

Journal Pre-proof

Metamodeling and metaquerying in OWL 2 QL

Maurizio Lenzerini, Lorenzo Lepore and Antonella Poggi

PII: S0004-3702(18)30187-5

DOI: <https://doi.org/10.1016/j.artint.2020.103432>

Reference: ARTINT 103432

To appear in: *Artificial Intelligence*

Received date: 24 April 2018

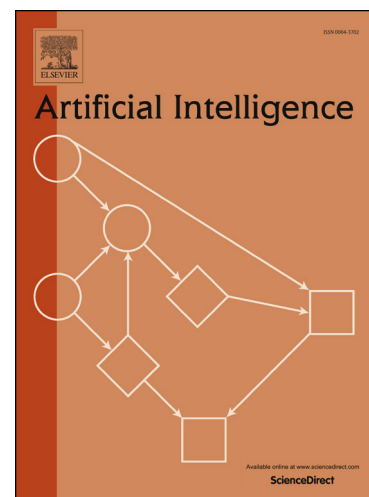
Revised date: 29 September 2020

Accepted date: 23 November 2020

Please cite this article as: M. Lenzerini, L. Lepore and A. Poggi, Metamodeling and metaquerying in OWL 2 QL, *Artificial Intelligence*, 103432, doi: <https://doi.org/10.1016/j.artint.2020.103432>.

This is a PDF file of an article that has undergone enhancements after acceptance, such as the addition of a cover page and metadata, and formatting for readability, but it is not yet the definitive version of record. This version will undergo additional copyediting, typesetting and review before it is published in its final form, but we are providing this version to give early visibility of the article. Please note that, during the production process, errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

© 2021 Published by Elsevier.



Metamodeling and metaquerying in OWL 2 QL

Maurizio Lenzerini, Lorenzo Lepore, Antonella Poggi

Abstract

OWL 2 QL is a standard profile of the OWL 2 ontology language, specifically tailored to Ontology-Based Data Management. Inspired by recent work on higher-order Description Logics, in this paper we present a new semantics for OWL 2 QL ontologies, called Metamodeling Semantics (MS), and show that, in contrast to the official Direct Semantics (DS) for OWL 2, it allows exploiting the metamodeling capabilities natively offered by the OWL 2 punning. We then extend unions of conjunctive queries with both metavariables, and the possibility of using TBox atoms, with the purpose of expressing meaningful metalevel queries. We first show that under MS both satisfiability checking and answering queries including only ABox atoms, have the same complexity as under DS. Second, we investigate the problem of answering general metaqueries, and single out a new source of complexity coming from the combined presence of a specific type of incompleteness in the ontology, and of TBox axioms among the query atoms. Then we focus on a specific class of ontologies, called TBox-complete, where there is no incompleteness in the TBox axioms, and show that general metaquery answering in this case has again the same complexity as under DS. Finally, we move to general ontologies and show that answering general metaqueries is coNP-complete with respect to ontology complexity, Π_2^P -complete with respect to combined complexity, and remains AC^0 with respect to ABox complexity.

1. Introduction

Representing knowledge in terms of an ontology expressed in a Description Logic [1], and then querying such representation to extract useful information about the modeled domain, has received a great deal of attention in recent years [2]. In particular, many research works have concentrated on designing optimal ontology languages with respect to the trade-off between expressive power and computational complexity of reasoning tasks, such as computing the answers to queries. Here, we focus on the OWL 2 QL ontology language, which is the most popular ontology language in Ontology-Based Data Management (OBDM) [3]. OBDM is a paradigm where ontologies are used both for querying the data of a pre-existing information system, and for carrying out data governance tasks, such as data profiling, or data quality assessment. The OWL 2 QL language [4], that is based on the Description Logic (DL) $DL-Lite_R$ [5], guarantees that the main reasoning tasks are tractable with respect to the size of the data, that is typically huge in real world applications.

Although the quest for tractability of algorithms for query answering and other reasoning tasks is extremely important for OBDM, there are other issues that are equally important, and have not yet been explored in depth. The one that we deal with in this paper is metamodeling and metaquerying [6]. Metamodeling is a form of higher-order modeling allowing for the definition of metaclasses and metaproperties in the ontology, where a metaclass is a class whose instances can be in turn classes, and a metaproperty is a property (or relation) whose instances are properties holding between classes, rather than individuals. Since metaclasses can be themselves instances of other metaclasses, *multi-level modeling* is another term used for metamodeling [7]. Metaquerying refers to the possibility of going beyond first order-logic in specifying queries. In particular, a metaquery not only can mention both metaclasses and metaproperties, but may also contain variables that can be bound to classes and properties, contrarily to the usual queries studied in OBDM, where variables only denote individuals in the domain.

While metamodeling has been studied in several fields, including Software Engineering, e.g., for conceptual modeling [8], for model-driven development [9] and for engineering design optimization [10], Relational Databases [11], and Artificial Intelligence (AI), e.g., for simulation modeling [12] and for knowledge representation [13], both metamodeling and metaquerying have not been investigated in detail in the context of ontology representation and reasoning, neither from the semantical nor from the computational point of view.

In this paper, we begin to fill this gap by investigating metamodeling and metaquerying in OWL 2 QL. Notice that OWL 2, and therefore OWL 2 QL, natively provides syntactic support for metamodeling through the so-called OWL 2 *punning*¹. Indeed, punning allows one to use the same name to denote ontology elements of different categories, such as a class and an individual, or a class and a property. However, we argue that the official semantics of OWL 2, the so-called *Direct Semantics* (DS) treats punning in a way that is not adequate for metamodeling and metaquerying. The reason is simply that proper metamodeling requires that the *same* element plays the role of both individual and class (or, class and property), while, coherently with the standard DL literature, DS relies on the notion of interpretation of the standard First-Order Logic (FOL) semantics and, thus, it treats punning by sanctioning that an individual and a class with the same name are different elements.

The following example illustrates how proper metamodeling can be useful in modeling a domain, how it can be captured in OWL 2 QL, and why query answering under DS can be quite unsatisfactory².

Example 1. Consider the domain of human resource management, within an aeronautical organization. Each employee in the organization has a salary, and is classified according to several types, i.e., “Engineer”, “Pilot”, “Secretary”, which are naturally modeled as subclasses of the class Employee. Each employee type

¹http://www.w3.org/TR/owl2-new-features/#F12:_Punning

²This example is inspired by an example in [8], exhibiting the need for the so-called multi-level modeling.

is associated with a standard salary, and is in turn classified into two categories: “Technical Employee Type”, and “Administrative Employee Type”. So, for example, “John” is an engineer who earns 200.000 euros, which is exactly the standard salary associated with the employee type “Engineer”. Also, “Engineer” and “Pilot” are technical employee types, as opposed to “Secretary”, which is an example of administrative employee type.

To model this domain, one needs to deal with entities at different classification levels, such as individual employees (“John”), employee types (“Engineer”, “Pilot”, “Secretary”), and types of employee types (“Technical Employee Type”, “Administrative Employee Type”). In particular, one needs to assert that “John” is an instance of the class “Engineer” and that the pair (“John”, “200.000”) is an instance of the property “has salary”. Moreover, one needs to predicate on metaclasses, such as “Technical Employee Type”, e.g., by asserting that “Engineer” is one of its instances, as well as on metaproperties, e.g., by asserting that the pair (“Engineer”, “200.000”) is an instance of the property “has standard salary”. While this can be easily expressed in OWL 2 QL by exploiting punning, the fact that DS treats as distinct elements different occurrences of the same entity at different classification levels prevents from extracting interesting information from the knowledge base. Indeed, suppose that one wants to know which employees have a salary that coincides with the standard salary associated with their employee types. The obvious query expression would be as follows:

$$q(x) \leftarrow Employee(x) \wedge has_salary(x, y) \wedge z(x) \wedge has_standard_salary(z, y).$$

Note that while this query would not be allowed in the standard DL literature because of the presence of variables in predicate positions, it is allowed in SPARQL. However, since in DS an individual and a class with the same name are different elements, under such semantics the occurrence of z in class position in the third atom is completely unrelated to the occurrence of z in individual position in the fourth atom, and therefore no element can be simultaneously bound to both occurrences. It is exactly for this reason that, under the DS entailment regime for dealing with SPARQL queries, the above query is not even legal. This implies that, both under DS and under the DS entailment regime of SPARQL, we miss the information that “John” is an engineer earning the amount of the standard salary for the type “Engineer”. \square

The main goal of this paper is to overcome the above mentioned drawback, presenting an approach where both metamodeling and metaquerying can be profitably used in ontology specification and in computing the answers to queries posed to ontologies.

Contributions. This work builds upon the preliminary works [14] and [15]³, and extends them by providing a systematic study of metamodeling and meta-

³In fact, in [14] and [15], we used the name *Higher Order Semantics (HOS)* to denote what we call here Metamodeling Semantics. We prefer the latter because, even though the proposed semantic structure has a second-order flavor, its expressive power does not exceed first-order.

querying in OWL 2 QL, along with algorithms and proofs. Specifically, we present a new semantics for OWL 2 QL, called *Metamodeling Semantics*, that effectively exploits the metamodeling capabilities offered by OWL 2, and address fundamental problems of OWL 2 QL interpreted under such semantics, analogous to those addressed for the seminal study of $DL-Lite_R$ [5], namely:

- classical DL reasoning tasks, such as satisfiability, and class and property subsumption;
- metaquery answering, with the main goal of singling out possible sources of complexity within metaqueries and ontologies that lead to intractability.

We fully characterize the complexity of query answering, and identify significant classes of metaqueries and ontologies, likely to be of interest in practice, for which metaquery answering is tractable. In particular, we focus on metaqueries, simply called queries from now on, that have the form of unions of conjunctive queries that may contain variables in class and/or property position, which correspond to unions of SPARQL Basic Graph Patterns. From a technical point of view, our investigation shows that the notion of canonical model and homomorphism, which are crucial for the study of query answering in $DL-Lite_R$, can be extended for the study of metaquerying in OWL 2 QL under the metamodeling semantics as well.

More precisely, the main contributions of this work can be summarized as follows.

- We present the *Metamodeling Semantics (MS)* for OWL 2 and unions of conjunctive queries over ontologies interpreted under MS. Although our semantics is defined for the whole OWL 2 language, we formulate it in terms of OWL 2 QL, which is the subject of our investigation on the complexity of answering queries carried out in the subsequent sections. The main goal of MS is to provide a clean model-theoretic semantics for queries with unrestricted use of variables and metavariables, i.e., in which atoms may contain variables in class and/or property position.
- We show that for OWL 2 QL ontologies, satisfiability under MS and DS are equivalent, in the sense that any ontology is satisfiable under MS if and only if it is satisfiable under DS. The same holds for entailment of standard first-order unions of conjunctive queries, i.e., unions of conjunctive queries consisting only of ABox atoms with no metavariables. In other words, as far as OWL 2 QL is concerned, MS is a conservative extension of DS: the difference between the two semantics shows up only in answering queries that go beyond standard first-order unions of conjunctive queries.
- We show that entailment of instance queries, i.e., unions of conjunctive queries including only ABox atoms, possibly with metavariables, over OWL 2 QL ontologies under MS is different from entailment under DS, in the sense that queries that are not legal under DS are perfectly legal under MS, and are assigned a semantics that is coherent with the intuitive meaning of metamodeling. In particular, we show that the metavariables

appearing in the query can be treated by means of metagrounding, which is a technique based on the substitution of metavariables with the classes, object properties and data properties appearing in the ontology. Based on this property, we show that entailment of instance queries under MS has exactly the same complexity as entailment of standard first-order unions of conjunctive queries under DS, i.e., it is in AC^0 w.r.t. ABox complexity, in PTIME w.r.t. ontology complexity, and NP-complete w.r.t. combined complexity.

- As for general queries, i.e., possibly containing both metavariables and TBox atoms, over OWL 2 QL ontologies under MS, we are able to single out a specific source of complexity making the reasoning task more complex with respect to instance queries. Such source of complexity is related to the presence of a form of incompleteness in the TBox axioms of the ontology. To prove the increase in complexity, we provide the following lower complexity bounds for conjunctive queries: coNP-completeness w.r.t. ontology complexity, and Π_2^P -completeness w.r.t. combined complexity. We point out that both these results sharpen the intractability results in [13], which are based on the presence of union.
- We show that, in the special case of TBox-complete ontologies, i.e., in the case of ontologies that do not exhibit the above-mentioned form of incompleteness, the complexity of entailment of general queries under MS is the same as in the case of entailment of instance queries. This confirms that TBox incompleteness is indeed a distinguished source of complexity for answering general queries.
- Finally, we address the complexity upper bounds for entailment of general queries over unrestricted ontologies under MS, by presenting an algorithm matching the established lower bounds, i.e., that is in AC^0 w.r.t. ABox complexity, in coNP w.r.t. ontology complexity and in Π_2^P w.r.t. combined complexity. As far as we know, this is the first decidability result for general metaqueries posed to ontologies expressed in (fragments of) OWL 2. Indeed, the decidability results reported in [13] hold only for a specific classes of queries, called guarded. We also point out that our result on ABox complexity is the first tractability results of answering metaqueries, since in [13], answering (guarded) general metaqueries is shown to be NP-complete in ABox-complexity, too (see the discussion in the related work section below). The complexity results for entailment of queries are summarized in Table 1.

Related work. While the study of knowledge representation languages with higher-order or multi-level modeling capabilities is far from being new, the interest in such issue in the context of OWL, the W3C standard ontology language, is recent. We focus on such a line of research here, which is the relevant one for our investigation, and concentrate our attention on research works addressing the computational properties of reasoning tasks in metamodeling extensions of OWL 2 DL, or fragments therein. We remind the reader that OWL 2 DL is the

Ontology	Complexity	Instance queries	General queries
TBox-complete	<i>ABox</i>	AC^0	AC^0
	<i>Ontology</i>	PTime	PTime
	<i>Combined</i>	NP-complete	NP-complete
General	<i>ABox</i>	AC^0	AC^0
	<i>Ontology</i>	PTime	coNP-complete
	<i>Combined</i>	NP-complete	Π_2^P -complete

Table 1: Computational complexity of query entailment under MS

logic-based decidable restriction of OWL 2, and corresponds to a very expressive Description Logic.

Following a line of reasoning similar to that of [16], higher-order languages can be classified depending on whether they admit a *higher-order syntax*, which informally means that they allow for the use of variables in predicate position, and/or for the use of predicates in individual positions, or they have a *higher-order semantics*, which means that variables may range over sets constructed out of the domains of individuals. Works such as [17, 18] aim at defining languages that have both a higher-order syntax and semantics, and study the problem of checking satisfiability of an ontology expressed in such languages. Other works focus on languages having a higher-order syntax and a first-order semantics. In particular, [19] shows that OWL 2 DL, when enriched with the metamodeling features of OWL 2, such as the ability of stating (i) axioms where the same entity plays simultaneously the role of class, individual and/or property, and (ii) axioms over the reserved vocabulary (e.g., `rdf:type` and `owl:someValuesFrom`), is undecidable.

