system $\Sigma_\phi = \langle \mathcal{O}_\phi, \mathcal{S}_\phi, \mathcal{M}_\phi, D_\phi \rangle$ as follows:

- $\mathcal{S}_\phi = \{s_X, s_Y^1, \ldots, s_Y^m, s_a, s_b, s_1, \ldots, s_k\}$.
- D_ϕ is the $\mathcal{S}_\phi$-database consisting of the following facts:
  - $s_X(x_1, \ldots, x_n)$. So, we have a constant x_i for each for each variable $x_i \in \vec{x}$;
  - $s_Y^i(0)$ and $s_Y^i(1)$ for each $i \in [1, m]$. In this way, a formula of the form $\bigwedge_i^m s_Y^i(y_i)$ has 2^m possible assignments, each corresponding to an assignment to the variables $\vec{y}$ in ϕ;
  - $s_a(x_i, a)$ for each $i \in [1, n]$;
  - $s_b(0, b)$ and $s_b(1, b)$;
  - for each $i \in [1, k]$, we have 7 different facts of the form $s_i(w_1, w_2, w_3)$, with w_j being either 0 or 1 for $j \in [1, 3]$. These 7 facts represent all the possible assignments that satisfy the clause c_i of ϕ. For example, if ϕ contains the clause $c_3 = (\neg x_2 \vee \neg y_5 \vee z_2)$, then D_ϕ would contain the facts $s_3(0, 0, 0)$, $s_3(0, 0, 1)$, $s_3(0, 1, 0)$, $s_3(0, 1, 1)$, $s_3(1, 0, 0)$, $s_3(1, 0, 1)$, $s_3(1, 1, 1)$, and it would not contain the fact $s_3(1, 1, 0)$ corresponding to an assignment that does not satisfy the clause c_3.
- $\mathcal{O}_\phi = \emptyset$. The alphabet of $\mathcal{O}_\phi$ contains a ternary predicate P_i for each $i \in [1, k]$ and a binary predicates P_{ab}. Informally, P_{ab} will store $P_{ab}(0, b)$ and $P_{ab}(1, b)$, where 0, 1, and b are constants. Moreover, for each $i \in [1, n]$, P_{ab} will store $P_a(x_i, a)$, where a is a constant and x_i is the constant representing the variable x_i in ϕ. The role played by the ternary predicates P_i is twofold. (i) They represent the clauses of ϕ, with $\vec{x}$ as constants coming from D_ϕ, each variable $y \in \vec{y}$ replaced with either 0 or 1, and the variables $\vec{z}$ as existential variables generated by a GLAV assertion in $\mathcal{M}_\phi$; (ii) They also represent the 7 possible assignments that satisfy the clause c_i.
- Finally, the mapping $\mathcal{M}_\phi$ consists of the following GLAV assertions:
  - m_ϕ is the GLAV assertion $\{\vec{x} \cup \vec{y} \mid s_X(x_1, \ldots, x_n) \wedge \bigwedge_{i=1}^m s_Y^i(y_i)\} \rightarrow \{\vec{x} \cup \vec{y} \mid \exists z_1, \ldots, z_p . P_1(v_{1,1}, v_{1,2}, v_{1,3}) \wedge \cdots \wedge P_k(v_{k,1}, v_{k,2}, v_{k,3})\}$, where $\vec{x} \cup \vec{y} = (x_1, \ldots, x_n, y_1, \ldots, y_m)$. Recall that, for each $i \in [i, k]$, we have that $v_{i,1}$, $v_{i,2}$, and $v_{i,3}$ denote, respectively, the variable in $\vec{x} \cup \vec{y} \cup \vec{z}$ appearing in the first, second, and third literal of the clause c_i;
  - for $i \in [1, k]$, we have the assertion $m_i : \{(w_1, w_2, w_3) \mid s_i(w_1, w_2, w_3)\} \rightarrow \{(w_1, w_2, w_3) \mid P_i(w_1, w_2, w_3)\}$;
  - $m_a : \{(w_1, w_2) \mid s_a(w_1, w_2)\} \rightarrow \{(w_1, w_2) \mid P_{ab}(w_1, w_2)\}$;
  - $m_b : \{(w_1, w_2) \mid s_b(w_1, w_2)\} \rightarrow \{(w_1, w_2) \mid P_{ab}(w_1, w_2)\}$.

Intuitively, m_ϕ populates a total of 2^m formulae based on ϕ. These formulae are obtained by assigning all possible combinations of 0 and 1 constants to the variables $\vec{y} = (y_1, \ldots, y_m)$. It is important to notice that in each of these 2^m formulae the variables in $\vec{x}$ are always represented by the same constants in D_ϕ, while the variables in $\vec{z}$ are represented each time by fresh variables, generated at each application of the mapping assertion corresponding to a combination of 0 and 1 constants for the variables $\vec{y}$ (i.e., the set of variables generated by an application of the assertion m_ϕ is disjoint from the set of variables generated by another application of the assertion m_ϕ).

For each $i \in [1, k]$, the assertion m_i generates on the predicate P_i all the possible assignments that satisfy the clause c_i in the formula ϕ. Finally, the assertions m_a and m_b are used to generate in $\mathcal{M}_\phi(D_\phi)$ the facts $P_{ab}(x_i, a)$ for each $i = [1, n]$ and the two facts $P_{ab}(0, b)$ and $P_{ab}(1, b)$, respectively.

Finally, the fixed D_ϕ-datasets are $\lambda^+ = \{(a)\}$ and $\lambda^- = \{(b)\}$. Notice that λ^+ and λ^- are fixed, whereas the OBDM system Σ_ϕ can be constructed in LogSpace from an input formula $\phi = \exists\vec{x}\forall\vec{y}\exists\vec{z}.c_1 \wedge \cdots \wedge c_k$ for the $\exists\forall\exists$-3CNF problem.

We are now going to prove that $(\mathcal{C}_{\mathcal{O}_\phi}^{\mathcal{M}_\phi(D_\phi)}, (a)) \rightarrow (\mathcal{C}_{\mathcal{O}_\phi}^{\mathcal{M}_\phi(D_\phi)}, (b))$ if and only if ϕ is satisfiable. This proves the correctness of the reduction, and therefore that STSep(Ø, GLAV, UCQ) is Π_3^p-hard. Indeed, according to the semantic test given in Section 6, we have that there exists a proper Σ_ϕ-separation of λ^+ and λ^- in UCQ if and only if $(\mathcal{C}_{\mathcal{O}_\phi}^{\mathcal{M}_\phi(D_\phi)}, (a)) \nrightarrow (\mathcal{C}_{\mathcal{O}_\phi}^{\mathcal{M}_\phi(D_\phi)}, (b))$. In other words, $(\mathcal{C}_{\mathcal{O}_\phi}^{\mathcal{M}_\phi(D_\phi)}, (a)) \rightarrow (\mathcal{C}_{\mathcal{O}_\phi}^{\mathcal{M}_\phi(D_\phi)}, (b))$ is equivalent to say that there does not exist a proper Σ_ϕ-separation of λ^+ and λ^- in UCQ.

Claim 5. $(\mathcal{C}_{\mathcal{O}_\phi}^{\mathcal{M}_\phi(D_\phi)}, (a)) \rightarrow (\mathcal{C}_{\mathcal{O}_\phi}^{\mathcal{M}_\phi(D_\phi)}, (b))$ *if and only if ϕ is satisfiable.*

Proof. Since $\mathcal{O}_\phi = \emptyset$, we have that $\mathcal{C}_{\mathcal{O}_\phi}^{\mathcal{M}_\phi(D_\phi)}$ and $\mathcal{M}_\phi(D_\phi)$ coincide. Notice that, by definition, every function f witnessing that $(\mathcal{M}_\phi(D_\phi), (a)) \rightarrow (\mathcal{M}_\phi(D_\phi), (b))$ must be such that $f(a) = b$. As a consequence, due to the P_{ab}-facts occurring in $\mathcal{M}_\phi(D_\phi)$, the function f must associate to each constant x_i representing a variable of the formula ϕ either the constant 0 or the constant 1, i.e., for each $i \in [1, n]$ we have that either $f(x_i) = 0$ or $f(x_i) = 1$.

Now, if ϕ is satisfiable, then it is not hard to verify that there exists an association of the constants $\vec{x}$ with 0 and 1 that satisfy all 2^m formulae represented in $\mathcal{M}_\phi(D_\phi)$. In particular, for each of these formulae, f also needs to associate to the existential variables $\vec{z}$ generated by the application of m_ϕ corresponding to that formula the constants 0 and 1 that will satisfy all the clauses of that formula. Thus, if ϕ is satisfiable, then we have that $(\mathcal{M}_\phi(D_\phi), (a)) \rightarrow (\mathcal{M}_\phi(D_\phi), (b))$.