To achieve decidability, [20] studies the logic obtained by extending the DL underpinning OWL 2, only with the ability of stating axioms over the `rdf:type` reserved predicate. In particular, it shows that such a logic is decidable, and studies the complexity of satisfiability and subsumption for it. With the same goal in mind, on the other hand, [19] proposes a new semantics for OWL 2 DL, enriched with the metamodeling features of OWL 2, which is inspired by HiLog [16]. In such semantics, intuitively, every object o in an interpretation domain is not only seen as an individual, but also as a class, a property and an attribute, by exploiting total functions which associate to every object appropriate sets of objects, pairs of objects, and pairs of objects and values, respectively.

Note, however, that all the works within this line of research investigate the satisfiability problem and (some of them) the subsumption problem, but not the query answering problem. The only exception is [13], where the authors study query answering over ontologies expressed in any DL extended with higher-order syntax, and interpreted under a semantics that is based on an interpretation structure again inspired by HiLog. In particular, they focus on so-called *guarded metaqueries*, i.e. queries for which every variable occurring in a TBox atom (in particular, in an “IS-A” atom), also occurs in a concept or role position of an ABox atom, and show that answering guarded metaqueries with union is

coNP-hard w.r.t. ABox complexity and Π_2^P -hard w.r.t. combined complexity. Note that the fact that metaquery answering is intractable in ABox complexity under the semantics of [13], while it is tractable under the MS proposed here, can be explained by the difference in the semantics, and, specifically, by the fact that under the semantics of [13], every individual occurring in the ABox simultaneously plays the role of class and property, and therefore induces a form of “reasoning by cases”. Under MS, on the contrary, only entities playing the role of classes and/or properties may induce such form of reasoning, which is in fact at the origin of the intractability in ontology complexity. Note also, that our results sharpen the lower-bounds of metaquery answering shown in [13] w.r.t. both ontology and combined complexity, because we show that they already hold for conjunctive metaqueries, i.e. metaqueries without union.

We cannot conclude the discussion on the related work without mentioning works on the problem of answering SPARQL queries over OWL 2. Such papers are relevant since SPARQL natively allows expressing metaqueries. In particular, [21, 22, 23, 24] consider SPARQL under the Direct Semantics Entailment Regime, while [25, 26, 27, 28] propose to modify the semantics of SPARQL to drop its “active domain restriction”. Similarly to our approach, this amounts to enable metaqueries where pure existential variables occur and consequently reconcile SPARQL solution mappings with the usual notion of certain answers. However, all such works, in fact, concentrate on OWL 2 DL, i.e., the restriction of OWL 2 not comprising any higher-order feature. In other words, they study metaquerying over ontologies that do not contain metamodeling structures.

All the above observations show that, with the only exception of [13], no work has investigated semantical and computational properties of metaquerying over ontologies that both are expressed in OWL 2 or its variants, and comprise metamodeling constructs.

Plan of the paper. The paper is organized as follows. In Section 2, we review the main aspects of the syntax of OWL 2 QL ontologies and metaqueries, we introduce our new semantics for both metamodeling and metaquerying, and we investigate the relationship between such semantics and the standard one based on DS. In Section 3, we introduce the chase procedure for our logic, that forms the basis for the main decidability and complexity results illustrated in the paper. In Section 4 we deal with entailment of instance queries in OWL 2 QL ontologies under MS. In Section 5 we study the complexity lower bounds of checking entailment of general queries under MS. In Section 6 we concentrate on the complexity upper bounds, and present algorithms both for the case of TBox complete ontologies, and for unrestricted ontologies. Section 7 concludes the paper by discussing future research directions of our work.

2. OWL 2 QL and queries under MS

The goal of this section is to present the Metamodeling Semantics for OWL 2 QL ontologies, to define a new entailment regime for queries over OWL 2 QL ontologies based on MS, and to investigate the relationship between the standard Direct Semantics (DS) and MS. To achieve our goal, we start by recalling the basics of the OWL 2 QL syntax.

2.1. The OWL 2 QL syntax

In a nutshell, an ontology is a set of axioms used to express knowledge about a set of *elements* that are relevant for the domain of interest. According to the OWL 2 jargon, ontology elements are classified into “literals” and “entities”.

Literals are ontology elements corresponding to values belonging to datatypes. The set of OWL 2 QL literals is denoted L_{QL} and is characterized by the *OWL 2 QL datatype map* DM_{QL} , defined as the restriction to OWL 2 QL of the standard OWL 2 datatype map. In particular, DM_{QL} is defined as

$$DM_{\text{QL}} = (\Delta^v, \mathbb{D}, N_{LS}, \cdot^{DT}, \cdot^{LS})$$

where

- Δ^v , called the *value domain*, is the set of all values that can be represented in OWL 2 QL;
- \mathbb{D} is the set of datatypes of OWL 2 QL (which comprises `rdfs:Literal`);
- N_{LS} is the function that assigns to each $t \in (\mathbb{D} \setminus \{\text{rdfs:Literal}\})$ the *lexical space* of t , i.e. the set of the *lexical forms* used in OWL 2 ontologies to refer to data values of type t ; note that `rdfs:Literal` has not any lexical form;
- \cdot^{DT} is the function that assigns to each $t \in \mathbb{D}$ the so-called *value space*, i.e. the set of data values represented by t in all OWL 2 ontologies; note, in particular, that `rdfs:Literal` $\cdot^{DT} = \Delta^v$;
- \cdot^{LS} is the function that specifies, for each literal $(l, t) \in L_{\text{QL}}$, which is the data value in t^{DT} denoted by such literal.

Thus, each literal l in L_{QL} denotes the same value l^{LS} in every ontology, and has the form of a pair (s, t) , where s is the lexical form of the literal, and t is a datatype in $(\mathbb{D} \setminus \{\text{rdfs:Literal}\})$, such that s belongs to the lexical space of t . In other words, each literal makes explicit both the name and the datatype of the value it denotes.

It is worth noting that the computational properties of reasoning in OWL 2 QL are guaranteed by the following property of \mathbb{D} : for any set $\{t_1, t_2, \dots, t_n\}$, with $t_i \in \mathbb{D}, i = 1, \dots, n$, the intersection of the value spaces of the datatypes in the set, is either empty or infinite.

Entities are ontology elements corresponding to individuals, classes, object properties, data properties and datatypes in the domain of interest. Entities are denoted by *expressions*, built on the basis of the symbols of a vocabulary. Following the same line of the standard OWL 2 QL specification, we next introduce the notion of vocabulary and expression in such language. A *vocabulary* V is constituted by the tuple

$$V = (V_e, V_c, V_p, V_d, \mathbb{D}, V_i, L_{\text{QL}})$$

where

- V_e is the union of V_c , V_p , V_d , and V_i , and its elements are called *atomic expressions*,
- V_c (resp., V_p , V_d , V_i) is a non-empty set of *Internationalized Resource Identifiers (IRIs)* identifying classes (resp., object properties, data properties, datatypes, and individuals),
- \mathbb{D} and L_{QL} are defined according to DM_{QL} , and
- (i) V_c (resp., V_p , and V_d) includes the special symbols \top_c and \perp_c (resp., \top_p and \perp_p , and \top_d and \perp_d), (ii) L_{QL} is disjoint from all other components of V , and (iii) V_c (V_p) is disjoint from \mathbb{D} (resp., V_d).

It is worth noting that, based on the definition of V , an atomic expression may denote an element simultaneously playing different roles within an ontology, such as, for example, the role of an individual and of a class. This feature is known as *OWL 2 punning*, and will be discussed in more details at the end of this subsection. Note also that item (iii) above prevents an expression to denote both a class and a datatype, or both an object property and a data property. Given a vocabulary V we next define the set of well-formed expressions over V . To this aim, let us define the following:

- $Exp_p(V) = V_p \cup \{e^- \mid e \in V_p\}$ is the set of *object property expressions*;
- $Bas_c(V) = V_c \cup \{\exists e \mid e \in Exp_p(V)\} \cup \{\delta(e_1).e_2 \mid e_1 \in V_d, e_2 \in \mathbb{D}\}$ is the set of *basic class expressions*;
- $Exp_c(V) = Bas_c(V) \cup \{\exists e_1.e_2 \mid e_1 \in Exp_p(V), e_2 \in V_c\}$ is the set of *class expressions*;
- the set of *data property expressions* coincides with V_d ;
- $Exp_t(V) = \mathbb{D} \cup \{\rho(d) \mid d \in V_d\}$ is the set of *datatype expressions*;
- the set of *individual expressions* coincides with V_i .

Then, *the set $Exp(V)$ of well-formed expressions over V* is the set

$$Exp(V) = Exp_c(V) \cup Exp_p(V) \cup V_d \cup Exp_t(V) \cup V_i.$$

Based on the previous definitions, an *OWL 2 QL ontology* (simply *ontology* in the following) over a vocabulary V is a finite set of logical axioms of the form listed in Table 2, where we adopt the following naming scheme (possibly including subscripts): (i) a , r , d , t , and i denote, respectively, an atomic class, an atomic object property, a data property, a datatype, and an individual expression; (ii) b denotes a basic class expression; (iii) c and p denote, respectively, a class and an object property expression; (iv) and, finally, l denotes a literal.

Positive TBox axioms	Negative TBox axioms
$b \sqsubseteq_c c$ (class inclusion)	$b_1 \sqsubseteq_c \neg b_2$ (class disjointness)
$p_1 \sqsubseteq_p p_2$ (object property inclusion)	$p_1 \sqsubseteq_p \neg p_2$ (object property disjointness)
$d_1 \sqsubseteq_d d_2$ (data property inclusion)	$d_1 \sqsubseteq_d \neg d_2$ (data property disjointness)
$\rho(d) \sqsubseteq_t t$ (value domain inclusion)	
$\text{Ref}(p)$ (reflexive object property)	$\text{Irr}(p)$ (irreflexive object property)
ABox axioms	
$a(i)$ (class membership)	
$r(i_1, i_2)$ (object property membership)	
$d(i, l)$ (data property membership)	

Table 2: OWL 2 QL axioms

As the table shows, axioms are partitioned into (i) *TBox axioms*, in turn partitioned into *negative* and *positive*, and (ii) *ABox axioms*. Note that, in this paper, we omit to deal with OWL 2 QL axioms that can be expressed by appropriate combinations of the axioms specified in Table 2, such as axioms expressing object property (a)symmetry, class (object or data property) equivalence, or disjointness of more than two classes (object or data properties). Similarly, we do not consider axioms allowing for the definition of new datatypes, since the restrictions on the use of datatypes in OWL 2 QL are such that they do not add expressive power to the language. Also, we do not consider axioms of the form `DifferentIndividuals`, because they correspond to inequalities, whose unrestricted use in conjunctive queries leads to insurmountable computational obstacles of query answering, already for *DL-Lite_R*, i.e., the logic underpinning OWL 2 QL, interpreted under the standard first-order semantics [29]. Finally, in what follows we assume that all symbols a , r and d appearing in the ABox axioms $a(i)$, $r(i_1, i_2)$ and $d(i, l)$ of an ontology \mathcal{O} , also appear in the TBox of \mathcal{O} . Note that this assumption does not hamper generality, because it can always be satisfied by introducing axioms of the forms $a \sqsubseteq_c \top_c$, $r \sqsubseteq_p \top_p$ or $d \sqsubseteq_d \top_d$ into the TBox without changing the semantics of the ontology.

Another notion that will be useful in the next sections is the one of signature of an ontology. Coherently with the DL literature (e.g., [1]) the *signature* of an ontology \mathcal{O} , denoted $\Sigma(\mathcal{O})$, is the tuple

$$(\Sigma_c(\mathcal{O}), \Sigma_p(\mathcal{O}), \Sigma_d(\mathcal{O}), \Sigma_t(\mathcal{O}), \Sigma_i(\mathcal{O}), \Sigma_{Lit}(\mathcal{O}))$$

where $\Sigma_c(\mathcal{O})$, $\Sigma_p(\mathcal{O})$, $\Sigma_d(\mathcal{O})$, $\Sigma_t(\mathcal{O})$, $\Sigma_i(\mathcal{O})$, $\Sigma_{Lit}(\mathcal{O})$ are respectively the subset of the elements of V_c , V_p , V_d , D , V_i , and L_{QL} that occur in \mathcal{O} . Also, we let $Exp(\Sigma(\mathcal{O}))$ ($Exp_c(\Sigma(\mathcal{O}))$, $Exp_p(\Sigma(\mathcal{O}))$, and $Exp_t(\Sigma(\mathcal{O}))$) denote the set of expressions (class expressions, object expressions and datatype expressions) that can be built over $\Sigma(\mathcal{O})$. We observe that $Exp(\Sigma(\mathcal{O}))$ is clearly finite and, even though it is built over the elements occurring in \mathcal{O} , it may contain expressions that do not occur in \mathcal{O} .

Let us now come back to OWL 2 punning. As already mentioned, “punning” is

the term used to denote the capability of OWL 2 atomic expressions to designate elements that simultaneously play different roles in an ontology. This means that an entity can occur in different *position* types, or simply *positions*, within the ontology axioms. More precisely, an entity is said to appear in: (i) *class position*, if it appears in any position in a class inclusion or disjointness axiom, or as a predicate in a class membership axiom; (ii) *object property position*, if it appears in any position in a object property inclusion, object property disjointness, irreflexivity, or reflexivity axiom, or as a predicate in an object property membership axiom; (iii) *data property position*, if it appears in any position in a data property inclusion or disjointness axiom, or as a predicate in a data property membership axiom; (iv) *datatype position*, if it appears in the right hand side of a value domain inclusion axiom, or as second element of a literal $lt = (l, d)$ appearing as literal in a data property membership axiom; (v) *individual position*, if it appears in a position different from the ones mentioned in the above items.

The following example illustrates how the features of OWL 2 QL, in particular punning, can be exploited to capture relevant meta-level properties of the domain discussed in Example 1.

Example 2. It is easy to verify that the domain described in Example 1. can be captured by the ontology in Fig. 1. For example, axioms (1) to (3) state that *Engineer*, *Pilot*, and *Secretary* are subclasses of *Employee*, and therefore they are concepts referring to types of employment. Axioms (4) and (5) state that *Secretary* is disjoint with both *Engineer* and *Pilot*. Note that the expression *Engineer* occurs both in class position (in axioms (1), (4), and (14)) and in individual position (in axioms (11) and (16)), which means that besides predicating on individuals, e.g., by stating that *John* is an engineer, the ontology states that engineer is a technical employee type (axiom (11)). \square

2.2. The Metamodeling Semantics

The Metamodeling Semantics (MS) is based on the notion of MS-interpretation, that we now define. In what follows, $\mathcal{P}(S)$ denotes the power set of the set S .

If \mathcal{O} is an ontology over the vocabulary V , an *MS-interpretation* for \mathcal{O} is a tuple $\mathcal{I} = \langle \Delta^o, \Delta^v, \cdot^I, \cdot^C, \cdot^P, \cdot^D, \cdot^T, \cdot^{\mathcal{I}} \rangle$ where:

- Δ^o , the *object domain*, is a non-empty set disjoint from the *value domain* Δ^v defined by the datatype map DM_{QL} ;
- $\cdot^C : \Delta^o \rightarrow \mathcal{P}(\Delta^o)$ is a partial function;
- $\cdot^P : \Delta^o \rightarrow \mathcal{P}(\Delta^o \times \Delta^o)$ is a partial function;
- $\cdot^D : \Delta^o \rightarrow \mathcal{P}(\Delta^o \times \Delta^v)$ is a partial function;
- $\cdot^T : \Delta^o \rightarrow \mathcal{P}(\Delta^v)$ is a partial function;

<i>Engineer</i> \sqsubseteq_c <i>Employee</i>	(1)
<i>Pilot</i> \sqsubseteq_c <i>Employee</i>	(2)
<i>Secretary</i> \sqsubseteq_c <i>Employee</i>	(3)
<i>Engineer</i> \sqsubseteq_c \neg <i>Secretary</i>	(4)
<i>Pilot</i> \sqsubseteq_c \neg <i>Secretary</i>	(5)
<i>TechnicalEmployeeType</i> \sqsubseteq_c <i>EmployeeType</i>	(6)
<i>AdministrativeEmployeeType</i> \sqsubseteq_c <i>EmployeeType</i>	(7)
<i>TechnicalEmployeeType</i> \sqsubseteq_c \neg <i>AdministrativeEmployeeType</i>	(8)
<i>Employee</i> \sqsubseteq_c \exists <i>has_salary</i>	(9)
<i>EmployeeType</i> \sqsubseteq_c \exists <i>has_standard_salary</i>	(10)
<i>TechnicalEmployeeType</i> (<i>Engineer</i>)	(11)
<i>TechnicalEmployeeType</i> (<i>Pilot</i>)	(12)
<i>AdministrativeEmployeeType</i> (<i>Secretary</i>)	(13)
<i>Engineer</i> (<i>John</i>)	(14)
<i>has_salary</i> (<i>John</i> , ("200.000", <i>xsd:integer</i>))	(15)
<i>has_standard_salary</i> (<i>Engineer</i> , ("200.000", <i>xsd:integer</i>))	(16)

Figure 1: Multi-level OWL 2 QL ontology

- $\cdot^I : \Delta^o \rightarrow \{\mathbf{true}, \mathbf{false}\}$ is a total function such that for each $o \in \Delta^o$, if $\cdot^C, \cdot^P, \cdot^D, \cdot^T$ are all undefined for o , or o is in the range of \cdot^C , or o appears in a pair in the range of \cdot^P or as first component of a pair in the range of \cdot^D , then $o^I = \mathbf{true}$.
- $\cdot^{\mathcal{I}}$ is a function that maps every expression in $Exp(V)$ into an object in Δ^o , and every literal in L_{QL} into a value in Δ^v .
- the conditions specified in Table 3.A are satisfied.

In the following, we denote by \mathcal{W} the tuple $\langle \Delta^o, \Delta^v, \cdot^I, \cdot^C, \cdot^P, \cdot^D, \cdot^T \rangle$ constituted by the first seven components of an MS-interpretation, and therefore we often write \mathcal{I} as the pair $\langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle$. Also, we call \mathcal{W} the *interpretation structure* of \mathcal{I} , $\cdot^{\mathcal{I}}$ the *interpretation function* of \mathcal{I} , and $\Delta^o \cup \Delta^v$ the *domain* of \mathcal{I} . This is done both for notational convenience, and for emphasizing the two main roles of an MS-interpretation, namely specifying a structured world (represented by \mathcal{W}), and mapping the symbols of the vocabulary to such world (function $\cdot^{\mathcal{I}}$). Here is a list of observations pointing out the characteristics of the various components of \mathcal{I} .

- Each element in the world represented by \mathcal{W} is in $\Delta = \Delta^o \cup \Delta^v$, and therefore is either an object or a value.
- \cdot^I is the boolean function specifying whether an object o is an individual object (if o^I is true), or not (if o^I is false) in the world represented by \mathcal{W} .

Table 3.A: conditions on \mathcal{I}	
if $i \in V_i$, then $(i^{\mathcal{I}})^I = \mathbf{true}$; if $c \in Exp_c(V)$, then \cdot^C is defined for $c^{\mathcal{I}}$, and \cdot^T is undefined for $c^{\mathcal{I}}$; if $p \in Exp_p(V)$, then \cdot^P is defined for $p^{\mathcal{I}}$, and \cdot^D is undefined for $p^{\mathcal{I}}$; if $d \in V_d$, then \cdot^D is defined for $d^{\mathcal{I}}$, and \cdot^P is undefined for $d^{\mathcal{I}}$; if $t \in Exp_t(V)$, then \cdot^T is defined for $t^{\mathcal{I}}$, and \cdot^C is undefined for $t^{\mathcal{I}}$; if $t \in \mathbb{D}$, then $(t^{\mathcal{I}})^T = t^{DT}$; if $l = (\mathbf{s}, t) \in L_{\mathbb{QL}}$, then $l^{\mathcal{I}} = (\mathbf{s}, t)^{LS}$; $((\top_c)^{\mathcal{I}})^C = \{o \mid o \in \Delta^o, o^I = \mathbf{true}\}$; $((\perp_c)^{\mathcal{I}})^C = \emptyset$; $((\exists p)^{\mathcal{I}})^C = \{o_1 \mid \exists \langle o_1, o_2 \rangle \in (p^{\mathcal{I}})^P\}$; $((\exists p.c)^{\mathcal{I}})^C = \{o_1 \mid \exists \langle o_1, o_2 \rangle \in (p^{\mathcal{I}})^P, o_2 \in (c^{\mathcal{I}})^C\}$; $((\delta(d).t)^{\mathcal{I}})^C = \{o \mid \exists \langle o, v \rangle \in (d^{\mathcal{I}})^D, v \in (t^{\mathcal{I}})^T\}$; $((\top_p)^{\mathcal{I}})^P = \{o \mid o \in \Delta^o, o^I = \mathbf{true}\} \times \{o \mid o \in \Delta^o, o^I = \mathbf{true}\}$; $((\perp_p)^{\mathcal{I}})^P = \emptyset$; $((p^-)^{\mathcal{I}})^P = ((p^{\mathcal{I}})^P)^{-1}$; $((\top_d)^{\mathcal{I}})^D = \{o \mid o \in \Delta^o, o^I = \mathbf{true}\} \times \Delta^v$; $((\perp_d)^{\mathcal{I}})^D = \emptyset$; $((\rho(d))^{\mathcal{I}})^T = \{v \mid \exists \langle o, v \rangle \in (d^{\mathcal{I}})^D\}$.	
Table 3.B: satisfaction of axioms by \mathcal{I}	
$\mathcal{I} \models b \sqsubseteq_c c$	if \cdot^C is defined for both $b^{\mathcal{I}}$ and $c^{\mathcal{I}}$, and $(b^{\mathcal{I}})^C \subseteq (c^{\mathcal{I}})^C$;
$\mathcal{I} \models p_1 \sqsubseteq_p p_2$	if \cdot^P is defined for both $p_1^{\mathcal{I}}$ and $p_2^{\mathcal{I}}$, and $(p_1^{\mathcal{I}})^P \subseteq (p_2^{\mathcal{I}})^P$;
$\mathcal{I} \models d_1 \sqsubseteq_d d_2$	if \cdot^D is defined for both $d_1^{\mathcal{I}}$ and $d_2^{\mathcal{I}}$, and $(d_1^{\mathcal{I}})^D \subseteq (d_2^{\mathcal{I}})^D$;
$\mathcal{I} \models \rho(d) \sqsubseteq_t t$	if \cdot^T is defined for both $\rho(d)^{\mathcal{I}}$ and $t^{\mathcal{I}}$, and $(\rho(d)^{\mathcal{I}})^T \subseteq (t^{\mathcal{I}})^T$;
$\mathcal{I} \models \text{Ref}(p)$	if \cdot^P is defined for $p^{\mathcal{I}}$, and $\forall o \in \Delta^o : \langle o, o \rangle \in (p^{\mathcal{I}})^P$;
$\mathcal{I} \models b_1 \sqsubseteq_c \neg b_2$	if \cdot^C is defined for both $b_1^{\mathcal{I}}$ and $b_2^{\mathcal{I}}$, and $(b_1^{\mathcal{I}})^C \cap (b_2^{\mathcal{I}})^C = \emptyset$;
$\mathcal{I} \models p_1 \sqsubseteq_p \neg p_2$	if \cdot^P is defined for both $p_1^{\mathcal{I}}$ and $p_2^{\mathcal{I}}$, and $(p_1^{\mathcal{I}})^P \cap (p_2^{\mathcal{I}})^P = \emptyset$;
$\mathcal{I} \models d_1 \sqsubseteq_d \neg d_2$	if \cdot^D is defined for both $d_1^{\mathcal{I}}$ and $d_2^{\mathcal{I}}$, and $(d_1^{\mathcal{I}})^D \cap (d_2^{\mathcal{I}})^D = \emptyset$;
$\mathcal{I} \models \text{Irr}(p)$	if \cdot^P is defined for $p^{\mathcal{I}}$, and $\forall o \in \Delta^o : \langle o, o \rangle \notin (p^{\mathcal{I}})^P$;
$\mathcal{I} \models a(i)$	if $(i^{\mathcal{I}})^I = \mathbf{true}$, \cdot^C is defined for $a^{\mathcal{I}}$, and $i^{\mathcal{I}} \in (a^{\mathcal{I}})^C$;
$\mathcal{I} \models r(i_1, i_2)$	if \cdot^P is defined for $r^{\mathcal{I}}$, $(i_1^{\mathcal{I}})^I = (i_2^{\mathcal{I}})^I = \mathbf{true}$, and $\langle i_1^{\mathcal{I}}, i_2^{\mathcal{I}} \rangle \in (r^{\mathcal{I}})^P$;
$\mathcal{I} \models d(i, l)$	if \cdot^D is defined for $d^{\mathcal{I}}$, $(i^{\mathcal{I}})^I = \mathbf{true}$, and $\langle i^{\mathcal{I}}, l^{\mathcal{I}} \rangle \in (d^{\mathcal{I}})^D$.

 Table 3: Conditions on \mathcal{I} and satisfaction of axioms

- \cdot^C is the function that, given a domain object o , either assigns to o a set of objects forming its extension as a class (in this case, o is called class), or is undefined for o , the latter case reflecting that o is not a class in the world represented by \mathcal{W} .
- \cdot^P is the function that, given a domain object o , either assigns to o a binary relation (a set of object pairs) constituting its extension as an object property (in this case, o is called object property), or is undefined for o , the latter case reflecting that o is not an object property in the world represented by \mathcal{W} .
- \cdot^D and \cdot^T are defined similarly, but for data properties and datatypes,

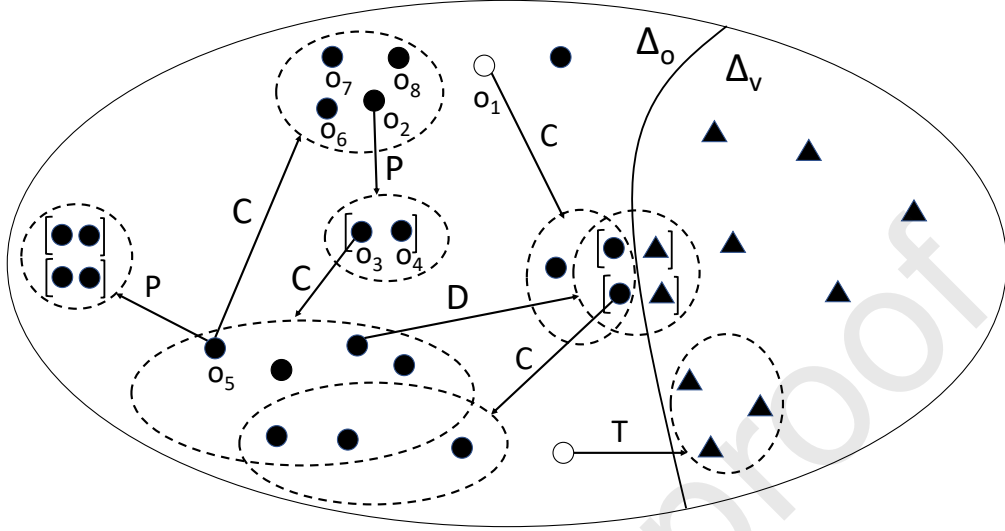


Figure 2: Example of interpretation structure

respectively.

Taking into account the above comments and compared to the usual notion of Description Logic interpretation, the interpretation structure \mathcal{W} appears in place of the classical interpretation domain Δ . So, while the world in a classical interpretation is constituted by a flat domain, in \mathcal{W} the objects are polymorphic, in the sense that each of them can simultaneously be an individual object (this is the case where function \cdot^I is **true**), a class (this is the case where the partial function \cdot^C is defined), an object property (\cdot^P is defined), a data property (\cdot^D is defined), and a value type (\cdot^T is defined). On the other hand, as usual in logic, the interpretation function \cdot^I states how to assign a meaning to every syntactic element (expressions in $Exp(V)$) into an object in the interpretation domain $\Delta^o \cup \Delta^v$. Fig. 2 shows an example of interpretation structure, where blank and full bullets denote object domains o such that o^I is false and true, respectively, triangles denote data values, and labeled arrows denote, for each object, its assignment through the functions \cdot^C , \cdot^P , \cdot^D , and \cdot^T , if defined. In particular, in the interpretation structure in Fig. 2, o_1 is a class, and is neither an individual object, nor an object property, nor a data property, nor a datatype. On the other hand, o_2 is both an individual and an object property whose set of instances contain the pair of individual objects (o_3, o_4) , while it is neither a class, nor a data property, nor a datatype; o_5 is both a class, and an object property, and, as a class, has o_2, o_6, o_7 and o_8 among its instances.

As usual, to define the semantics of logical axioms, we refer to the notion of satisfaction of an axiom with respect to an MS-interpretation \mathcal{I} . The rules

defining such notion are specified in Table 3.B. An MS-interpretation satisfying all the axioms of an ontology \mathcal{O} is called a *model*, in particular an *MS-model* of \mathcal{O} , and we denote by $Mod_{MS}(\mathcal{O})$ the set of MS-models of \mathcal{O} .