Conversely, if ϕ is not satisfiable, then it is not hard to verify that for every possible association made by a function f for the constants $\vec{x}$ there exist at least one of those 2^m formulae for which it is not possible to assign to the variables $\vec{z}$ of that formula the constants 0 and 1 that will satisfy all the clauses of that formula. Thus, if ϕ is not satisfiable, then we have that $(\mathcal{M}_\phi(D_\phi), (a)) \nrightarrow (\mathcal{M}_\phi(D_\phi), (b))$. □

As for the case of STCHAR(∅, GLAV, UCQ), it is possible to use exactly the same reduction provided above by discarding λ^- and considering only $\lambda^+ = \{(a)\}$. In particular, it is immediate to verify that $(\mathcal{M}_\phi(D_\phi), (a)) \nrightarrow (\mathcal{M}_\phi(D_\phi), (c))$ for each unary tuple (c) with c being a constant different from b (while whether it is the case that $(\mathcal{M}_\phi(D_\phi), (a)) \nrightarrow (\mathcal{M}_\phi(D_\phi), (b))$ depends solely on the satisfiability of ϕ).

We now discuss the necessary changes to be made to the above reduction to work in the presence of only binary predicates (atomic roles) in the ontology alphabet.

Without ternary predicates (only atomic roles): Let $\phi = \exists\vec{x}\forall\vec{y}\exists\vec{z}.c_1 \wedge \cdots \wedge c_k$ be an input formula for the ∃∀∃-3CNF problem, where $\vec{x} = (x_1, \ldots, x_n)$, $\vec{y} = (y_1, \ldots, y_m)$, and $\vec{z} = (z_1, \ldots, z_p)$. We build an OBDM system $\Sigma_\phi = \langle \mathcal{O}_\phi, \mathcal{S}_\phi, \mathcal{M}_\phi, D_\phi \rangle$ as follows:

- $\mathcal{S}_\phi = \{s_X, s_Y^1, \ldots, s_Y^m, s_a, s_b, s_{1,1}, s_{1,2}, s_{1,3}, s_{2,1}, s_{2,2}, s_{2,3}, \ldots, s_{k,1}, s_{k,2}, s_{k,3}\}$. Differently from the previous case, for each $i \in [1, k]$, we have three binary predicates $s_{i,1}$, $s_{i,2}$, and $s_{i,3}$ instead of a single ternary relation s_i.
- D_ϕ is the $\mathcal{S}_\phi$-database consisting of the following facts:
  - $s_X(x_1, \ldots, x_n)$;
  - $s_Y^i(0)$ and $s_Y^i(1)$ for each $i \in [1, m]$;
  - $s_b(0, b)$ and $s_b(1, b)$;
  - for each $i \in [1, k]$, we make use of 7 different constants $A_{i,1}, \ldots, A_{i,7}$. Moreover, for each $i \in [1, k]$ and $j \in [1, 7]$, we have three facts of the form $s_{i,1}(A_{i,j}, w_{i,j}^1)$, $s_{i,2}(A_{i,j}, w_{i,j}^2)$, and $s_{i,3}(A_{i,j}, w_{i,j}^3)$ with $w_{i,j}^l$ being either 0 or 1 for $l \in [1, 3]$. These 7 constants represent all the possible assignments that satisfy the clause c_i of ϕ, and the value assumed by each $w_{i,j}^l$ is the value corresponding to the j-th assignment in the l-th position of clause c_i. For example, if ϕ contains the clause $c_3 = (\neg x_2 \vee \neg y_5 \vee z_2)$, then D_ϕ would contain the following facts: $s_{3,1}(A_{3,1}, 0)$, $s_{3,2}(A_{3,1}, 0)$, and $s_{3,3}(A_{3,1}, 0)$; $s_{3,1}(A_{3,2}, 0)$, $s_{3,2}(A_{3,2}, 0)$, and $s_{3,3}(A_{3,2}, 1)$; $s_{3,1}(A_{3,3}, 0)$, $s_{3,2}(A_{3,3}, 1)$, and $s_{3,3}(A_{3,3}, 0)$; $s_{3,1}(A_{3,4}, 0)$, $s_{3,2}(A_{3,4}, 1)$, and $s_{3,3}(A_{3,4}, 1)$; $s_{3,1}(A_{3,5}, 1)$, $s_{3,2}(A_{3,5}, 0)$, and $s_{3,3}(A_{3,5}, 0)$; $s_{3,1}(A_{3,6}, 1)$, $s_{3,2}(A_{3,6}, 0)$, and $s_{3,3}(A_{3,6}, 1)$; $s_{3,1}(A_{3,7}, 1)$, $s_{3,2}(A_{3,7}, 1)$, and $s_{3,3}(A_{3,7}, 1)$. It would not contain the facts $s_{3,1}(A_{3,8}, 1)$, $s_{3,2}(A_{3,8}, 1)$, and $s_{3,3}(A_{3,8}, 0)$ corresponding to an assignment that does not satisfy the clause c_3.
- $\mathcal{O}_\phi = \emptyset$. The alphabet of $\mathcal{O}_\phi$ contains the atomic role P_{ab} and three atomic roles $P_{i,1}$, $P_{i,2}$, and $P_{i,3}$ for each $i \in [1, k]$.
- Finally, the mapping $\mathcal{M}_\phi$ consists of the following GLAV assertions:
  - m_ϕ is the GLAV assertion $\{\vec{x} \cup \vec{y} \mid s_X(x_1, \ldots, x_n) \wedge \bigwedge_{i=1}^m s_Y^i(y_i)\} \rightarrow \{\vec{x} \cup \vec{y} \mid \exists g_1, \ldots, g_k, z_1, \ldots, z_p . P_{1,1}(g_1, v_{1,1}) \wedge P_{1,2}(g_1, v_{1,2}) \wedge P_{1,3}(g_1, v_{1,3}) \wedge P_{2,1}(g_2, v_{2,1}) \wedge P_{2,2}(g_2, v_{2,2}) \wedge P_{2,3}(g_2, v_{2,3}) \wedge \cdots \wedge P_{k,1}(g_k, v_{k,1}) \wedge P_{k,2}(g_k, v_{k,2}) \wedge P_{k,3}(g_k, v_{k,3})\}$, where $\vec{x} \cup \vec{y} = (x_1, \ldots, x_n, y_1, \ldots, y_m)$. Recall that, for each $i \in [i, k]$, we have that $v_{i,1}$, $v_{i,2}$, and $v_{i,3}$ denote, respectively, the variable in $\vec{x} \cup \vec{y} \cup \vec{z}$ appearing in the first, second, and third literal of the clause c_i;

- for $i \in [1,k]$, we have the assertions $m_{i,1} : \{(w_1,w_2) \mid s_{i,1}(w_1,w_2)\} \rightarrow \{(w_1,w_2) \mid P_{i,1}(w_1,w_2)\}$, $m_{i,2} : \{(w_1,w_2) \mid s_{i,2}(w_1,w_2)\} \rightarrow \{(w_1,w_2) \mid P_{i,2}(w_1,w_2)\}$, and $m_{i,3} : \{(w_1,w_2) \mid s_{i,3}(w_1,w_2)\} \rightarrow \{(w_1,w_2) \mid P_{i,3}(w_1,w_2)\}$;
- $m_a : \{(w_1,w_2) \mid s_a(w_1,w_2)\} \rightarrow \{(w_1,w_2) \mid P_{ab}(w_1,w_2)\}$;
- $m_b : \{(w_1,w_2) \mid s_b(w_1,w_2)\} \rightarrow \{(w_1,w_2) \mid P_{ab}(w_1,w_2)\}$.

Finally, exactly as in the previous case of ternary predicates, the fixed D_ϕ-datasets are $\lambda^+ = \{(a)\}$ and $\lambda^- = \{(b)\}$. Notice that λ^+ and λ^- are fixed, whereas the OBDM system Σ_ϕ can be constructed in LogSpace from an input formula $\phi = \exists\vec{x}\forall\vec{y}\exists\vec{z}.c_1 \wedge \cdots \wedge c_k$ for the $\exists\forall\exists$-3CNF problem.