Note that the definition of MS-interpretation is valid for the whole OWL 2 and, actually, for every Description Logic. The only aspect that is specific to OWL 2 QL, which is the logic of interest in this paper, is the set of conditions reported in Table 3. If one wants to specialize MS to another language, it is sufficient to appropriately change Table 3 so as to take into account the form of expressions and the form of axioms in such language.

Let us come back to metamodeling in OWL 2, and in particular in OWL 2 QL. The following example illustrates how an ontology exploiting punning is interpreted under MS.

Example 3. Consider the ontology of Fig. 1, and observe that *Engineer* occurs both in class position (axiom (1) and axiom (14)), and in individual position (axiom (11)); so *Engineer* belongs both to V_c and V_i . Now, let \mathcal{I} be an MS-models of the ontology, and let o be an object of Δ^o such that $Engineer^{\mathcal{I}} = o$. Hence, by definition of MS-model, we will have $o^I = \text{true}$, and, by virtue of axiom (14), \cdot^C is defined for o , and o^C will contain the object o' such that $o' = John^{\mathcal{I}}$. \square

2.3. Queries under MS

In this section we present the language we use to query ontologies interpreted under MS. We focus on the class of *union of boolean conjunctive queries*, but all the results presented in the paper can be easily extended to the case of non-boolean queries as well. Conjunctive queries are based on the notion of *query atom*, that in turn is based on the notion of *query term*.

Given an ontology \mathcal{O} over a vocabulary V and a set of variables \mathcal{V} disjoint from V , a *query term* (or simply *term*) is an expression contained in the set $Exp(V \cup \mathcal{V})$, i.e., a query term may involve a variable as atomic expression. A *query atom* (or simply *atom*) has the same form of an axiom, with the difference that query terms (and therefore variables) may occur in it. A *boolean conjunctive query* is an expression of the form

$$q \leftarrow Conj(\vec{y})$$

where $Conj(\vec{y})$, called *body of q* and denoted by $body(q)$, is a conjunction of atoms built over a tuple of terms \vec{y} . A *union of boolean conjunctive queries* (or simply a *query*) is an expression of the form

$$Q \leftarrow \bigcup_{1 \leq i \leq n} Conj_i(\vec{y}_i)$$

where $Conj_i(\vec{y}_i)$ is a conjunction of atoms built over tuples of terms \vec{y}_i for $i = 1, \dots, n$.

Since query atoms have the form of logical axioms, we also call them ABox (query) atoms or TBox (query) atoms in the obvious way. A conjunctive query whose body contains only ABox atoms is called *instance conjunctive query*. An *instance query* is a union of instance conjunctive queries. A query that is not an instance query will be called simply query, or *general query*, when we want to emphasize the difference with instance queries. We call *metavariable* of a query a variable that occurs in class position, object property position, dataproperty position or datatype position in the query, and we say that an atom is *ground* (resp. *metaground*) if no variable (resp. metavariable) occurs in it. A query is *ground* (resp. *metaground*) if all its atoms are ground (resp. metaground).

Example 4. Consider the ontology \mathcal{O} of Fig. 1, and suppose one wants to check the following conditions:

1. Is there an amount representing the standard salary earned by a pilot?
2. Is there an employee earning the standard salary of his/her type of employment?
3. Is there a technician who belongs to a class disjoint from pilot, and earns the standard salary of his/her type of employment?

Such conditions can be expressed by the following queries:

- $q_1 \leftarrow has_standard_salary(Pilot, y)$
- $q_2 \leftarrow Employee(x) \wedge has_salary(x, y) \wedge z(x) \wedge has_standard_salary(z, y)$
- $q_3 \leftarrow Employee(x) \wedge z(x) \wedge TechnicalEmployeeType(z) \wedge z \sqsubseteq_c \neg Pilot \wedge has_salary(x, y) \wedge has_standard_salary(z, y)$

Note that q_1 is a metaground instance query, q_2 is an instance query, and q_3 is a general query. Note also that q_2 is the boolean version of the query discussed in the introduction. \square

As for the semantics of Q under MS, we rely on the usual definition of the meaning of disjunction, conjunction, and existential quantification in logic, and we do not delve into the formal details. We simply observe that, given a boolean union of conjunctive queries Q expressed over the ontology \mathcal{O} , and an interpretation $\mathcal{I} = \langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle$ of \mathcal{O} , Q is true in \mathcal{I} if there is a conjunctive query q in Q that is true in \mathcal{I} , where the truth value of q in \mathcal{I} is established by checking if

there exists a variable assignment that associates to each variable in q an object or a value in the domain $\Delta^o \cup \Delta^v$ of \mathcal{I} such that the resulting atoms in $body(q)$ are all true in \mathcal{I} . Now, given an ontology \mathcal{O} and a query Q , we say that Q is *entailed by \mathcal{O} under the MS entailment regime* (or, simply, *under MS*), written $\mathcal{O} \models_{MS} Q$, if Q is true in every MS-model of \mathcal{O} .

Example 5. Consider the ontology \mathcal{O} and the query q_2 of Example 4. Clearly, q_2 is entailed by \mathcal{O} under MS. Indeed, for every MS-model \mathcal{I} of \mathcal{O} , we can assign to the variable x the object $John^{\mathcal{I}}$, to y the value ("200.000", $xsd:integer$) $^{\mathcal{I}}$, and to z the object $Engineer^{\mathcal{I}}$, and conclude that all atoms in $body(q_2)$ are true in \mathcal{I} under such assignment. \square

Actually, in our technical development, we use a characterization of query entailment based on the notion of query homomorphism, which is an extension of the well-known notion of homomorphism studied in [30]. Specifically, let \mathcal{O} be an ontology defined over the vocabulary V , let q be a conjunctive query over \mathcal{O} , and let $\mathcal{I} = \langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle$ be an MS-model of \mathcal{O} , with $\mathcal{W} = \langle \Delta^o, \Delta^v, \cdot^I, \cdot^C, \cdot^P, \cdot^D, \cdot^T \rangle$. A *query homomorphism* from q to \mathcal{I} is a function f from the terms of q to the domain $\Delta^o \cup \Delta^v$ of \mathcal{I} such that

- for every $e \in Exp(V) \cup L_{QL}$, $f(e) = e^{\mathcal{I}}$,
- for all terms e_1, e_2, e_3 (in what follows, when we write $f(e_i)^C$, $f(e_i)^P$, $f(e_i)^D$, and $f(e_i)^T$, for $i = 1, 2, 3$, we implicitly mean that \cdot^C , \cdot^P , \cdot^D , and \cdot^T , is respectively defined for $f(e_i)$):
 - if $e_1 \in Exp(V)$ occurs in individual position in $body(q)$, then $f(e_1)^I = \mathbf{true}$;
 - if $e_1(e_2) \in body(q)$, then $f(e_2) \in f(e_1)^C$;
 - if $e_1(e_2, e_3) \in body(q)$ and P is defined for $f(e_1)$ (implying that D is not defined for $f(e_1)$), then $(f(e_2), f(e_3)) \in (f(e_1))^P$;
 - if $e_1(e_2, e_3) \in body(q)$ and D is defined for $f(e_1)$ (implying that P is not defined for $f(e_1)$), then $(f(e_2), f(e_3)) \in f(e_1)^D$;
 - if $e_1 \sqsubseteq_c e_2 \in body(q)$, then $f(e_1)^C \subseteq f(e_2)^C$;
 - if $e_1 \sqsubseteq_p e_2 \in body(q)$, then $f(e_1)^P \subseteq f(e_2)^P$;
 - if $e_1 \sqsubseteq_d e_2 \in body(q)$, then $f(e_1)^D \subseteq f(e_2)^D$;
 - if $e_1 \sqsubseteq_t e_2 \in body(q)$, then $f(e_1)^T \subseteq f(e_2)^T$;
 - if $e_1 \sqsubseteq_c \neg e_2 \in body(q)$, then $f(e_1)^C \cap f(e_2)^C = \emptyset$;
 - if $e_1 \sqsubseteq_p \neg e_2 \in body(q)$, then $f(e_1)^P \cap f(e_2)^P = \emptyset$;
 - if $e_1 \sqsubseteq_d \neg e_2 \in body(q)$, then $f(e_1)^D \cap f(e_2)^D = \emptyset$;
 - if $Ref(e_1) \in body(q)$, then for every $e_2 \in \Delta^o$, $(e_2, e_2) \in f(e_1)^P$;
 - if $Irr(e_1) \in body(q)$, then for every $e_2 \in \Delta^o$, $(e_2, e_2) \notin f(e_1)^P$.

Furthermore, if Q is a union of boolean conjunctive queries, then any query homomorphism from a disjunct of Q to \mathcal{I} is said to be a *query homomorphism* from Q to \mathcal{I} .

Example 6. We showed in Example 5 that q_2 is entailed by \mathcal{O} under MS. We now show that there is a query homomorphism from q_2 to every MS-model of \mathcal{O} . Let $\mathcal{I} = \langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle$ be an MS-model of \mathcal{O} , such that $Employee^{\mathcal{I}} = o_1$, $Engineer^{\mathcal{I}} = o_2$, $has_salary^{\mathcal{I}} = o_3$, $has_standard_salary^{\mathcal{I}} = o_4$, $TechnicalEmployeeType^{\mathcal{I}} = o_5$, $Pilot^{\mathcal{I}} = o_6$, $John^{\mathcal{I}} = o_7$, and $(\text{"200.000"}, xsd:integer)^{\mathcal{I}} = v$, where $o_i \in \Delta^o$, for $i = 1, \dots, 7$, and $v \in \Delta^v$. Also, let f be a function from the terms of q_2 to \mathcal{I} , such that $f(Employee) = o_1$, $f(x) = o_7$, $f(has_salary) = o_3$, $f(\text{"200.000"}, xsd:integer) = v$, $f(z) = o_2$, and $f(has_standard_salary) = o_4$. It is easy to see that f is a query homomorphism from q_2 to \mathcal{I} , and that one such query homomorphism exists from q_2 to every MS-model of \mathcal{O} . \square

Based on the notion of query homomorphism, it is immediate to see that, analogously to what happens for conjunctive queries under the first-order semantics, query entailment under MS can be characterized in terms of query homomorphisms. This is formalized in the following proposition.

Proposition 7. *A union of boolean conjunctive queries Q is entailed by an OWL2QL ontology \mathcal{O} under MS if and only if there is a query homomorphism from Q to \mathcal{I} for every $\mathcal{I} \in Mod_{MS}(\mathcal{O})$.*

2.4. Relationships between MS and DS

In this subsection, we investigate the relationships between MS and DS. We remind the reader that the DS of OWL2 relies on the notion of interpretation of the standard First-Order Logic (FOL) semantics⁴. Specifically, given a vocabulary $V = (V_e, V_c, V_p, V_d, D, V_i, L_{QL})$, an *interpretation* \mathcal{I} of \mathcal{O} over V under DS is a 8-tuple $\langle \Delta^o, \Delta^v, \cdot^{CL}, \cdot^{OP}, \cdot^{DP}, \cdot^{DT}, \cdot^{IN}, \cdot^{LT} \rangle$ where the *interpretation domain* for \mathcal{I} consists of two disjoint sets Δ^o and Δ^v , and the other components of the tuple constitute the classical interpretation function for \mathcal{I} . Such an interpretation \mathcal{I} is a *DS-model* of \mathcal{O} if \mathcal{I} satisfies every axiom of \mathcal{O} under DS, and the set of DS-models of \mathcal{O} is denoted by $Mod_{DS}(\mathcal{O})$.

Example 8. Consider the ontology of Fig. 1. As already noticed in Example 3, *Engineer* occurs in class position in axiom (1) and in individual position in axiom (11). Let $\mathcal{M} = \langle \Delta^o, \Delta^v, \cdot^{CL}, \cdot^{OP}, \cdot^{DP}, \cdot^{DT}, \cdot^{IN}, \cdot^{LT} \rangle$ be a DS-model of \mathcal{O} for V . By definition, $Engineer^{CL}$ is a subset of Δ^o , while $Engineer^{IN}$ is an

⁴We refer here to the definition of *interpretation under DS*, provided at the W3C Recommendation "OWL 2 Web Ontology Language Direct Semantics (Second Edition)", (cf. <https://www.w3.org/TR/owl2-direct-semantics/>).

object in Δ^o . It follows that the two occurrences of *Engineer* in axioms (1) and (11) are seen by DS as referring to completely unrelated ontology elements. \square

As for queries, under DS, legal queries are obtained by restricting the general queries defined above to those satisfying the so-called *typing constraint*⁵, i.e., the condition that prevents the same variable from occurring in positions of different types⁶ We further illustrate this point by the following example.

Example 9. Consider the ontology \mathcal{O} of Fig. 1 and the queries q_2 , and q_3 of Example 4. It is easy to see that q_2 and q_3 are all entailed by \mathcal{O} under MS. On the contrary, q_2 and q_3 are not legal under DS, since the variable x occurs simultaneously in individual and class position. Note in particular that, even if the typing constraint was ignored, q_2 and q_3 would not be entailed by \mathcal{O} under DS because, as discussed in Example 8, different types of occurrences of the expression *Engineer* are interpreted under DS as if they were different ontology elements, and hence no element of the ontology exists that can be bound to the variable x in such a way to make the query true. \square

We are now able to establish the following strong relationship between the MS-models and the DS-models of an OWL 2 QL ontology.

Proposition 10. *For every OWL 2 QL ontology \mathcal{O} there is a function from $Mod_{MS}(\mathcal{O})$ to $Mod_{DS}(\mathcal{O})$ that is total and surjective.*

Proof. Let \mathcal{O} be an ontology over the vocabulary $V = (V_e, V_c, V_p, V_d, D, V_i, L_{QL})$, and consider the function γ from $Mod_{MS}(\mathcal{O})$ to $Mod_{DS}(\mathcal{O})$ defined as follows. For an MS-model $\mathcal{I} = \langle \mathcal{W}, \mathcal{I} \rangle$ of \mathcal{O} , with $\mathcal{W} = \langle \Delta^o, \Delta^v, \cdot, I, \cdot, C, \cdot, P, \cdot, D, \cdot, T \rangle$, we obtain the interpretation of \mathcal{O} under DS $\gamma(\mathcal{I}) = \langle \Delta^{o'}, \Delta^{v'}, \cdot, CL, \cdot, OP, \cdot, DP, \cdot, DT, \cdot, IN, \cdot, LT \rangle$ from \mathcal{I} by setting $\Delta^{o'} = \Delta^o$ and $\Delta^{v'} = \{e^{\mathcal{I}} \mid e \in V_e, (e^{\mathcal{I}})^I = \mathbf{true}\}$, by setting for each $a \in V_c$, $a^{CL} = (a^{\mathcal{I}})^C$; for each $r \in V_p$, $r^{OP} = (r^{\mathcal{I}})^P$; for each $d \in V_d$, $d^{DP} = (d^{\mathcal{I}})^D$; for each $t \in D$, $t^{DT} = (t^{\mathcal{I}})^T$; for each $i \in V_i$, $i^{IN} = (i^{\mathcal{I}})$; for each $l \in L_{QL}$, $l^{LT} = l^{\mathcal{I}}$. It is easy to see that γ is total, and $\gamma(\mathcal{I})$ is indeed a DS-model of \mathcal{O} .

Let us now show that γ is a surjective. Consider a DS-model $\mathcal{M} = \langle \Delta^o, \Delta^v, \cdot, CL, \cdot, OP, \cdot, DP, \cdot, DT, \cdot, IN, \cdot, LT \rangle$ of \mathcal{O} . We next show that there exists at least an MS-model $\mathcal{I} = \langle \mathcal{W}, \mathcal{I} \rangle$ of \mathcal{O} such that $\gamma(\mathcal{I}) = \mathcal{M}$. Let $\mathcal{W} = \langle \Delta^{o'}, \Delta^{v'}, \cdot, I, \cdot, C, \cdot, P, \cdot, D, \cdot, T \rangle$, where $\Delta^{o'} = \Delta^o$ and for every $l \in L_{QL}$, $l^{\mathcal{I}} = l^{LT}$. As

⁵<http://www.w3.org/TR/sparql11-entailment/#VarTyping>

⁶Observe that, while DS is the semantics adopted by current off-the-shelf OWL 2 QL reasoners, such as Mastro [31] or Ontop [32], it deviates from the DS entailment regime that is used for interpreting SPARQL queries over OWL 2 QL ontologies, in particular because the DS entailment regime handles existential quantifiers and union differently from First-Order Logic.

for $\Delta^{o'}$, we set it initially equal to Δ^o and progressively extend it as follows. For each entity e of the ontology, if $e \in V_i$, we set $e^{\mathcal{I}} = e^{IN}$ and $(e^{\mathcal{I}})^I = \mathbf{true}$. Otherwise, we extend $\Delta^{o'}$ with a new object o and set $e^{\mathcal{I}} = o$ and $o^I = \mathbf{false}$. Moreover, if $e \in V_c$, we set $(e^{\mathcal{I}})^C = e^{CL}$, otherwise $(e^{\mathcal{I}})^C$ is undefined. Similarly, if $e \in V_p$, $(e^{\mathcal{I}})^P = e^{OP}$, otherwise $(e^{\mathcal{I}})^{OP}$ is undefined; if $e \in V_d$, $(e^{\mathcal{I}})^D = e^{DP}$, otherwise $(e^{\mathcal{I}})^{DP}$ is undefined; if $e \in \mathbf{D}$, $(e^{\mathcal{I}})^T = e^{DT}$, otherwise $(e^{\mathcal{I}})^T$ is undefined. It is easy to see that \mathcal{I} is indeed an MS-model of \mathcal{O} such that $\gamma(\mathcal{I}) = \mathcal{M}$. \square

Based on the proposition above, we can also characterize the relationships between query entailment under MS and DS.

Proposition 11. *Let \mathcal{O} be an OWL 2 QL ontology, let α be an OWL 2 QL axiom, let Q be a union of boolean conjunctive metaground instance queries, and let q be a boolean conjunctive metaground query. Then,*

1. \mathcal{O} is satisfiable under MS if and only if it is satisfiable under DS.
2. $\mathcal{O} \models_{MS} \alpha$ if and only if $\mathcal{O} \models_{DS} \alpha$.
3. $\mathcal{O} \models_{MS} Q$ if and only if $\mathcal{O} \models_{DS} Q$.
4. $\mathcal{O} \models_{MS} q$ if and only if $\mathcal{O} \models_{DS} \text{int}(q)$ and $\mathcal{O} \models_{DS} \text{ext}(q)$, where $\text{ext}(q)$ and $\text{int}(q)$ denote the conjunctions of ABox and TBox atoms of q , respectively.

The above proposition shows that MS is a conservative extension of DS, and that we can rely on current off-the-shelf OWL 2 QL reasoners, for checking both satisfiability, logical implication, query entailment of metaground instance queries, and query entailment of metaground conjunctive queries.

2.5. Computational problems

In this paper, we focus on the computational complexity of two fundamental problems, namely satisfiability and query entailment. The former checks whether an ontology is satisfiable under MS, whereas the latter checks whether a query is entailed by an ontology under MS.

The formal definition of the two problems is as follows.

- *Satisfiability:* given as input an ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$, is there at least one MS-model for \mathcal{O} , i.e., is $\text{Mod}_{MS}(\mathcal{O}) \neq \emptyset$?

- *Query entailment*: given as input an ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ and a query Q , is Q entailed by \mathcal{O} under MS, i.e., $\mathcal{O} \models_{MS} Q$?

As for the computational complexity, we will measure it under different assumptions about which components of the input are fixed:

- *ABox complexity*: \mathcal{T} and Q are fixed,
- *ontology complexity*: Q is fixed,
- *combined complexity*: no component of the input is fixed.

As easy corollaries of the propositions illustrated in the previous subsection, we directly derive the following complexity results.

Corollary 12. *For an OWL 2 QL ontology, checking satisfiability under MS can be done in AC^0 w.r.t. ABox complexity, and in PTIME w.r.t. ontology complexity.*

Corollary 13. *If \mathcal{O} is an OWL 2 QL ontology and Q is a union of boolean conjunctive metaground instance queries, then checking whether $\mathcal{O} \models_{MS} Q$ can be done in AC^0 w.r.t. ABox complexity, in PTIME w.r.t. ontology complexity, and is NP-complete w.r.t. combined complexity.*

In the next sections, we implicitly refer to an OWL 2 QL ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ over vocabulary V and with signature Σ . Also, to simplify the notation, we use $\Sigma_c, \Sigma_p, \Sigma_d, \Sigma_t, \Sigma_i$, and Σ_{Lit} to refer to the components of Σ and Exp, Exp_c, Exp_p , and Exp_t to refer to the expressions that can be built over Σ . Also, without loss of generality, we assume that there are no two distinct literals l, l' in Σ_{Lit} such that $l^{LS} = l'^{LS}$. Clearly, by means of suitable substitutions, we can always reduce any ontology to an equivalent one that satisfies this condition.

Finally, when we talk about an “interpretation” $\mathcal{I} = \langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle$, we implicitly mean an “MS-interpretation”, with $\mathcal{W} = \langle \Delta^o, \Delta^v, \cdot^I, \cdot^C, \cdot^P, \cdot^D, \cdot^T \rangle$, when we talk about “model”, we implicitly mean “MS-model”, and when we use “ \models ”, we implicitly mean “ \models_{MS} ”.

3. The notion of canonical pseudo-model

In order to obtain complexity results for the computational problems introduced in the previous section, in this section we present a structure, called canonical pseudo-model, which enjoys the properties of representing all the models of the associated ontology. To this aim, we start by defining a chase procedure in our setting, and then we show how to exploit it for building the canonical pseudo-model of the ontology.

3.1. Chase procedure for OWL 2 QL under MS

The basic idea of the chase for our logic is similar to that of the chase for *DL-Lite_R* [5]: given \mathcal{O} , we build a (possibly infinite) set of axioms, starting from an initial structure $Chase^0(\mathcal{O})$, and then repeatedly computing $Chase^{j+1}(\mathcal{O})$ from $Chase^j(\mathcal{O})$ by applying suitable rules, for as long as they are applicable. In the process, we make use of two infinite ordered alphabets \mathcal{S}_o and \mathcal{S}_v of variables, both disjoint from Exp and Σ_{Lit} , for introducing new unknown individuals and new literals, when needed.

In the following, we use a naming scheme similar to that of the previous section, i.e., we use c, p, d, t, i , and l , possibly with subscripts, to denote classes in Exp_c , object properties in Exp_p , data properties in Σ_d , datatypes in Exp_t , individuals in Σ_i , and literals in Σ_{Lit} , respectively. Moreover, we use s and w to denote variables in \mathcal{S}_o and variables in \mathcal{S}_v , respectively. Finally, we use i^j (resp., l^j), possibly with subscripts, to denote individuals (resp., values) or variables occurring in $Chase^j(\mathcal{O})$, and thus belonging either to Σ_i (resp., Σ_{Lit}) or to \mathcal{S}_o^j (resp., \mathcal{S}_v^j), while we use s^{j+1} (resp., w^{j+1}) to denote variables in \mathcal{S}_o^{j+1} (resp., \mathcal{S}_v^{j+1}) that do not occur in $Chase^j(\mathcal{O})$.

It is worth noting that during the construction of the chase, we may introduce axioms that go beyond the syntax of OWL 2 QL, in particular having one of the following forms:

1. $t(w)$, where $t \in \Sigma_t$, and $w \in \mathcal{S}_v$,
2. $\exists p.c(i)$,
3. $\exists p^-.c(i)$,
4. $\delta(d).c(i)$,
5. $p^-(i_1, i_2)$.

Intuitively, we need such types of axiom in order both to trace the datatypes for the variables in \mathcal{S}_v , and to treat complex class expressions correctly.

We are now ready to illustrate the procedure for building $Chase(\mathcal{O})$, that is structured in two phases.

In the first phase, we initially set $Chase^0(\mathcal{O}) = \mathcal{O}$, and then we extend $Chase^0(\mathcal{O})$ by executing the following steps.

1. We add the following sets of obviously valid axioms:

$$\{\top_c(i) \mid i \in \Sigma_i\}, \{\top_p(i_1, i_2) \mid i_1, i_2 \in \Sigma_i\}, \{\top_d(i, l) \mid i \in \Sigma_i, l \in \Sigma_{Lit}\}.$$

This step ensures that \top_c , \top_p , and \top_d are treated in $Chase(\mathcal{O})$ coherently with their semantics.

2. We add the following sets of axioms:

- (a) $\{c(\alpha_c), \top_c(\alpha_c) \mid c \in Exp_c, \mathcal{O} \cup \{c \sqsubseteq_c \neg c\}$ is unsatisfiable, $\alpha_c \in \mathcal{S}_o^0\}$,
- (b) $\{p(\beta_p^1, \beta_p^2), \top_c(\beta_p^1), \top_c(\beta_p^2) \mid p \in Exp_p, \mathcal{O} \cup \{p \sqsubseteq_p \neg p\}$ is unsatisfiable, $\beta_p^1, \beta_p^2 \in \mathcal{S}_o^0\}$,
- (c) $\{d(\gamma_d, w_d), \top_c(\gamma_d), \mathbf{rdfs:Literal}(w_d) \mid d \in \Sigma_d, \gamma_d \in \mathcal{S}_o^0, w_d \in \mathcal{S}_v^0\}$
 $\mathcal{O} \cup \{d \sqsubseteq_d \neg d\}$ is unsatisfiable $\}$.

In other words, for every class expression c that can be built on the elements occurring in \mathcal{O} and is non-empty in every model of \mathcal{O} , we add axioms stating that a new unknown individual α_c , specific for c , belongs to c , so that α_c is the witness both of the non-emptiness of c , and of all non-subsumption relationships holding between c and the other classes. We proceed analogously for object property and data property expressions. The importance of this step will be discussed at the end of this section.

In the second phase, starting from the structure $Chase^0(\mathcal{O})$ built in the first phase, we carry out a sequence of steps, where in each step we compute $Chase^{j+1}(\mathcal{O})$ from $Chase^j(\mathcal{O})$ by applying suitable rules, called chase rules. We do so following a deterministic strategy that is *fair*, i.e., is such that if at some point a rule is applicable then it will be eventually applied. Note that at each step $j \geq 0$, $Chase^j(\mathcal{O})$ is a structure over $Exp \cup \mathcal{S}_o^j$ and $\Sigma_{Lit} \cup \mathcal{S}_v^j$, where $\mathcal{S}_o^j \subseteq \mathcal{S}_o$ and $\mathcal{S}_v^j \subseteq \mathcal{S}_v$.

We now turn our attention to the *chase rules* we use to compute $Chase^{j+1}(\mathcal{O})$ from $Chase^j(\mathcal{O})$, for each $j \geq 0$. The complete list of rules is specified in Table 4. For each rule, the first column specifies the rule number, the second column specifies the conditions under which it is applicable on $Chase^j(\mathcal{O})$, and the third column specifies the axioms to be added to $Chase^j(\mathcal{O})$ to compute $Chase^{j+1}(\mathcal{O})$. Consider for example Rule 1. It specifies that if $c_1(i^j)$ and $c_1 \sqsubseteq_c c_2$ are in $Chase^j(\mathcal{O})$, and $c_2(i^j)$ is not in $Chase^j(\mathcal{O})$, then $Chase^{j+1}(\mathcal{O})$ is obtained by extending $Chase^j(\mathcal{O})$ with the assertion $c_2(i^j)$. Note, in particular, that if c_2 has the form $\exists p.c$, then $Chase^{j+1}(\mathcal{O})$ is obtained by adding the assertion $\exists p.c(i^j)$, that is of one of the new types of axioms that we allow in $Chase(\mathcal{O})$.

Finally, we define $Chase(\mathcal{O})$ as $\bigcup_{j \in \mathbb{N}} Chase^j(\mathcal{O})$. Note that $Chase(\mathcal{O})$ is a possibly infinite set of axioms whose syntax is the usual OWL 2 QL syntax, except for the possible presence of variables in individual and literal positions, as well as axioms of the form specified at the beginning of this subsection.