Intuitively, differently from the case of ternary predicates, each of the 2^m formulae generated by m_ϕ is represented by binary predicates $P_{1,1}, P_{1,2}, P_{1,3}, \ldots, P_{k,1}, P_{k,2}, P_{k,3}$, with the additional freshly introduced existential variables $g_1, \ldots, g_k$ representing the resulting clauses $c_i, \ldots, c_k$ of that formula. Thus, for each of the 2^m formulae, a function f witnessing that $(\mathcal{M}_\phi(D_\phi), (a)) \rightarrow (\mathcal{M}_\phi(D_\phi), (b))$ must associate to each variable g_i of that formula one of the 7 constants $A_{i,1}, \ldots, A_{i,7}$. Using analogous arguments as those given in the proof of Claim 5, it is immediate to verify that $(\mathcal{M}_\phi(D_\phi), (a)) \rightarrow (\mathcal{M}_\phi(D_\phi), (b))$ if and only if ϕ is satisfiable, from which we derive that STSep(∅, GLAV, UCQ) is Π_3^p-hard (observe that $\mathcal{M}_\phi(D_\phi) = \mathcal{C}_{\mathcal{O}_\phi}^{\mathcal{M}_\phi(D_\phi)}$). As in the previous case of ternary predicates, for the case of STChar(∅, GLAV, UCQ), it is possible to use exactly the same reduction provided above by discarding λ^- and considering only $\lambda^+ = \{(a)\}$. Since we always have that $(\mathcal{M}_\phi(D_\phi), (a)) \not\rightarrow (\mathcal{M}_\phi(D_\phi), (c))$ for each unary tuple (c) with c being a constant different from b (while whether it is the case that $(\mathcal{M}_\phi(D_\phi), (a)) \not\rightarrow (\mathcal{M}_\phi(D_\phi), (b))$ depends solely on the satisfiability of ϕ), we derive that STChar(∅, GLAV, UCQ) is Π_3^p-hard as well.

7.2. *Proof of Theorem 8: Upper bound*

First of all, we assume to only deal with consistent OBDM systems Σ. Indeed, given an inconsistent OBDM system $\Sigma = \langle J, D\rangle$ and two D-datasets λ^+ and λ^- (resp., a D-dataset λ^+) of arity n, we point out that there exists a proper Σ-separation (resp., Σ-characterization) of λ^+ and λ^- (resp., of λ^+) in UCQ if and only if $\lambda^+ = dom(D)^n$, where n is the arity of λ^+. Since we will provide a Π_3^p upper bound for the case of consistent OBDM systems and since for an OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D\rangle$ such that $\mathcal{O}$ is a *DL-Lite*$_\mathcal{R}$ ontology and $\mathcal{M}$ is a GLAV mapping checking whether Σ is inconsistent can be done in NP using the technique described in the proof of Theorem 1, for simplifying the presentation of the proof we can assume to deal only with consistent OBDM systems.

Recall from the semantic tests given at the end of Section 6 that there exists a proper Σ-separation (resp. Σ-characterization) of λ^+ and λ^- (resp., of λ^+) in UCQ if and only if it is the case that $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{a}) \not\rightarrow (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{b})$ for all $\vec{a} \in \lambda^+$ and all $\vec{b} \in \lambda^-$ (resp., all $\vec{b} \in dom(D)^n \setminus \lambda^+$). With this observation at hand, we can solve the *complements* of Sep(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ) and Char(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ) in Σ_3^p using the following intuition: given an OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D\rangle$ such that $\mathcal{O}$ is a *DL-Lite*$_\mathcal{R}$ ontology and $\mathcal{M}$ is a GLAV mapping, and given two D-datasets λ^+ and λ^- (resp., a D-dataset λ^+) of arity n, we first guess a tuple $\vec{a}$, a tuple $\vec{b}$, and a function $\mathcal{W}$ from $dom(D)$ to $\mathsf{Const} \cup \mathcal{V}$. Then, we check whether $\vec{a} \in \lambda^+$, $\vec{b} \in \lambda^-$ (resp., $\vec{b} \in dom(D)^n \setminus \lambda^+$), $\mathcal{W}$ is such that $\mathcal{W}(\vec{a}) = \vec{b}$ and every constant in $dom(D)$ is mapped to an existing term in $\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}$. Finally, we call an oracle for the problem of checking whether $\mathcal{W}$ can be extended to a function f witnessing that $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{a}) \rightarrow (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{b})$.

In order to realize the presented intuition, we first introduce some preliminary results that will allow us to simplify the presentation of the problem, as well as some naming conventions that we will adopt for the fresh variables generated when computing $\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}$ starting from D.

We start by observing that checking whether $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{a}) \rightarrow (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{b})$ is equivalent to checking whether $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{a}) \rightarrow ((\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)})', \vec{b}')$, where $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)})'$ is obtained from $\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}$ simply by substituting every occurrence of a constant $c \in dom(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)})$ with its fresh copy c'.

Lemma 1. *$(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{a}) \rightarrow (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{b})$ if and only if $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{a}) \rightarrow ((\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)})', \vec{b}')$.*

Proof. **"Only-if part":** Let h be a homomorphism witnessing $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{a}) \rightarrow (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}, \vec{b})$, i.e. it holds that: (i) $h(\vec{a}) = \vec{b}$ and (ii) $\alpha \in \mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}$ implies $h(\alpha) \in \mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}$ for each $\alpha \in \mathcal{C}_\mathcal{O}^{\mathcal{M}(D)}$. Let h' be a function such that, for every

term t (i.e., either a constant or a variable): (i) if $h(t) = c$ for a constant c, then $h'(t) = c'$, i.e. h' assigns to t the copy c' of c; (ii) if $h(t) = v$ for a variable v, then $h'(v) = v$, i.e. h' assigns to t the same variable v. Since h is such that $h(\vec{a}) = \vec{b}$, we have that $h'(\vec{a}) = \vec{b'}$. Moreover, since $\alpha \in \mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}$ implies $h(\alpha) \in \mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}$ for each $\alpha \in \mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}$, by construction we have that $\alpha \in \mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}$ implies $h'(\alpha) \in (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)})'$ for each $\alpha \in \mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}$. Thus, h' witnesses that $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{a}) \to ((\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)})', \vec{b'})$.

"If part": The proof can be obtained by following the same considerations of the "Only-if" part, by switching the roles of the two functions h and h'. □

As a consequence of the above lemma, we can show that it is possible to reduce the problem of verifying whether $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$ to the equivalent problem of verifying whether $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$, where D' is computed from D by substituting every occurrence of a constant $c \in dom(D)$ with its fresh copy $c' \in dom(D')$. Although strictly not necessary, in the problem of verifying whether $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$ (equivalent to $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$) this simple consideration will allow us to simplify the presentation of the upper bound proof, because we can distinguish between the canonical structure on the left-hand side of the arrow and the canonical structure on the right-hand side of the arrow.

Lemma 2. *$(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{b})$ if and only if $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$.*

Proof. This is a straightforward consequence of Lemma 1 and the fact that $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)})'$ and $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}$ coincide (up to variable renaming). □

In all what follows, given an $\mathcal{S}$-database D and a tuple $\vec{b}$ of constants from $dom(D)$, we will implicitly assume that D', called the *copy of* D, represents the $\mathcal{S}$-database obtained from D by substituting every occurrence of a constant $c \in dom(D)$ with its fresh copy $c' \in dom(D')$, and $\vec{b'}$ represents the copy of the tuple $\vec{b}$. It is clear that both D' and $\vec{b'}$ can be computed in polynomial time from D and $\vec{b}$.

The following fundamental lemma tells us that, when verifying whether $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$, on the left-hand side of the arrow we can actually restrict the attention only at $\mathcal{M}(D)$.

Lemma 3. *$(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$ if and only if $(\mathcal{M}(D), \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$.*

Proof. The "Only-if" part is trivial, since by restricting the homomorphism witnessing $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$ to the terms in $\mathcal{M}(D)$ we derive an homomorphism witnessing $(\mathcal{M}(D), \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$.

As for the "If" part, let h be a homomorphism witnessing $(\mathcal{M}(D), \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$. Consider any concept inclusion assertion $B_1 \sqsubseteq B_2$ (resp., role inclusion assertion $R_1 \sqsubseteq R_2$) in $\mathcal{O}$ such that $\mathcal{M}(D) \models B_1(t)$ for some $t \in dom(\mathcal{M}(D))$ (resp, $\mathcal{M}(D) \models R_1(t_1, t_2)$ for some terms $t_1, t_2 \in dom(\mathcal{M}(D))$). This means that $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)} \models B_2(t)$ (resp., $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)} \models R_2(t_1, t_2)$). Now, since by assumption we have that $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')} \models B_1(h(t))$ (resp, $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')} \models R_1(h(t_1), h(t_2))$) because h is a homomorphism from $\mathcal{M}(D)$ to $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}$, we derive that $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')} \models B_2(h(t))$ (resp., $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')} \models R_2(h(t_1), h(t_2))$). With this observation at hand, it is easy to see that any homomorphism h witnessing $(\mathcal{M}(D), \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$ can be extended to an homomorphism h' witnessing $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}, \vec{a}) \to (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$. □

We now introduce some naming conventions we will adopt for the fresh variables generated when computing $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D)}$ starting from D.