Number	Applicability conditions	Axioms to be added
1	$c_1(i_1^j), c_1 \sqsubseteq_c c_2 \in Chase^j(\mathcal{O})$ $c_2(i_1^j) \notin Chase^j(\mathcal{O})$	$\{c_2(i_1^j)\}$
2	$p_1(i_1^j, i_2^j), p_1 \sqsubseteq_p p_2 \in Chase^j(\mathcal{O})$ $p_2(i_1^j, i_2^j) \notin Chase^j(\mathcal{O})$	$\{p_2(i_1^j, i_2^j)\}$
3	$d_1(i_1^j, l^j), d_1 \sqsubseteq_d d_2 \in Chase^j(\mathcal{O})$ $d_2(i_1^j, l^j) \notin Chase^j(\mathcal{O})$	$\{d_2(i_1^j, l^j)\}$
4	$d(i^j, l^j), \rho(d) \sqsubseteq_t t \in Chase^j(\mathcal{O})$ $t(l^j) \notin Chase^j(\mathcal{O})$	$\{t(l^j)\}$
5	$\text{Ref}(p) \in Chase^j(\mathcal{O})$, and there exists $i^j \in \Sigma_i \cup \mathcal{S}_o^j$ such that $p(i^j, i^j) \notin Chase^j(\mathcal{O})$	$\{p(i^j, i^j)\}$
6	$\exists p.c(i_1^j) \in Chase^j(\mathcal{O})$, and no i_2^j exists such that $c(i_2^j) \in Chase^j(\mathcal{O})$ and $p(i_1^j, i_2^j) \in Chase^j(\mathcal{O})$	$\{c(s^{j+1}), p(i_1^j, s^{j+1})\}$
7	$p(i_1^j, i_2^j), c(i_2^j) \in Chase^j(\mathcal{O})$ $\exists p.c(i_1^j) \notin Chase^j(\mathcal{O})$	$\{\exists p.c(i_1^j)\}$
8	$\delta(d).t(i^j) \in Chase^j(\mathcal{O})$, and no l^j exists such that $t(l^j) \in Chase^j(\mathcal{O})$ and $d(i^j, l^j) \in Chase^j(\mathcal{O})$	$\{t(w^{j+1}), d(i^j, w^{j+1})\}$
9	$d(i^j, l^j), t(l^j) \in Chase^j(\mathcal{O})$ $\delta(d).t(i^j) \notin Chase^j(\mathcal{O})$	$\{\delta(d).t(i^j)\}$
10	$p^-(i_1^j, i_2^j) \in Chase^j(\mathcal{O})$ $p(i_2^j, i_1^j) \notin Chase^j(\mathcal{O})$	$\{p(i_2^j, i_1^j)\}$
11	$p(i_1^j, i_2^j) \in Chase^j(\mathcal{O})$ $p^-(i_2^j, i_1^j) \notin Chase^j(\mathcal{O})$	$\{p^-(i_2^j, i_1^j)\}$

Table 4: Chase rules for OWL 2 QL ontologies

3.2. The canonical pseudo-model

We now show how, based on $Chase(\mathcal{O})$, we can build a semantic structure, called canonical pseudo-model, that enjoys specific properties useful for query entailment under MS.

Definition 14. The *canonical pseudo-model* $Can(\mathcal{O})$ for \mathcal{O} (under MS) is defined as $\langle \bar{\mathcal{W}}, \bar{\mathcal{I}} \rangle$, where:

- $\bar{\mathcal{W}} = \langle \bar{\Delta}^o, \bar{\Delta}^v, \bar{I}, \bar{C}, \bar{P}, \bar{D}, \bar{T} \rangle$ has the form of an interpretation structure for \mathcal{O} under MS, and is defined as follows:
 - $\bar{\Delta}^o = Exp \cup \mathcal{S}_o$;
 - $\bar{\Delta}^v = \Delta^v \cup \mathcal{S}_v$;

- for every $e \in \bar{\Delta}^o$, if e occurs in individual position in $Chase(\mathcal{O})$, then $e^{\bar{I}} = \mathbf{true}$; otherwise, $e^{\bar{I}} = \mathbf{false}$;
 - for every $e \in \bar{\Delta}^o$, if $e \notin Exp_c$, then $\cdot^{\bar{C}}$ is undefined for e , otherwise $e^{\bar{C}}$ is defined for e , and is such that $e^{\bar{C}} = \{i \mid e(i) \in Chase(\mathcal{O})\}$;
 - for every $e \in \bar{\Delta}^o$, if $e \notin Exp_p$, then $\cdot^{\bar{P}}$ is undefined for e , otherwise $e^{\bar{P}}$ is defined for e , and is such that $e^{\bar{P}} = \{(i_1, i_2) \mid e(i_1, i_2) \in Chase(\mathcal{O})\}$;
 - for every $e \in \bar{\Delta}^o$, if $e \notin \Sigma_d$, then $\cdot^{\bar{D}}$ is undefined for e , otherwise $e^{\bar{D}}$ is defined for e , and is such that $e^{\bar{D}} = \{(i, v) \mid v \in \mathcal{S}_v, e(i, v) \in Chase(\mathcal{O})\} \cup \{(i, l^{LS}) \mid e(i, l) \in Chase(\mathcal{O}), l \in \Sigma_{Lit}\}$;
 - for every $e \in \bar{\Delta}^o$, if $e \notin \Sigma_t$, then $\cdot^{\bar{T}}$ is undefined for e , otherwise $e^{\bar{T}}$ is defined for e and is such that if $e \in \mathbf{D}$, then $e^{\bar{T}} = e^{DT} \cup \{s \mid e(s) \in Chase(\mathcal{O}), s \in \mathcal{S}_v\}$, otherwise e has the form $\rho(d)$ and $e^{\bar{T}} = \{v \mid (o, v) \in d^{\bar{D}}\}$.
- $\cdot^{\bar{I}}$ has the form of an interpretation function of \mathcal{O} under MS, and is defined as follows:
 - for every $e \in Exp$, $e^{\bar{I}} = e$, and for every $s \in \mathcal{S}_o$, $s^{\bar{I}} = s$;
 - for every $l \in \Sigma_{Lit}$, $l^{\bar{I}} = l^{LS}$, and for every $s \in \mathcal{S}_v$, $s^{\bar{I}} = s$. □

We end this section with a list of observations on the properties of $Can(\mathcal{O})$.

- $Can(\mathcal{O})$ is called “pseudo-model” (and not “model”) because it does not strictly conform to the definition of interpretation of \mathcal{O} under MS. Indeed, it deviates from such definition for two reasons: (1) $t^{\bar{T}}$ may not coincide with t^{DT} , because it may include variables in \mathcal{S}_v . (2) $\perp_c^{\bar{C}}$, $\perp_p^{\bar{P}}$, or $\perp_d^{\bar{D}}$ may not coincide with the empty set (note that, in all these cases, \mathcal{O} would be unsatisfiable). However, it is not difficult to see that, besides these properties, $Can(\mathcal{O})$ enjoys all other properties for being an interpretation of \mathcal{O} .
- By the same arguments used in [5], it can be verified that $Can(\mathcal{O})$ satisfies all axioms of \mathcal{O} , and it does so in a sort of minimal way. Indeed, we call it “canonical” just because, as we will show later, as far as instance relationships are concerned, it represents all the models of \mathcal{O} . Also, for every $c \in Exp_c$ that is non-empty in every model of \mathcal{O} (i.e., for which $\mathcal{O} \cup \{c \sqsubseteq_c \neg c\}$ is unsatisfiable), α_c is an instance in $Can(\mathcal{O})$ of exactly the class expressions subsuming c in \mathcal{O} and therefore, as we will see in Section 6, it is used to falsify all the subclass relationships that are not logically implied by \mathcal{O} . A similar property holds for object and data property expressions. In particular, for every $p \in Exp_p$ that is non-empty, (β_p^1, β_p^2) is an instance of exactly the object property expressions subsuming p in \mathcal{O} .

and (β_p^1, β_p^1) is an instance of p if and only if p is reflexive. It follows that $Can(\mathcal{O})$ is representative of all models of \mathcal{O} with respect to all positive axioms: those that hold in $Can(\mathcal{O})$ are exactly those that are logically implied by \mathcal{O} .

- For every variable s in \mathcal{S}_o , if s occurs in $Chase(\mathcal{O})$, then it occurs in individual position. Hence, by definition of $Can(\mathcal{O})$, $s^{\bar{I}} = \mathbf{true}$, i.e., s is an individual in $Can(\mathcal{O})$. On the other hand, the functions $\cdot^{\bar{C}}$, $\cdot^{\bar{P}}$, $\cdot^{\bar{D}}$, and $\cdot^{\bar{T}}$ are all undefined for s . This fact reflects the intuition that $Can(\mathcal{O})$ represents the minimal knowledge satisfying \mathcal{O} : indeed, since s is one of the objects introduced in $Can(\mathcal{O})$ by the chase, considering it as a class, an object property or a data property would represent an arbitrary assumption, that is not valid in all models. This confirms the importance of allowing the above functions to be partial in MS, contrarily to many semantics defined for languages supporting metamodeling, such as, for example, those used in [19, 13]. Indeed, as a consequence of the fact that under these semantics the above functions are total, for every newly introduced unknown individual s in the chase, we would have two possibilities in building in $Can(\mathcal{O})$: (i) either we would sanction that the set of instances associated to s seen as a class is empty, in which case s would represent a subset of every class in the ontology, or (ii) we would make the extension of s containing at least one instance. Both cases would make $Can(\mathcal{O})$ not general enough to represent all models of the ontology.
- Finally, for every element e in $\bar{\Delta}^o$, if the function $\cdot^{\bar{C}}$ (resp., $\cdot^{\bar{P}}$, $\cdot^{\bar{D}}$, $\cdot^{\bar{T}}$) is defined for e , then e is in Exp_c (resp., Exp_p , Σ_d , Exp_t).

4. Entailment of instance queries

In this section we deal with entailment of instance queries in OWL 2 QL under MS. In particular, we first show that the canonical pseudo-model $Can(\mathcal{O})$ of an ontology \mathcal{O} under MS represents the set of all models of \mathcal{O} with respect to instance queries, and then we present an algorithm based on such property.

In order to exploit the role of $Can(\mathcal{O})$ with respect to instance queries, we start by defining the notion of instance-based homomorphism from $Can(\mathcal{O})$ to an interpretation \mathcal{I} of \mathcal{O} .

Definition 15. Let $\mathcal{I} = \langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle$ be an interpretation of \mathcal{O} . An *instance-based homomorphism* from $Can(\mathcal{O})$ to \mathcal{I} is a function Ψ from $\bar{\Delta}^o$ to Δ^o and from $\bar{\Delta}^v$ to Δ^v such that:

- $\Psi(e) = e^{\mathcal{I}}$ for every $e \in Exp$,

- $\Psi(v) = v$ for every $v \in \Delta^v$, and
- Ψ preserves in \mathcal{W} the following *instance-based* properties of $Can(\mathcal{O})$:
 1. for every $e_1, e_2 \in \bar{\Delta}^o$ such that $e_2 \in e_1^{\bar{C}}$, $\Psi(e_2) \in \Psi(e_1)^{\bar{C}}$;
 2. for every $e_1, e_2, e_3 \in \bar{\Delta}^o$ such that $\langle e_2, e_3 \rangle \in e_1^{\bar{P}}$, $\langle \Psi(e_2), \Psi(e_3) \rangle \in \Psi(e_1)^{\bar{P}}$;
 3. for every $e_1, e_2 \in \bar{\Delta}^o$ and every $v \in \bar{\Delta}^v$ such that $\langle e_2, v \rangle \in e_1^{\bar{D}}$, $\langle \Psi(e_2), \Psi(v) \rangle \in \Psi(e_1)^{\bar{D}}$;
 4. for every $e \in \bar{\Delta}^o$ and every $v \in \bar{\Delta}^v$ such that $v \in e^{\bar{T}}$, $\Psi(v) \in \Psi(e)^{\bar{T}}$.

□

The following proposition shows that $Can(\mathcal{O})$ enjoys a notable property, namely it has an instance-based homomorphism to every model of \mathcal{O} , which essentially means that what $Can(\mathcal{O})$ sanctions about the objects and the values in the ontology is true in every model of the ontology. The proof of the proposition uses the notion of k -portion of $Can(\mathcal{O})$ that we now introduce. Just as the canonical pseudo-model $Can(\mathcal{O})$ of \mathcal{O} is based on $Chase(\mathcal{O}) = \bigcup_{j \in \mathbb{N}} Chase^j(\mathcal{O})$, the k -portion of $Can(\mathcal{O})$, denoted $Can_k(\mathcal{O})$, is based on $Chase_k(\mathcal{O}) = \bigcup_{j \leq 0 \leq k} Chase^j(\mathcal{O})$, in the sense that its definition is simply derived from definition 14 by substituting $Chase(\mathcal{O})$ with $Chase_k(\mathcal{O})$, thus obtaining $\mathcal{W}_k = \langle \bar{\Delta}_k^o, \bar{\Delta}_k^v, \bar{I}_k, \bar{C}_k, \bar{P}_k, \bar{D}_k, \bar{T}_k \rangle$, $\bar{\Delta}_k^o = Exp \cup (\bigcup_{0 \leq i \leq k} \mathcal{S}_o^i)$, and $\bar{\Delta}_k^v = \Delta^v \cup (\bigcup_{0 \leq i \leq k} \mathcal{S}_v^i)$, where $\mathcal{S}_o^i \cup \mathcal{S}_v^i$ is the set of variables introduced when computing $Chase^i(\mathcal{O})$.

Proposition 16. *For every $\mathcal{I} \in Mod_{MS}(\mathcal{O})$, there exists an instance-based homomorphism from $Can(\mathcal{O})$ to \mathcal{I} .*

Proof. Let $\mathcal{I} = \langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle \in Mod_{MS}(\mathcal{O})$ be a model of \mathcal{O} . We define a function Ψ from $\bar{\Delta}^o$ to Δ^o and from $\bar{\Delta}^v$ to Δ^v , and we show that Ψ is an instance-based homomorphism from $Can(\mathcal{O})$ to \mathcal{I} . Specifically, we define Ψ by induction on k , where k is the number of rule applications which leads to $Chase_k(\mathcal{O})$, by setting Ψ_k as a function on $\bar{\Delta}_k^o \cup \bar{\Delta}_k^v$, and letting $\Psi = \bigcup_{j \in \mathbb{N}} \Psi_j$. Simultaneously, we show, by induction on k , that Ψ_k is an instance-based homomorphism from $Can_k(\mathcal{O})$ to \mathcal{I} .

Base step. Let $k = 0$. First of all, we set $\Psi_0(e) = e^{\mathcal{I}}$ for every $e \in Exp$, and $\Psi_0(v) = v$, for every $v \in \Delta^v$. As for every element that is not in $Exp \cup \Delta^v$, and therefore is in $\mathcal{S}_o^0 \cup \mathcal{S}_v^0$, we observe that it appears exactly once in $Chase_0(\mathcal{O})$. Based on this observation, for every $s_1, s_2 \in \mathcal{S}_o^0$ and for every $w \in \mathcal{S}_v^0$, we set Ψ_0 as follows:

- if $s_1 = \alpha_c$, then $\Psi_0(s_1) = o$, where o is any object in $(c^{\mathcal{I}})^C$. Note that at least one such object exists in Δ^o , since by construction $\mathcal{O} \cup \{c \sqsubseteq_c \neg c\}$ is unsatisfiable;
- if $s_1 = \beta_p^1$ and $s_2 = \beta_p^2$, then $\Psi_0(s_1) = o_1$, and $\Psi_0(s_2) = o_2$, where o_1, o_2 is any pair of objects such that $\langle o_1, o_2 \rangle \in (p^{\mathcal{I}})^P$. Note that such pair must exist, since by construction $\mathcal{O} \cup \{p \sqsubseteq_p \neg p\}$ is unsatisfiable;
- if $s_1 = \gamma_d$ and $w = w_d$, then $\Psi(s_1) = o$ and $\Psi_0(w) = v$, where o, v is any pair such that $\langle o, v \rangle \in (d^{\mathcal{I}})^D$. Note that such pair must exist since by construction $\mathcal{O} \cup \{d \sqsubseteq_d \neg d\}$ is unsatisfiable;

It remains to prove that Ψ_0 is an instance-based homomorphism from $Can_0(\mathcal{O})$ to \mathcal{I} , by showing that it satisfies all properties of Definition 15. We next prove that it satisfies Property (1) and refer to Appendix A for Properties (2), (3), and (4).