Naming convention for the chase of the database with respect to the mapping: Let $\mathcal{M}$ be a GLAV mapping relating a source schema $\mathcal{S}$ to an ontology $\mathcal{O}$, and let D an $\mathcal{S}$-database. We call $p = (m, h_m)$ a *single application* of $\mathcal{M}$ on D, where $m = \{\vec{x} \mid \exists \vec{y}.\phi_{\mathcal{S}}(\vec{x}, \vec{y})\} \to \{\vec{x} \mid \exists \vec{z}.\varphi_{\mathcal{O}}(\vec{x}, \vec{z})\}$, with $\vec{z} = (z_1, \ldots, z_n)$, is a mapping assertion in $\mathcal{M}$, and h_m is a homomorphism from $set(\phi)$ to D. We denote by $p(D)$ the set of atoms $h'(set(\varphi_{\mathcal{O}}))$, where h' extends h_m by assigning to the existential variable $z_i \in \vec{z}$ the variable $z_i^p \in dom(\mathcal{M}(D))$, for $i \in [1, n]$. Notice that, up to variable renaming, $\mathcal{M}(D)$ coincides with $\bigcup_{p \in \mathcal{P}} p(D)$, where $\mathcal{P}$ is the set of all the possible single applications of $\mathcal{M}$ on D.

It is important to notice that, in this way, each variable z_i^p generated when chasing D with respect to $\mathcal{M}$ carries in its name both the application $p = (m, h_m)$ that generated it and which was the variable z_i inside m.

Naming convention for the chase with respect to the ontology: Let $\mathcal{M}$ be a GLAV mapping relating a source schema $\mathcal{S}$ to an ontology $\mathcal{O}$, let D an $\mathcal{S}$-database, and $\mathcal{O}$ be a *DL-Lite*$_\mathcal{R}$ ontology. We follow standard conventions for the name given to the freshly introduce variables when computing the chase of $\mathcal{M}(D)$ with respect to $\mathcal{O}$ (see, e.g. [33]). More specifically, $dom(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)})$ contains all the constants and variables occurring in $dom(\mathcal{M}(D))$ and all the variables of the form $tR_1, \ldots, R_n$, with $t \in dom(\mathcal{M}(D))$, $n \geqslant 1$, and R_i a basic role for each $i \in [1, n]$, such that:

- there is a basic concept B with $\mathcal{M}(D) \models B(t)$ and $\mathcal{O} \models B \sqsubseteq \exists R_1$, but $R'(t, t') \notin \mathcal{M}(D)$ for all $t' \in dom(\mathcal{M}(D))$ and basic role R' with $\mathcal{O} \models R' \sqsubseteq R_1$;
- $\mathcal{O} \models \exists R_i^- \sqsubseteq \exists R_{i+1}$ and $\mathcal{O} \not\models R_i^- \sqsubseteq R_{i+1}$, for each $i \in [1, n-1]$.

As an example, suppose that $\mathcal{M}(D) = \{A(t)\}$ for a term t that is either a constant in D or a fresh variable introduced when chasing D with respect to $\mathcal{M}$, and let $\mathcal{O} = \{A \sqsubseteq C, C \sqsubseteq \exists R^-, \exists R \sqsubseteq \exists S\}$. Then, $\mathcal{C}_\mathcal{O}^{\mathcal{M}(D)} = \{A(t), C(t), R(tR^-, t), S(tR^-, tR^-S)\}$, where tR^- and tR^-S are the fresh variables introduced when chasing $\mathcal{M}(D)$ with respect to $\mathcal{O}$.

Before illustrating the non-deterministic algorithms for solving the complements of SEP(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ) and CHAR(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ), we introduce a new notion and some fundamental technical results. Let $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$ be an OBDM system such that $\mathcal{O}$ is a *DL-Lite*$_\mathcal{R}$ ontology and $\mathcal{M}$ is a GLAV mapping, D be an $\mathcal{S}$-database, $\vec{a}$ be a tuple of constants from $dom(D)$, D' be the copy of D, and $\vec{b'}$ be a tuple of constants from $dom(D')$. A *partial witnessing function* (in short, *pwf*) of $(\mathcal{M}(D), \vec{a})$ on $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}, \vec{b'})$ is a function $\mathcal{W}$ from $dom(D)$ to $dom(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')})$ such that $\mathcal{W}(\vec{a}) = \vec{b'}$. We call such function partial because only the constants of $\mathcal{M}(D)$ have an assignment to a term in $\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}$, while the variables of $\mathcal{M}(D)$ have not been assigned yet. The following lemma follows by definition.

Lemma 4. *$(\mathcal{M}(D), \vec{a}) \to (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}, \vec{b'})$ if and only if there exists a pwf $\mathcal{W}$ of $(\mathcal{M}(D), \vec{a})$ on $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}, \vec{b'})$ that can be extended to a function f from $dom(\mathcal{M}(D))$ to $dom(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')})$ witnessing that $(\mathcal{M}(D), \vec{a}) \to (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}, \vec{b'})$, i.e., such that $\alpha \in \mathcal{M}(D)$ implies $f(\alpha) \in \mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}$ for each $\alpha \in \mathcal{M}(D)$.*

We now present a fundamental technical result that will allow us to restrict the attention only to those pwfs $\mathcal{W}$ such that the range of the function $\mathcal{W}$ contains constants and variables whose names lengths are of polynomial size with respect to the size of the input OBDM system Σ.

Lemma 5. *Let $\mathcal{P}$ be the set of all the possible single applications of $\mathcal{M}$ on D, and let $\mathcal{W}$ be a pwf of $(\mathcal{M}(D), \vec{a})$ on $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}, \vec{b'})$. Then, we have that $\mathcal{W}$ can be extended to a function f from $dom(\mathcal{M}(D))$ to $dom(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')})$ witnessing that $(\mathcal{M}(D), \vec{a}) \to (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}, \vec{b'})$ if and only if the following holds:*

- *for every single application $p \in \mathcal{P}$ of $\mathcal{M}$ on D, there is a function f'_p from variables in $dom(p(D))$ to $dom(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')})$ such that $\alpha \in p(D)$ implies $f_p(\alpha) \in \mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}$, where $f_p = \mathcal{W} \cup f'_p$ denotes the function from $dom(p(D))$ to $dom(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')})$ such that $f_p(t) = \mathcal{W}(t)$ if t is a constant and $f_p(t) = f'_p(t)$ if t is a variable.*

Proof. The proof can be obtained by simply combining these two observations: (i) all the constants in $dom(\mathcal{M}(D))$ are already assigned to a term in $dom(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')})$ by the function $\mathcal{W}$; (ii) every single application $p \in \mathcal{P}$ of $\mathcal{M}$ on D produces new variables in $\mathcal{M}(D)$ without ever reusing the variables introduced in other applications of the mappings. In other words, the set of all variables produced by a single application p of $\mathcal{M}$ on D, and the set of all variables produced by all the other possible single applications $p' \in \mathcal{P} \setminus \{p\}$ of $\mathcal{M}$ on D are disjoint. Thus, a function f extending $\mathcal{W}$ and witnessing $(\mathcal{M}(D), \vec{a}) \to (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}, \vec{b'})$ can always be split into many f_ps functions as in the statement of the lemma, and, vice versa, if the functions f_ps as in the statement of the lemma exist, then they can be combined into an overall function f witnessing $(\mathcal{M}(D), \vec{a}) \to (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}, \vec{b'})$. □

Algorithm 3 CharExistence

Input: Consistent OBDM system $\Sigma = \langle J, D\rangle$ with $J = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}\rangle$; D-dataset λ^+ of arity n
Output: `true` or `false`

1: *Guess* tuples $\vec{a}$ and $\vec{b}$ of arity n and a function $\mathcal{W}$ from $dom(D)$ to $\mathsf{Const} \cup \mathcal{V}$ such that $\mathcal{W}(\vec{a}) = \vec{b}'$
2: **for** each term t in the range of $\mathcal{W}$ **do**
3: *Guess* a pair $p_t = (m, h_m)$ and a sequence ρ_t of ontology assertions in $\mathcal{O}$. Here, $m = \{\vec{x} \mid \exists \vec{y}.\phi_{\mathcal{S}}(\vec{x}, \vec{y})\} \rightarrow \{\vec{x} \mid \exists \vec{z}.\varphi_{\mathcal{O}}(\vec{x}, \vec{z})\}$ is a mapping assertion in $\mathcal{M}$ and h_m is a function from the variables in $\vec{x} \cup \vec{y}$ to $dom(D')$
4: **end for**
5: Check whether $\vec{a} \in \lambda^+$, $\vec{b} \notin \lambda^+$, and $\mathcal{W}$ is a pwf of $(\mathcal{M}(D), \vec{a})$ on $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b}')$ (if not, then return `false`)
6: Call an oracle for the decision problem *CheckTotalExtension* with input Σ and $\mathcal{W}$
7: **if** the oracle accepts **then**
8: **return** `true`
9: **else**
10: **return** `false`
11: **end if**

As announced before, the above lemma has a crucial consequence. In particular, let $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D\rangle$ be an OBDM system. Since the number of atoms in $p(D)$ generated by a single application p of $\mathcal{M}$ on D is at most polynomial with respect to the size of $\mathcal{M}$ (i.e., it is at most the number of atoms occurring in the longest right-hand side among all $m \in \mathcal{M}$), from the above lemma and due to the forest-shaped form of the canonical structure $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}$, it is not hard to see that the following holds: if there exists a pwf $\mathcal{W}$ of $(\mathcal{M}(D), \vec{a})$ on $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b}')$ that can be extended to a function f from $dom(\mathcal{M}(D))$ to $dom(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')})$ witnessing that $(\mathcal{M}(D), \vec{a}) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b}')$, then there exists another pwf $\mathcal{W}'$ that can be extended to a function f' from $dom(\mathcal{M}(D))$ to $dom(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')})$ witnessing that $(\mathcal{M}(D), \vec{a}) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b}')$ and such that each variable in the range of f' (and therefore also in the range of $\mathcal{W}'$) has a name length that is bounded by a polynomial in the size of $\mathcal{O}$ and $\mathcal{M}$, which implies that the variable can be generated after polynomially many chase steps of $\mathcal{M}(D')$ with respect to $\mathcal{O}$.

We are now finally ready to present the non-deterministic algorithms for solving the complements of SEP($DL\text{-}Lite_{\mathcal{R}}$, GLAV, UCQ) and CHAR($DL\text{-}Lite_{\mathcal{R}}$, GLAV, UCQ). We first concentrate on CHAR($DL\text{-}Lite_{\mathcal{R}}$, GLAV, UCQ) by providing the non-deterministic Algorithm 3 (CharExistence).

Notice that the Algorithm 3 (CharExistence) uses an oracle for the auxiliary decision problem *CheckTotalExtension*, which is the problem of deciding, given an OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D\rangle$ and a pwf $\mathcal{W}$ of $(\mathcal{M}(D), \vec{a})$ on $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b}')$, whether $\mathcal{W}$ can be actually extended to a function f from $dom(\mathcal{M}(D))$ to $dom(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')})$ witnessing that $(\mathcal{M}(D), \vec{a}) \rightarrow (\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b}')$.

Lemma 6. *Let $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D\rangle$ be a consistent OBDM system and λ^+ be a D-dataset. We have that CharExistence(Σ, λ^+) terminates and returns* `true` *if and only if there exists a proper Σ-characterization of λ^+ in UCQ.*

Proof. Termination of the algorithm is immediate, while its correctness can be obtained by simply combining the semantic test given at the end Section 6 with Lemmata 2, 3, and 4. □

As for the running time, we observe that, due to the discussion after Lemma 5, we can concentrate only on pwfs $\mathcal{W}$ whose range contains only constants and variables whose names lengths are of polynomial size with respect to the input OBDM system Σ, and therefore the overall $\mathcal{W}$ is of polynomial size with respect to Σ since the number of constants in D is polynomial. Furthermore, we can check in polynomial time whether $\mathcal{W}$ is a pwf of $(\mathcal{M}(D), \vec{a})$ on $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b}')$, i.e., whether all the terms in the range of $\mathcal{W}$ occur in $dom(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')})$ as follows. For each term t in the range of $\mathcal{W}$, we compute $p_t(D')$, where p_t is the guessed application of $\mathcal{M}$ on D' for the term t (of course, we first need to check whether h_m is a homomorphism from the atoms on the left-hand side of m to D'). Starting from the set $p_t(D')$ of atoms, we then iteratively apply the guessed ontology assertions in the sequence ρ_t for the term t

one after the other, each time by applying the considered ontology assertion in a single chase step over all the atoms obtained so far. Finally, we check whether the term t occurs in the overall set of atoms obtained in this way.

Thus, with respect to the size of its input, the overall running time of the non-deterministic Algorithm 3 (CharExistence) is polynomial with an oracle for the decision problem *CheckTotalExtension*. As a consequence, we derive that the complement of CHAR(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ) is in NP with an oracle in $\mathcal{C}$, i.e., NP$^\mathcal{C}$, where $\mathcal{C}$ is the computational complexity of *CheckTotalExtension*. As for the case of SEP(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ), we can use a slight adaptation of the Algorithm 3 (CharExistence), which takes in the input an additional D-dataset λ^- and modifies the step 5 of the algorithm by checking $\vec{b} \in \lambda^-$ instead of checking $\vec{b} \notin \lambda^+$. This means that the complement of SEP(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ) is in NP$^\mathcal{C}$ as well, where $\mathcal{C}$ is the computational complexity of *CheckTotalExtension*.

To conclude the proof of the upper bound, we are now going to show that *CheckTotalExtension* is in Π_2^p (thus, $\mathcal{C} = \Pi_2^p$), which gives us the desired Σ_3^p upper bound for the complements of both CHAR(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ) and SEP(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ), and therefore this will show that both CHAR(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ) and SEP(*DL-Lite*$_\mathcal{R}$, GLAV, UCQ) are in Π_3^p, as required.

Upper bound for the decision problem CheckTotalExtension

Lemma 7. *CheckTotalExtension is in* Π_2^p.

Proof. We recall that *CheckTotalExtension* is the problem of deciding, given an OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$ and a pwf $\mathcal{W}$ of $(\mathcal{M}(D), \vec{a})$ on $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}, \vec{b}')$, whether $\mathcal{W}$ can be extended to a function f from $dom(\mathcal{M}(D))$ to $dom(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')})$ witnessing that $(\mathcal{M}(D), \vec{a}) \rightarrow (\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}, \vec{b}')$, i.e., such that $\alpha \in \mathcal{M}(D)$ implies $f(\alpha) \in \mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}$ for each $\alpha \in \mathcal{M}(D)$.

By Lemma 5, we can immediately derive the following non-deterministic algorithm to solve the *complement* of the *CheckTotalExtension* decision problem:

- Guess a pair $p = (m, h_m)$, where $m = \{\vec{x} \mid \exists \vec{y}.\phi_\mathcal{S}(\vec{x}, \vec{y})\} \rightarrow \{\vec{x} \mid \exists \vec{z}.\varphi_\mathcal{O}(\vec{x}, \vec{z})\}$ is a mapping assertion in $\mathcal{M}$, and h_m is a function from the variables in $\vec{x} \cup \vec{y}$ to $dom(D)$;
- Check whether p is an application of m on D, which amounts to check whether $m \in \mathcal{M}$ and h_m is a homomorphism from $set(\phi_\mathcal{S})$ to D (if not, then return `false`);
- Compute $p(D)$;
- Call an oracle for the decision problem *CheckSingleExtension* with input Σ, $\mathcal{W}$, $p(D)$;
- If the oracle accepts, then return `false`; otherwise return `true`.

Notice that the above non-deterministic algorithm uses an oracle for the auxiliary decision problem *CheckSingleExtension*, which is the problem of deciding, given an OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$, a pwf $\mathcal{W}$ of $(\mathcal{M}(D), \vec{a})$ on $(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}, \vec{b}')$, and a set of atoms $p(D)$ from $\mathcal{M}(D)$, whether $\mathcal{W}$ can be actually extended to a function f from $dom(p(D))$ to $dom(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')})$ such that f is a homomorphism from $p(D)$ to $\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}$, i.e., $\alpha \in p(D)$ implies $f(\alpha) \in \mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}$ for each $\alpha \in p(D)$. Notice the difference between the decision problems *CheckTotalExtension* and *CheckSingleExtension*. While *CheckTotalExtension* additionally takes in the input only Σ and $\mathcal{W}$, *CheckSingleExtension* takes in the input also the set of atoms $p(D)$.