Let e_1, e_2 be two objects in $\bar{\Delta}_0^o$ such that $e_2 \in e_1^{\bar{C}_0}$. We want to show that $\Psi_0(e_2) \in \Psi_0(e_1)^C$. Since $e_2 \in e_1^{\bar{C}_0}$, by construction of \bar{W} , we have $e_1(e_2) \in Chase_0(\mathcal{O})$, where e_1 is a class expression belonging to Exp_c . Thus, by definition, $\Psi_0(e_1) = e_1^{\mathcal{I}}$. Also, by construction of $Chase_0(\mathcal{O})$, $e_1(e_2)$ either belongs to \mathcal{O} or it was added in one of the steps 1, 2(a), 2(b), 2(c), or 3 of its construction.

- Suppose that $e_1(e_2) \in \mathcal{O}$. Then, $e_2 \in \Sigma_i$ and therefore $e_2 \in Exp$, and hence $\Psi_0(e_2) = e_2^{\mathcal{I}}$. But then, since \mathcal{I} is a model of \mathcal{O} , $e_2^{\mathcal{I}} \in (e_1^{\mathcal{I}})^C$ and therefore $\Psi_0(e_2) \in \Psi_0(e_1)^C$.
- Suppose that $e_1(e_2)$ was added in step 1. Then $e_1 = \top_c$ and $e_2 \in \Sigma_i$. It follows that $\Psi_0(e_2) = e_2^{\mathcal{I}}$. Also, since \mathcal{I} is a model of \mathcal{O} , $(\top_c^{\mathcal{I}})^C = \{o \mid o^{\mathcal{I}} = \text{true}\}$ and $(e_2^{\mathcal{I}})^{\mathcal{I}} = \text{true}$ (because $e_2 \in \Sigma_i$). Therefore, $\Psi_0(e_2) \in \Psi_0(e_1)^C$.
- Suppose that $e_1(e_2)$ was added in step 2(a). Then, $e_2 = \alpha_c$, where $c \in Exp_c$, $\mathcal{O} \cup \{c \sqsubseteq_c \neg c\}$ is unsatisfiable, $\alpha_c \in \mathcal{S}_o^0$, and either $e_1 = c$ or $e_1 = \top_c$. In both cases, by definition of Ψ_0 , $\Psi_0(e_2) = o$ where $o \in (c^{\mathcal{I}})^C$ and $o^{\mathcal{I}} = \text{true}$. Hence, in both cases, $\Psi_0(e_2) \in \Psi_0(e_1)^C$.
- Suppose that $e_1(e_2)$ was added in step 2(b). Then, $e_1 = \top_c$ and either $e_2 = \beta_p^1$ or $e_2 = \beta_p^2$, where $p \in Exp_p$, $\mathcal{O} \cup \{p \sqsubseteq_p \neg p\}$ is unsatisfiable, and $\beta_p^1, \beta_p^2 \in \mathcal{S}_o^0$. Suppose that $e_2 = \beta_p^1$ (resp., $e_2 = \beta_p^2$), then by definition of Ψ_0 , $\Psi_0(e_2) = o_1$ where $\langle o_1, o_2 \rangle \in (p^{\mathcal{I}})^P$ (resp. $\langle o_2, o_1 \rangle \in (p^{\mathcal{I}})^P$), for some $o_2 \in \Delta_o$, and $o_1^{\mathcal{I}} = \text{true}$. Hence, $\Psi_0(e_2) \in \Psi_0(e_1)^C$.
- By following the same line of reasoning of the previous case, we can prove that if $e_1(e_2)$ was added in step 2(c) or step 3, then $\Psi_0(e_2) \in \Psi_0(e_1)^C$.

Inductive step. Let k be the number of rule applications which leads to $Chase_k(\mathcal{O})$ and suppose that Ψ_k is an instance-based homomorphism from $Can_k(\mathcal{O})$ to \mathcal{I} . Then, suppose that a new chase rule is applied to get $Chase_{k+1}(\mathcal{O})$. We define $\Psi_{k+1} = \Psi_k \cup \Psi'$, where Ψ' is the part of Ψ_{k+1} taking care of the new variables possibly introduced in $\mathcal{S}_o^{k+1} \cup \mathcal{S}_v^{k+1}$. We have to prove that Ψ_{k+1} is an instance-based homomorphism from $Can_{k+1}(\mathcal{O})$ to \mathcal{I} . Since this proof is rather “classical” and very similar to the one used in [5] for *DL-Lite* ontologies, in the following, we do not consider all possible rule applications (which one can find in the complete proof in Appendix A), but rather illustrate two example cases of rule applications, such that the first one does not introduce any new variable and the second one does.

Suppose that the rule applied by the $(k+1)$ -th rule application is **(1)**. In particular, suppose that $c_1(i_1) \in Chase_k(\mathcal{O})$, $c_1 \sqsubseteq_c c_2 \in Chase_k(\mathcal{O})$, and $c_2(i_1) \notin Chase_k(\mathcal{O})$. Then, by applying rule **(1)**, $c_2(i_1)$ is introduced in $Chase_{k+1}(\mathcal{O})$, and hence the only difference between Can_k and Can_{k+1} is that $i_1 \notin c_2^{\bar{C}_k}$, whereas $i_1 \in c_2^{\bar{C}_{k+1}}$. Let us now consider Ψ_{k+1} . Since no new variable is introduced, $\Psi_{k+1} = \Psi_k$. Hence, on one hand, since $c_1(i_1) \in Chase_k(\mathcal{O})$ and since, by inductive hypothesis, Ψ_{k+1} is an instance-based homomorphism from Can_k to \mathcal{I} , $\Psi_{k+1}(i_1) \in (\Psi_{k+1}(c_1))^C$. On the other hand, since \mathcal{I} is a model of \mathcal{O} , we know that $\mathcal{I} \models c_1 \sqsubseteq_c c_2$, implying that $\Psi_{k+1}(i_1) \in (\Psi_{k+1}(c_2))^C$, and, hence, that Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

Suppose that the rule applied by the $(k+1)$ -th rule application is **(6)**. In particular, suppose that $\exists p.c(i_1) \in Chase_k(\mathcal{O})$, and that no i exists such that $c(i) \in Chase_k(\mathcal{O})$ and $p(i_1, i) \in Chase_k(\mathcal{O})$. Then, by applying rule **(6)**, $c(s)$ and $p(i_1, s)$ are introduced in $Chase_{k+1}(\mathcal{O})$, where s is a new object variable introduced in $Chase_{k+1}(\mathcal{O})$ such that $\mathcal{S}_o^{k+1} = \mathcal{S}_o^k \cup \{s\}$ and $\mathcal{S}_v^{k+1} = \mathcal{S}_v^k$. Let us now consider Ψ_{k+1} . We define it by extending Ψ_k , choosing an appropriate object for $\Psi_{k+1}(s)$, and we have to prove that all the instance-based properties of Ψ_{k+1} are preserved from Can_{k+1} to \mathcal{I} . Since $\exists p.c(i_1) \in Chase_k(\mathcal{O})$, we know that $i_1 \in (\exists p.c)^{\bar{C}_k}$ and, by inductive hypothesis, $\Psi_k(i_1) \in (\Psi_k(\exists p.c))^C$. Moreover, since \mathcal{I} is a model of \mathcal{O} , there exists at least one object o in Δ^o such that:

$$\langle \Psi_k(i_1), o \rangle \in (p^{\mathcal{I}})^P \text{ and } o \in (c^{\mathcal{I}})^C \quad (*).$$

Choose one such o , and set $\Psi_{k+1}(s) = o$. Since $c(s) \in Chase_{k+1}(\mathcal{O})$, we have $s \in c^{\bar{C}_{k+1}}$, and since $p(i_1, s) \in Chase_{k+1}(\mathcal{O})$, we have $\langle i_1, s \rangle \in p^{\bar{P}_{k+1}}$. But then, from $\Psi_{k+1}(s) = o$, $\Psi_{k+1}(c) = c^{\mathcal{I}}$, $\Psi_{k+1}(p) = p^{\mathcal{I}}$, and $(*)$, it immediately follows $\Psi_{k+1}(s) \in (\Psi_{k+1}(c))^C$ and $\langle \Psi_{k+1}(i_1), \Psi_{k+1}(s) \rangle \in (\Psi_{k+1}(p))^P$, which is enough to prove that Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} . \square

Based on the above proposition, we next show that the canonical pseudo-

model $Can(\mathcal{O})$ is representative of all the models of \mathcal{O} under MS with respect to instance queries.

Proposition 17. *If \mathcal{O} is satisfiable, and Q is an instance query over \mathcal{O} , then $\mathcal{O} \models Q$ if and only if there exists a query homomorphism from Q to $Can(\mathcal{O})$.*

Proof. If-part. Suppose that there exists a query homomorphism f from $Q = q_1 \cup \dots \cup q_n$ to $Can(\mathcal{O})$. This means that there exists a disjunct q_i of Q , for $i \in \{1, \dots, n\}$, that is an instance query such that f is a query homomorphism from q_i to $Can(\mathcal{O})$. By Proposition 16, we know that for every model \mathcal{I} of \mathcal{O} , there exists an instance-based homomorphism Ψ from $Can(\mathcal{O})$ to \mathcal{I} . Therefore, if we compose the functions f and Ψ , we obtain a query homomorphism from q_i to \mathcal{I} , which implies that $\mathcal{O} \models Q$.

Only-if-part. Suppose that there is no query homomorphism from $Q = q_1 \cup \dots \cup q_n$ to $Can(\mathcal{O})$. We show how to derive from $Can(\mathcal{O})$ an MS-model of \mathcal{O} where Q is false. Since \mathcal{O} is satisfiable, the only reason why $Can(\mathcal{O})$ is not directly an MS-model of \mathcal{O} is because $\bar{\Delta}^v$ is equal to $\Delta^v \cup \mathcal{S}_v$, instead of simply Δ^v , and values in the extension of data properties may belong to \mathcal{S}_v rather than Δ^v . We now define an interpretation \mathcal{I}^c from $Can(\mathcal{O})$ as follows. First, observe that for every $w \in \mathcal{S}_v$ appearing in $Chase(\mathcal{O})$, a set of axioms of the form $t(w)$, with $t \in \mathbb{D}$, appears also in $Chase(\mathcal{O})$. Therefore, for every $w \in \mathcal{S}_v$, we can define the set $D_w = \{t \mid w \in t^{\bar{T}}\}$, and then the assignment $\eta_v : \mathcal{S}_v \rightarrow \Delta^v$ such that for every $w \in \mathcal{S}_v$:

- $\eta_v(w) \in \bigcap_{t \in D_w} t^{DT}$,
- $\eta_v(w) \notin \{l^{LS} \mid l \in \Sigma_{Lit}\}$, and
- $\forall w' \in \mathcal{S}_v, w' \neq w, \eta_v(w') \neq \eta_v(w)$.

Note that, as we already recalled in Section 2, the OWL 2 QL datatype map is defined in such a way that the intersection of the value spaces of any set of admitted datatypes is either empty or infinite. Hence, $\bigcap_{t \in D_w} t^{DT}$ is infinite, which ensures that such a η_v always exists. Now, let the interpretation \mathcal{I}^c be obtained from $Can(\mathcal{O})$ by substituting every $w \in \mathcal{S}_v$ with $\eta_v(w)$. It is easy to see that \mathcal{I}^c is an MS-model of \mathcal{O} . Now, suppose that Q is true in \mathcal{I}^c . This implies that there is a conjunct q_i such that there is a query homomorphism f from q_i to \mathcal{I}^c , for $i \in \{1, \dots, n\}$. Let f' be the function obtained from f by setting $f'(v) = w$, for every v such that there exists w in \mathcal{S}_v such that $\eta_v(w) = v$. It is immediate to verify that f' is a query homomorphism from q_i to $Can(\mathcal{O})$, thus contradicting the hypothesis that there is no query homomorphism from Q to $Can(\mathcal{O})$. So, we conclude that Q is false in \mathcal{I}^c , and therefore $\mathcal{O} \not\models Q$. \square

We are now ready to present our technique for checking entailment of instance queries. The technique, called *query entailment through metagrounding*, is essentially the one proposed in [15] for *DL-Lite*, and relies on the notion of instantiation of a conjunctive query, that we recall as follows. Given a conjunctive query q , an n -tuple $\vec{x} = (x_1, x_2, \dots, x_n)$ of variables of q , an n -tuple $\vec{a} = (a_1, a_2, \dots, a_n)$ of expressions in Exp , and a substitution $\mu : \vec{x} \leftarrow \vec{a}$, we denote by $\mu(q)$ the query, called *μ -instantiation* (or simply *instantiation*) of q , resulting from the following steps:

1. we apply the substitution μ to q , thus replacing each x_i in \vec{x} with a_i in \vec{a} , for $i \in \{1, \dots, n\}$, and obtaining the query q' ;
2. we replace the atoms in q' that are not legal for OWL2QL with equivalent legal atoms involving new variables; for example, an atom of the form $\exists p.c(i)$ in the query is replaced with the conjunction of the atoms $p(i, z)$ and $c(z)$, where z is a new variable.

Definition 18. Let q be a conjunctive query over \mathcal{O} . A *metagrounding* of q with respect to \mathcal{O} is a μ -instantiation of q , for some $\mu : \vec{x} \leftarrow \vec{a}$, such that $\vec{x} = (x_1, x_2, \dots, x_n)$ are all the metavariables occurring in q , and for every $i \in \{1, \dots, n\}$, if x_i occurs in class (resp. object property, data property, or datatype) position in q , then a_i is any expression in Exp_c (resp. Exp_p , Σ_d , or Exp_t). \square

Note that, by the above definition, if a variable x_i occurs in more than one type of position in q , then the expression substituted for x_i must be taken from the intersection of more than one set of expressions. For example if x_i occurs both in class and object property position in q , then the expression substituting x_i are taken from the set $Exp_c \cap Exp_p$.

If q is a conjunctive query, then we denote by $MG(q, \mathcal{O})$ the metaground query that is the union of all metagroundings of q with respect to \mathcal{O} . If Q is a union of conjunctive queries, then we denote by $MG(Q, \mathcal{O})$ the metaground query that is the union of all metagroundings of every disjunct in Q with respect to \mathcal{O} .