The termination of the algorithm is straightforward. Furthermore, due to Lemma 5 and the definition of the auxiliary decision problem *CheckSingleExtension*, it is immediate to verify that the above non-deterministic algorithm solves the complement of *CheckTotalExtension*, i.e., returns `true` if and only if $\mathcal{W}$ cannot be extended to a function f from $dom(\mathcal{M}(D))$ to $dom(\mathcal{C}_\mathcal{O}^{\mathcal{M}(D')})$ such that $\alpha \in \mathcal{M}(D)$ implies $f(\alpha) \in \mathcal{C}_\mathcal{O}^{\mathcal{M}(D')}$ for each $\alpha \in \mathcal{M}(D)$. As for its running time, we observe that the size of the guessed p is clearly polynomial with respect to the size of the input, and both checking whether h_m is a homomorphism from $set(\phi_\mathcal{S})$ to D, and computing the set $p(D)$ of atoms, given h_m, can be performed in polynomial time with respect to the size of the input. As a consequence, we derive that the complement of *CheckTotalExtension* is in NP with an oracle in $\mathcal{C}'$, i.e., NP$^{\mathcal{C}'}$, where $\mathcal{C}'$ is the computational complexity of *CheckSingleExtension*. Thus, *CheckTotalExtension* is in coNP$^{\mathcal{C}'}$. In the following lemma, we are going to show that *CheckSingleExtension* is in NP (thus, $\mathcal{C}' = \text{NP}$), which concludes the proof of the desired Π_2^p upper bound for the decision problem *CheckTotalExtension*. □

Upper bound for the decision problem CheckSingleExtension

Lemma 8. *CheckSingleExtension is in NP.*

Proof. Recall that *CheckSingleExtension* is the problem of deciding, given an OBDM system $\Sigma = \langle \mathcal{O}, \mathcal{S}, \mathcal{M}, D \rangle$, a pwf $\mathcal{W}$ of $(\mathcal{M}(D), \vec{a})$ on $(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}, \vec{b'})$, and a set $p(D)$ such that $p(D) \subseteq \mathcal{M}(D)$, whether $\mathcal{W}$ can be extended to an homomorphism f from $p(D)$ to $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}$. In other words, the problem seeks for a function f' that assign to each variable in $p(D)$ a term occurring in $dom(\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')})$ such that $\alpha \in p(D)$ implies $f(\alpha) \in \mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}$ for each $\alpha \in p(D)$, where $f = \mathcal{W} \cup f'$ is the function such that $f(t) = \mathcal{W}(t)$ is t is a constant, and $f(t) = f'(t)$ if t is a variable.

Obviously, for an homomorphism f from $p(D)$ to $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}$ to exist, we do not need to generate all the atoms in $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}$ but we actually need only a number of atoms that is less than or equal the number of atoms in $p(D)$. Furthermore, due to the forest-shaped form of the canonical structure $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}$, each of such atoms can be generated after polynomially many chase steps of $\mathcal{M}(D')$ with respect to $\mathcal{O}$.

From the above consideration, we can immediately derive the following non-deterministic algorithm to solve the *CheckSingleExtension* decision problem.

- $A' := \emptyset$;
- For each $\alpha \in p(D)$:
  - *Guess* a pair $p'_\alpha = (m, h_m)$ and a sequence ρ_α of ontology assertions in $\mathcal{O}$. Here, $m = \{\vec{x} \mid \exists \vec{y}.\phi_{\mathcal{S}}(\vec{x}, \vec{y})\} \rightarrow \{\vec{x} \mid \exists \vec{z}.\varphi_{\mathcal{O}}(\vec{x}, \vec{z})\}$ is a mapping assertion in $\mathcal{M}$ and h_m is a function from the variables in $\vec{x} \cup \vec{y}$ to $dom(D')$;
  - Check whether h_m is a homomorphism from $set(\phi_{\mathcal{S}})$ to D' (if not, then return `false`);
  - Compute $p'_\alpha(D')$ and set $A' := A' \cup p'_\alpha(D')$;
  - Let $\rho_\alpha = (o_1, \ldots, o_n)$;
  - For each $i \leftarrow 1, \ldots, n$:
    - Let $\mathcal{N}$ be the set of all the fresh atoms obtained by applying the ontology assertion o_i to all the atoms in A' in a single chase step;
    - Set $A' := A' \cup \mathcal{N}$;
  - End For
- End For
- *Guess* a function f' from the variables in $dom(p(D))$ to $dom(A')$;
- Let $f = \mathcal{W} \cup f'$ be the function from $dom(p(D))$ to $dom(A')$ such that $f(t) = \mathcal{W}(t)$ if t is a constant and $f(t) = f'(t)$ if t is a variable;
- If f consists in a homomorphism from $p(D)$ to A', then return `true`; otherwise return `false`.

In a nutshell, for each atom $\alpha \in p(D)$, the algorithm generates a set A'_α of atoms in $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}$ by first computing the guessed application p'_α of $\mathcal{M}$ on D', and then by iteratively applying the guessed ontology assertions in the sequence ρ_α one after the other, each time by applying the considered ontology assertion in a single chase step over all the atoms obtained so far. Finally, the algorithm checks whether the given $\mathcal{W}$ can be extended to a homomorphism from $p(D)$ to A', where $A' = \bigcup_{\alpha \in p(D)} A'_\alpha$ is the set of all the atoms obtained as above.

The termination and the correctness of the algorithm are straightforward. It is indeed immediate to verify that the above non-deterministic algorithm solves *CheckSingleExtension*, i.e., returns `true` if and only if $\mathcal{W}$ can be extended to an homomorphism f from $p(D)$ to $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}$.

As for its running time, as already observed, we can restrict the attention only to those atoms of $\mathcal{C}_{\mathcal{O}}^{\mathcal{M}(D')}$ that can be generated after polynomially many chase steps of $\mathcal{M}(D')$ with respect to $\mathcal{O}$. Thus, for each $\alpha \in p(D)$, both the guessed p'_α and the guessed sequence ρ_α are of polynomial size with respect to the input, which implies that the overall number of atoms in A' is of polynomial size with respect to the input. Furthermore, checking whether h_m is a homomorphism from $set(\phi_{\mathcal{S}})$ to D', computing the set $p'_\alpha(D')$ of atoms, applying in sequence the ontology assertions in ρ_α in single chase steps, and checking whether f is a homomorphism from $p(D)$ to A' (i.e., whether

$\alpha \in p(D)$ implies $f(\alpha) \in A'$ for each $\alpha \in p(D)$) are all steps that can be carried out in polynomial time with respect to the size of the input. This means that the above non-deterministic algorithm runs in polynomial time with respect to the size of the input, and therefore that *CheckSingleExtension* is in NP, as required. □

8. Conclusion and future work

In this paper, we have studied logical separability in OBDM. As a first contribution, we have illustrated a general framework for separability in OBDM, by also proposing natural relaxations of the classical separability notion (called here proper separation). In the general envision of separability as a tool for providing global post-hoc explanations of black-box models, we argue that such relaxations (especially the best approximations) are crucial in order to always be able to provide meaningful explanations even when proper separations do not exist. As a second contribution, by instantiating the general framework with the most common languages used in OBDM, we have provided a comprehensive study of three computational problems associated with the framework, namely Verification, Computation, and Existence. For the decision problems related to Verification and Existence, we have provided tight complexity results, whereas for the Computation problem we have devised two algorithms for computing the two forms of best approximated separations considered in this paper, thus proving they always exist.

We conclude the paper by discussing some interesting avenues for future work that deserve more investigation.

More expressive scenarios It would be interesting to study extensions of the scenario considered in this paper for the computational problems. For example, one may consider more expressive target query languages that go beyond UCQ, in order to capture proper separations in more cases. Natural candidates for this are UCQ$^{\neq}$, i.e., UCQ with *inequalities*, First-order logic, and *EQL-Lite* [16]. However, differently from our case, by extending the considered scenario with one of these query languages, adopting or not the UNA makes a difference.

As other interesting extensions of the considered scenario, one may investigate highly expressive ontology languages, such as $\mathcal{SHIQ}$ [36] or even $\mathcal{SROIQ}$ [35], where the separability task has not yet been studied even in the ontology-enriched query answering setting (i.e., without considering mapping assertions).

Strong separability In [37], separability comes into two flavours: *weak separability* and *strong separability*. In weak separability, which is the one we addressed in this paper, the requirement on the negative examples is that none of them is included in the set of certain answers of the separating query. On the other hand, in strong separability, the requirement on the negative examples is that all of them are included in the set of the certain answers of the *negation of the separating query*. Thus, a natural problem to be addressed in our considered OBDM scenario is strong separability, by considering relaxations also in this case.

Quantitative metrics for best approximations In this work, we have defined best approximations of the proper separation notion by adopting the set-inclusion metric. Another common choice that would be interesting to investigate is the cardinality criteria. Moreover, for a more fine-grained usage of separability as a tool for explanation, it would be natural to assign *weights* to both positive and negative examples, and to consider such weights when computing best approximations of proper separations.

Restricted signature A possible constraint that can be imposed when finding a separating query is that the query expression uses only a specific subset of the predicates available in the alphabet of the ontology $\mathcal{O}$. This may be important to meet some users' requirements regarding the separating query, as for example a user may ask for a separating query that does not make use of a specific concept. The separating task under restricted signature has been studied in [39] in the ontology-enriched query answering setting, in the particular case of expressive ontology languages, such as $\mathcal{ALC}$ or $\mathcal{ALCO}$. Typically, the computational complexity of checking for a proper separation increases under the restricted signature constraint. Thus, another notable direction is to study separability in the considered OBDM scenario under the restricted signature constraint.

Intelligible queries A delicate question from an end-user perspective is the number of atoms involved in each of the disjuncts of the separating queries. While under both GAV and LAV mappings our algorithms presented for the computation problem ensure that this number of atoms is "only" polynomially related to the size of the mapping and

the database, the same is not true in the presence of GLAV mappings. In particular, it remains an interesting open question whether or not, under GLAV mappings, there are cases in which separations (or their best approximated versions) must necessarily be of exponential size with respect to the size of the mapping and the database.

More generally speaking, the separating queries may sometimes be very hard to understand from an end-user perspective. Thus, it would be natural to consider also the length of each disjunct, as well as the number of disjuncts, as additional parameters when considering approximations. This requires to consider novel definitions and techniques that may allow obtaining, from end users' perspectives, more intelligible (approximated) separating queries.

Implementation Finally, we mention that we are currently implementing the algorithms and techniques proposed in this paper using the OBDM engine Mastro [14]. The implementation we are working on follows the recently introduced *Human-in-the-loop Artificial Intelligence* (HitAI) paradigm [69]. By running some experiments with a first prototype in real-world settings, we observed as the above-mentioned issues for future work turn out to be crucial, especially the ones concerning intelligible queries and quantitative metrics.

Acknowledgements

This work has been supported by MUR under the PRIN 2017 project HOPE (prot. 2017MMJJRE) and under the PNRR project FAIR (PE0000013), 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). Gianluca Cima has been fully supported by MUR under the PNRR project FAIR (PE0000013).

References

[1] S. Abiteboul, R. Hull and V. Vianu, *Foundations of Databases*, Addison Wesley Publ. Co., 1995.
[2] D. Angluin, Queries and concept learning, *Machine Learning* **2**(4) (1987), 319–342.
[3] M. Arenas and G.I. Diaz, The exact complexity of the first-order logic definability problem, *ACM Trans. Database Syst.* **41**(2) (2016), 13. doi:10.1145/2886095.
[4] M. Arenas, G.I. Diaz and E.V. Kostylev, Reverse engineering SPARQL queries, in: *Proceedings of the 25th International Conference on World Wide Web*, 2016.
[5] A. Artale, D. Calvanese, R. Kontchakov and M. Zakharyaschev, The DL-lite family and relations, *Journal of Artificial Intelligence Research* **36** (2009), 1–69. doi:10.1613/jair.2820.
[6] F. Baader, D. Calvanese, D. McGuinness, D. Nardi and P.F. Patel-Schneider (eds), *The Description Logic Handbook: Theory, Implementation and Applications*, Cambridge University Press, 2003.
[7] P. Barceló and M. Romero, The complexity of reverse engineering problems for conjunctive queries, in: *20th International Conference on Database Theory (ICDT 2017)*, M. Benedikt and G. Orsi, eds, Leibniz International Proceedings in Informatics (LIPIcs), Vol. 68, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2017, pp. 7:1–7:17, ISSN 1868-8969, http://drops.dagstuhl.de/opus/volltexte/2017/7052. ISBN 978-3-95977-024-8. doi:10.4230/LIPIcs.ICDT.2017.7.
[8] A. Bondy and M.R. Murty, *Graph Theory*, Graduate Texts in Mathematics, Springer, 2008.
[9] E. Botoeva, R. Kontchakov, V. Ryzhikov, F. Wolter and M. Zakharyaschev, Games for query inseparability of description logic knowledge bases, *Artificial Intelligence* **234** (2016), 78–119. doi:10.1016/j.artint.2016.01.010.
[10] E. Botoeva, C. Lutz, V. Ryzhikov, F. Wolter and M. Zakharyaschev, Query inseparability for ALC ontologies, *Artif. Intell.* **272**(C) (2019), 1–51. doi:10.1016/j.artint.2018.09.003.
[11] L. Bü, J. Lehmann, P. Westphal and S. Bin, DL-learner structured machine learning on Semantic Web data, in: *Companion Proceedings of the The Web Conference 2018, WWW '18, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva*, Switzerland, 2018, pp. 467–471. ISBN 978-1-4503-5640-4. doi:10.1145/3184558.3186235.
[12] S.R. Buss and L. Hay, On truth-table reducibility to SAT, *Information and Computation* **91**(1) (1991), 86–102. doi:10.1016/0890-5401(91)90075-D.
[13] A. Calì, G. Gottlob and M. Kifer, Taming the infinite chase: Query answering under expressive relational constraints, *Journal of Artificial Intelligence Research* **48** (2013), 115–174. doi:10.1613/jair.3873.
[14] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodriguez-Muro, R. Rosati, M. Ruzzi and D.F. Savo, The mastro system for ontology-based data access, *Semantic Web Journal* **2**(1) (2011), 43–53. doi:10.3233/SW-2011-0029.
[15] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini and R. Rosati, Tractable reasoning and efficient query answering in description logics: The DL-lite family, *Journal of Automated Reasoning* **39**(3) (2007), 385–429. doi:10.1007/s10817-007-9078-x.

[16] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini and R. Rosati, EQL-Lite: Effective first-order query processing in description logics, in: *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI 2007)*, 2007, pp. 274–279.
[17] D. Calvanese, G. De Giacomo, M. Lenzerini and M.Y. Vardi, Query processing under GLAV mappings for relational and graph databases, *Proceedings of the Very Large Database Endowment* **6**(2) (2012), 61–72.
[18] G. Cima, Preliminary results on ontology-based open data publishing, in: *Proceedings of the Thirtieth International Workshop on Description Logics (DL 2017), CEUR Electronic Workshop Proceedings*, Vol. 1879, 2017. http://ceur-ws.org/.
[19] G. Cima, *Abstraction in Ontology-Based Data Management, Frontiers in Artificial Intelligence and Applications*, Vol. 348, IOS Press, 2022.
[20] G. Cima, M. Console, M. Lenzerini and A. Poggi, Abstraction in data integration, in: *Proceedings of the Thirty-Sixth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2021)*, IEEE, 2021, pp. 1–11.
[21] G. Cima, M. Console, M. Lenzerini and A. Poggi, Monotone abstractions in ontology-based data management, in: *Proceedings of the Thirty-Sixth AAAI Conference on Artificial Intelligence (AAAI 2022)*, 2022, pp. 5556–5563.
[22] G. Cima, F. Croce and M. Lenzerini, Query definability and its approximations in ontology-based data management, in: *CIKM'21: The 30th ACM International Conference on Information and Knowledge Management*, ACM, 2021.
[23] G. Cima, M. Lenzerini and A. Poggi, Semantic characterization of data services through ontologies, in: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI 2019)*, 2019, pp. 1647–1653.
[24] G. Cima, M. Lenzerini and A. Poggi, Non-monotonic ontology-based abstractions of data services, in: *Proceedings of the Seventeenth International Conference on Principles of Knowledge Representation and Reasoning (KR 2020)*, 2020, pp. 243–252. doi:10.24963/kr.2020/25.
[25] G. Ciravegna, F. Giannini, M. Gori, M. Maggini and S. Melacci, Human-driven FOL explanations of deep learning, in: *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, 2020.
[26] R. Confalonieri, T. Weyde, T.R. Besold and F.M. del Prado Martín, Using ontologies to enhance human understandability of global post-hoc explanations of black-box models, *Artif. Intell.* (2021).
[27] A. Doan, A.Y. Halevy and Z.G. Ives, *Principles of Data Integration*, Morgan Kaufmann, 2012. ISBN 978-0-12-416044-6.
[28] R. Fagin, P.G. Kolaitis, R.J. Miller and L. Popa, Data exchange: Semantics and query answering, *Theoretical Computer Science* **336**(1) (2005), 89–124. doi:10.1016/j.tcs.2004.10.033.
[29] M. Friedman, A. Levy and T. Millstein, Navigational plans for data integration, in: *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI 1999)*, AAAI Press, 1999, pp. 67–73.
[30] M. Funk, J.C. Jung and C. Lutz, Frontiers and exact learning of ELI queries under DL-lite ontologies, in: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI 2022)*, 2022, pp. 2627–2633.
[31] M. Funk, J.C. Jung, C. Lutz, H. Pulcini and F. Wolter, Learning description logic concepts: When can positive and negative examples be separated? in: *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, International Joint Conferences on Artificial Intelligence Organization*, 2019, pp. 1682–1688. doi:10.24963/ijcai.2019/233.
[32] M.R. Garey, D.S. Johnson and L.J. Stockmeyer, Some simplified NP-complete graph problems, *Theoretical Computer Science* **1**(3) (1976), 237–267. doi:10.1016/0304-3975(76)90059-1.
[33] V. Gutié, Y.A. Ibá, R. Kontchakov and E.V. Kostylev, Queries with negation and inequalities over lightweight ontologies, *Journal of Web Semantics* **35** (2015), 184–202. doi:10.1016/j.websem.2015.06.002.
[34] V. Gutié, J.C. Jung and L. Sabellek, 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*, 2018, pp. 1847–1853. doi:10.24963/ijcai.2018/255.
[35] I. Horrocks, O. Kutz and U. Sattler, The even more irresistible SROIQ, in: *Proceedings of the Tenth International Conference on the Principles of Knowledge Representation and Reasoning (KR 2006)*, 2006, pp. 57–67.
[36] I. Horrocks, U. Sattler and S. Tobies, Reasoning with individuals for the description logic SHIQ, in: *Proceedings of the Seventeenth International Conference on Automated Deduction (CADE 2000)*, D. McAllester, ed., Lecture Notes in Computer Science, Vol. 1831, Springer, 2000, pp. 482–496.
[37] J.C. Jung, C. Lutz, H. Pulcini and F. Wolter, Logical separability of incomplete data under ontologies, in: *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning, KR 2020*, Rhodes, Greece, 2020.
[38] J.C. Jung, C. Lutz, H. Pulcini and F. Wolter, Separating data examples by description logic concepts with restricted signatures, in: *Proceedings of the Eighteenth International Conference on Principles of Knowledge Representation and Reasoning, International Joint Conferences on Artificial Intelligence Organization*, 2021.
[39] J.C. Jung, C. Lutz, H. Pulcini and F. Wolter, Separating data examples by description logic concepts with restricted signatures, in: *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning, KR 2021*, Online Event, November 3–12, 2021, 2021, pp. 390–399. doi:10.24963/kr.2021/37.
[40] J.C. Jung, C. Lutz, H. Pulcini and F. Wolter, Logical separability of labeled data examples under ontologies, *Artificial Intelligence* **313** (2022), 103785. doi:10.1016/j.artint.2022.103785.
[41] D.V. Kalashnikov, L.V.S. Lakshmanan and D. Srivastava, FastQRE: Fast query reverse engineering, in: *Proceedings of the 2018 International Conference on Management of Data*, Association for Computing Machinery, 2018.
[42] B. Konev, A. Ozaki and F. Wolter, A model for learning description logic ontologies based on exact learning, in: *AAAI*, 2016.
[43] M. Law, A. Russo and K. Broda, Inductive learning of answer set programs, in: *Logics in Artificial Intelligence*, 2014.
[44] J. Lehmann and P. Hitzler, Concept learning in description logics using refinement operators, *Mach. Learn.* (2010).

[45] M. Lenzerini, Data integration: A theoretical perspective, in: *Proceedings of the Twentyfirst ACM SIGACT SIGMOD SIGART Symposium on Principles of Database Systems (PODS 2002)*, 2002, pp. 233–246.
[46] M. Lenzerini, Ontology-based data management, in: *Proceedings of the 20th ACM Conference on Information and*, Knowledge Management, CIKM, 2011.
[47] M. Lenzerini, Ontology-based data management, in: *Proceedings of the Twentieth International Conference on Information and Knowledge Management (CIKM 2011)*, 2011, pp. 5–6. doi:10.1145/2063576.2063582.
[48] M. Lenzerini, Managing data through the lens of an ontology, *AI Magazine* **39**(2) (2018), 65–74. doi:10.1609/aimag.v39i2.2802.
[49] J. Liartis, E. Dervakos, O.M. Mastromichalakis, A. Chortaras and G. Stamou, Semantic queries explaining opaque machine learning classifiers, in: *Proceedings of the Workshop on Data Meets Applied Ontologies in Explainable AI*, 2021.
[50] C. Lutz, J. Marti and L. Sabellek, Query expressibility and verification in ontology-based data access, in: *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference (KR 2018)*, 2018, pp. 389–398.
[51] D.M.L. Martins, Reverse engineering database queries from examples: State-of-the-art, challenges, and research opportunities, *Information Systems* **83** (2019), 89–100. doi:10.1016/j.is.2019.03.002.
[52] D.M.L. Martins, Reverse engineering database queries from examples: State-of-the-art, challenges, and research opportunities, *Information Systems* **83** (2019), 89–100. doi:10.1016/j.is.2019.03.002.
[53] B. Motik, B. Cuenca Grau, I. Horrocks, Z. Wu, A. Fokoue and C. Lutz, *OWL 2 Web Ontology Language Profiles (Second Edition), W3C Recommendation, World Wide Web Consortium*, 2012, available at http://www.w3.org/TR/owl2-profiles/.
[54] D. Mottin, M. Lissandrini, Y. Velegrakis and T. Palpanas, New trends on exploratory methods for data analytics, *Proc. VLDB Endow.* **10**(12) (2017), 1977–1980. doi:10.14778/3137765.3137824.
[55] M. Ortiz, 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, 2019, pp. 1–14.
[56] C.H. Papadimitriou and M. Yannakakis, The complexity of facets (and some facets of complexity), *Journal of Computer and System Sciences* **28**(2) (1984), 244–259. doi:10.1016/0022-0000(84)90068-0.
[57] C. Persia, A. Ozaki and A. Mazzullo, Learning query inseparable ELH ontologies, in: *AAAI*, 2019.
[58] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini and R. Rosati, Linking data to ontologies, *Journal on Data Semantics* **X** (2008), 133–173.
[59] J. Rothe, Exact complexity of exact-four-colorability, *Information Processing Letters* **87**(1) (2003), 7–12. doi:10.1016/S0020-0190(03)00229-1.
[60] M.K. Sarker, N. Xie, D. Doran, M. Raymer and P. Hitzler, Explaining trained neural networks with Semantic Web technologies: First steps, in: *Proceedings of the Twelfth International Workshop on Neural-Symbolic Learning and Reasoning*, 2017.
[61] L.J. Stockmeyer, The polynomial-time hierarchy, *Theoretical Computer Science* **3**(1) (1976), 1–22. doi:10.1016/0304-3975(76)90061-X.
[62] L.J. Stockmeyer, The polynomial-time hierarchy, *Theoretical Computer Science* **3**(1) (1976), 1–22. doi:10.1016/0304-3975(76)90061-X.
[63] B. ten Cate and V. Dalmau, The product homomorphism problem and applications, in: *Proceedings of the Eigthteenth International Conference on Database Theory (ICDT 2015), LIPIcs*, Vol. 31, 2015, pp. 161–176.
[64] B. ten Cate and V. Dalmau, Conjunctive queries: Unique characterizations and exact learnability, in: *24th International Conference on Database Theory (ICDT 2021)*, Vol. 186, 2021.
[65] Q.T. Tran, C.-Y. Chan and S. Parthasarathy, *Query Reverse Engineering, The VLDB Journal* **23**(5) (2014).
[66] K.W. Wagner, More complicated questions about maxima and minima, and some closures of NP, *Theoretical Computer Science* **51** (1987), 53–80. doi:10.1016/0304-3975(87)90049-1.
[67] K.W. Wagner, Bounded query classes, *SIAM Journal on Computing* **19**(5) (1990), 833–846. doi:10.1137/0219058.
[68] Y.Y. Weiss and S. Cohen, Reverse engineering SPJ-queries from examples, in: *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, 2017.
[69] F.M. Zanzotto, Viewpoint: Human-in-the-loop artificial intelligence, *Journal of Artificial Intelligence Research* **64** (2019), 243–252. doi:10.1613/jair.1.11345.
[70] M.M. Zloof, Query-by-example: The invocation and definition of tables and forms, in: *Proceedings of the 1st International Conference on Very Large Data Bases, VLDB '75*, ACM, New York, NY, USA, 1975, pp. 1–24. ISBN 978-1-4503-3920-9. doi:10.1145/1282480.1282482.