The following proposition shows that, based on the notion of metagrounding, we can reduce the problem of checking the existence of a query homomorphism from an instance query to $Can(\mathcal{O})$ to the problem of checking the existence of a query homomorphism from a metaground instance query to $Can(\mathcal{O})$.

Proposition 19. *If Q is an instance query over \mathcal{O} , then there exists a query homomorphism from Q to $Can(\mathcal{O})$ if and only if there exists a query homomorphism from $MG(Q, \mathcal{O})$ to $Can(\mathcal{O})$.*

Proof. Let $MG(Q, \mathcal{O}) = q_1 \cup \dots \cup q_m$, where every q_i , for $i \in \{1, \dots, m\}$, is a conjunctive metaground instance query.

If-part. Suppose that there exists a query homomorphism f from $MG(Q, \mathcal{O})$ to $Can(\mathcal{O})$. Then, f is a query homomorphism from a disjunct q_j of $MG(Q, \mathcal{O})$ to $Can(\mathcal{O})$, for some $j \in \{1, \dots, m\}$. By definition of metagrounding of an union of conjunctive queries, there exists a disjunct q' of Q such that $q_j \in MG(q', \mathcal{O})$. Let $\{x_1, \dots, x_n\}$ be the metavariables of q' , and μ be the substitution $\mu : (x_1, \dots, x_n) \leftarrow (a_1, \dots, a_n)$ such that $\mu(q') = q_j$. If we define f' as the extension of f such that $f'(x_i) = a_i$, for $i \in \{1, \dots, n\}$, we obtain a query homomorphism from q' to $Can(\mathcal{O})$, which, by definition, is a query homomorphism from Q to $Can(\mathcal{O})$.

Only-if-part. Suppose that there exists a query homomorphism f from Q to $Can(\mathcal{O})$. Then, by definition, f is a query homomorphism from a disjunct q' of Q to $Can(\mathcal{O})$. Let $\{x_1, \dots, x_n\}$ be the metavariables of q' . If x_i , for $i \in \{1, \dots, n\}$, occurs in class (resp., object property, data property, or datatype) position in q' , then by definition of query homomorphism, the function $\cdot^{\bar{C}}$ (resp., $\cdot^{\bar{P}}$, $\cdot^{\bar{D}}$, or $\cdot^{\bar{T}}$) must be defined for $f(x_i)$. Then, by construction of $Can(\mathcal{O})$, we have that $f(x_i)$ must be in Exp_c (resp., Exp_p , Σ_d , Exp_t). Let μ be the substitution $\mu : (x_1, \dots, x_n) \leftarrow (f(x_1), \dots, f(x_n))$. Clearly $\mu(q')$ is a metagrounding of q' w.r.t. \mathcal{O} . Then there exists a disjunct q_j of $MG(Q, \mathcal{O})$, for $j \in \{1, \dots, m\}$, such that $q_j = \mu(q')$. Therefore, if we define f' as the restriction of f to the variables occurring only in individual or value position in q' , we obtain a query homomorphism from q_j to $Can(\mathcal{O})$, and, by definition, from $MG(Q, \mathcal{O})$ to $Can(\mathcal{O})$. \square

We stress the importance of keeping undefined the partial functions $\cdot^{\bar{C}}$, $\cdot^{\bar{P}}$, $\cdot^{\bar{D}}$, and $\cdot^{\bar{T}}$ for variables in \mathcal{S}_ρ . Indeed, this is the property that we have exploited in the proof in order to conclude that if the function $\cdot^{\bar{C}}$ (resp., $\cdot^{\bar{P}}$, $\cdot^{\bar{D}}$, or $\cdot^{\bar{T}}$) is defined for $f(x_i)$, then, by construction of $Can(\mathcal{O})$, we have that $f(x_i)$ is in Exp_c (resp., Exp_p , Σ_d , Exp_t).

By combining Proposition 17 and Proposition 19, we obtain the following theorem.

Theorem 20. *If Q is an instance query over \mathcal{O} , then $\mathcal{O} \models Q$ if and only if $\mathcal{O} \models MG(Q, \mathcal{O})$.*

Observe that each atom in a metagrounding of a general query Q is either a ground TBox atom, or an ABox atom without metavariables. Thus, in particular, if Q is an instance query, a metagrounding of Q is a metaground instance query. Based on this observation, the above theorem, combined with proposition 11, directly provides us with an algorithm for checking instance queries

entailment over OWL 2 QL ontologies under MS, that exploits current off-the-shelf OWL 2 QL reasoners. The algorithm simply nondeterministically guesses a metagrounding Q' of Q , and then checks whether $\mathcal{O} \models_{DS} Q'$ by resorting to an existing OWL 2 QL reasoner. Based on the fact that all the symbols used to compute Exp_c , Exp_p , Σ_d , and Exp_t in order to form the metagrounding Q' are in the TBox (see the definition of OWL 2 QL ontology in Section 2.1), it is easy to see that this leads to the following complexity results.

Theorem 21. *Entailment of instance queries is in AC^0 w.r.t. ABox complexity, PTIME w.r.t. ontology complexity, and NP-complete w.r.t. combined complexity.*

In other words, in OWL 2 QL, entailment of instance queries has exactly the same complexity as entailment of standard first-order unions of conjunctive queries under DS.

5. Entailment of general queries: lower bounds

We start the section by addressing entailment of general queries under MS. While, as shown in the previous section, entailment of instance queries in OWL 2 QL can be checked by resorting to the metagrounding technique, we argue that entailment of general queries is more intricate. To illustrate why, we start by presenting an example, and then we investigate the complexity lower-bounds of the problem.

Example 22. Let \mathcal{O} be the ontology formed by the TBox axioms $\{A \sqsubseteq_c \neg C, A \sqsubseteq_c \neg \exists R, A \sqsubseteq_c \neg \exists R^-\}$, and the ABox axioms $\{B(F), C(F), R(F, F), R(C, A), R(B, C)\}$, and let q be the following general query:

$$q \leftarrow B(y) \wedge z(y) \wedge A \sqsubseteq_c \neg x \wedge R(x, z)$$

Note that \mathcal{O} is clearly satisfiable, and that the only metavariables occurring in q are x and z . One can easily verify that there exists no substitution $\mu : (x, z) \leftarrow (c_1, c_2)$ such that $c_1, c_2 \in Exp_c$, and $\mathcal{O} \models \mu(q)$. Thus, if we rely only on metagrounding for checking query entailment, we conclude that q is not entailed by \mathcal{O} . However, let us partition the set of models of \mathcal{O} into the two sets M_1 and M_2 , where M_1 is the set of models in which A and B are interpreted as non-disjoint classes, and M_2 is the set of models in which A and B are interpreted as disjoint classes, and let us consider the following substitutions:

$$\begin{aligned} \mu_1 &: (x, z) \leftarrow (C, A) \\ \mu_2 &: (x, z) \leftarrow (B, C) \end{aligned}$$

Since entities A , B , and C are all in Exp_c , $\mu_1(q)$ and $\mu_2(q)$ are two metagroundings of q w.r.t. \mathcal{O} . Clearly, $\mu_1(q)$ is true in every model in M_1 , while $\mu_2(q)$ is

not. On the other hand, $\mu_2(q)$ is true in every model in M_2 , while $\mu_1(q)$ is not. Therefore, for every model of \mathcal{O} , there exists a substitution of variables that makes q true, which implies that $\mathcal{O} \models q$. \square

The example suggests that, in the presence of TBox atoms in the query, checking query entailment may require to reason by cases. We next illustrate two complexity lower bounds confirming that, in general, reasoning by cases is indeed unavoidable.

We start by proving that query entailment under MS is coNP-hard w.r.t. ontology complexity, based on a reduction from the complement of query entailment to the well-known NP-complete problem 3-SAT. Let F be a formula in the class 3CNF, let p_1, \dots, p_m be the propositional letters in F , and let $\{1..k\}$ denote the clauses in F . From F we construct an OWL 2 QL ontology \mathcal{O}^F and a query Q^F as follows.

- The signature Σ of \mathcal{O}^F is such that:
 - Σ_i contains B, G, E_1, E_2, H , and c_i for each clause i in F , with $j \in \{1 \dots k\}$;
 - Σ_c contains D, E_1, E_2, H ;
 - both Σ_i and Σ_c contain p_j, \bar{p}_j for each propositional letter p_j in F , with $j \in \{1 \dots m\}$;
 - Σ_p contains T, S, R_1, R_2, R_3 .
- The axioms of the ontology are used to force \mathcal{O}^F to represent the formula F , with the following basic ideas: (i) the fact $R_i(c_j, p_h)$ (resp. $R_i(c_j, \bar{p}_h)$) will model the situation where the i -th literal of c_j is p_h (resp. $\neg p_h$), and (ii) a model of \mathcal{O}^F where p_h is disjoint from D will represent a truth assignment for F where p_h is false, whereas a model of \mathcal{O}^F where p_h is not disjoint from D will represent a truth assignment for F where p_h is true.

In what follows, we denote by $L_{i,j}$ the class corresponding to the j -th literal in the i -th clause, where $i \in \{1, \dots, k\}$, and $j \in \{1, 2, 3\}$. For example, if the first literal in the clause 2 is p_2 , and the third literal in the clause 4 is $\neg p_3$, then $L_{2,1}$ is p_2 , and $L_{4,3}$ is \bar{p}_3 . The ontology \mathcal{O}^F is constituted by the following axioms:

- $H \sqsubseteq_c \neg D$,
- $E_1(B), E_2(B), S(E_1, E_2), R_1(G, H), R_2(G, H), R_3(G, H)$,
- $\forall i \in \{1, \dots, m\}: S(p_i, \bar{p}_i), T(G, p_i)$,
- $\forall i \in \{1, \dots, k\}: T(c_i, E_1), R_1(c_i, L_{i,1}), R_2(c_i, L_{i,2}), R_3(c_i, L_{i,3})$.

Fig. 3 shows an example of \mathcal{O}^F for $F = (p_1 \vee \bar{p}_2 \vee p_3) \wedge (p_2 \vee \bar{p}_1 \vee \bar{p}_3)$.

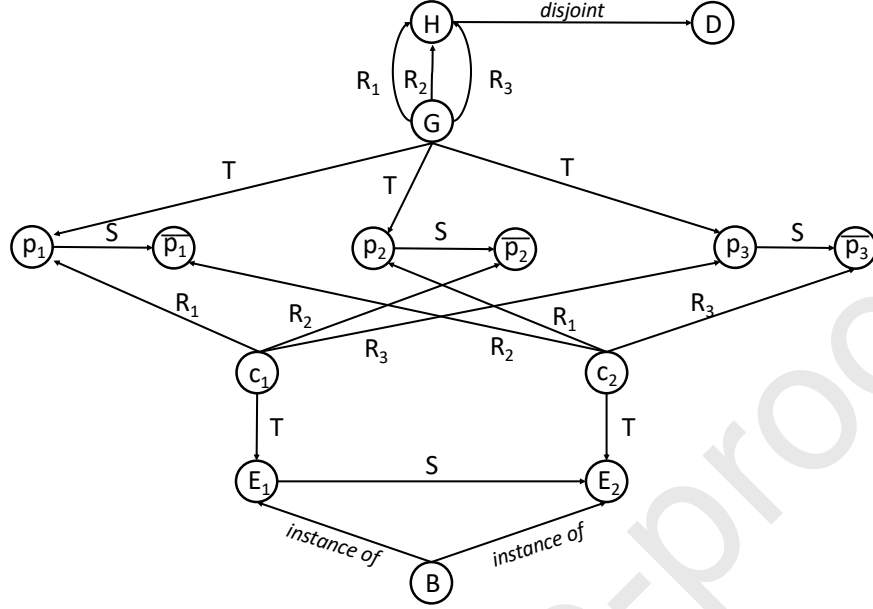


Figure 3: Encoding of the formula $F = (p_1 \vee \bar{p}_2 \vee p_3) \wedge (p_2 \vee \bar{p}_1 \vee \bar{p}_3)$

- According to the two ideas illustrated above, F is satisfiable exactly if there is a model of \mathcal{O}^F where for every pair p_i, \bar{p}_i , one of them is not disjoint from D , and every c_i is related by means of R_1, R_2 or R_3 to at least one literal that is not disjoint from D . We will prove that the negation of the above condition can be checked by means of a boolean conjunctive query Q^F that expresses that in every model of \mathcal{O}^F : (i) there exists i such that both p_i and \bar{p}_i are non-empty, or (ii) there exists a c_i such that all its R_1, R_2 , and R_3 -fillers are disjoint from D . The query Q^F is defined as follows:

$$Q^F \leftarrow T(x, z_1) \wedge R_1(x, y_1) \wedge R_2(x, y_2) \wedge R_3(x, y_3) \wedge y_1 \sqsubseteq_c \neg D \wedge y_2 \sqsubseteq_c \neg D \wedge y_3 \sqsubseteq_c \neg D \wedge S(z_1, z_2) \wedge z_1(w_1) \wedge z_2(w_2)$$

Note that the “or” of conditions (i) and (ii) is obtained by the atom $T(x, z_1)$, that is valid in \mathcal{O}^F in exactly two cases: in the first case, x is bound to a specific c_i , and z_1 is bound to E_1 , so that condition (i) is trivially true with z_2 bound to E_2 , and condition (ii) is checked on c_i by the appropriate atoms of Q^F (i.e., $R_1(x, y_1), R_2(x, y_2), R_3(x, y_3), y_1 \sqsubseteq_c \neg D, y_2 \sqsubseteq_c \neg D, y_3 \sqsubseteq_c \neg D$); in the second case, z_1 is bound to a certain p_i , and x is bound to G , so that condition (i) is checked on p_i, \bar{p}_i by the atoms $S(z_1, z_2), z_1(w_1), z_2(w_2)$ of Q^F , and condition (ii) is trivially true with y_1, y_2, y_3 bound to H .

Theorem 23. F is satisfiable if and only if $\mathcal{O}^F \not\models Q^F$.

Proof. If-part. Let $\mathcal{I} = \langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle$ be a model of \mathcal{O}^F such that $\mathcal{I} \not\models Q^F$, and let M_F be the propositional assignment defined as follows:

- if $(p_i^{\mathcal{I}})^C$ is not disjoint from $(D^{\mathcal{I}})^C$, then p_i is true, and \bar{p}_i is false in M_F ;
- if $(p_i^{\mathcal{I}})^C$ is disjoint from $(D^{\mathcal{I}})^C$, then p_i is false, and \bar{p}_i is true in M_F .

We show that M_F is a model of F . First, notice that for every i , p_i and \bar{p}_i have clearly different truth value. Second we show that for every i , if $(p_i^{\mathcal{I}})^C$ is not disjoint from $(D^{\mathcal{I}})^C$, then $(\bar{p}_i^{\mathcal{I}})^C$ is disjoint from $(D^{\mathcal{I}})^C$. Indeed, suppose, by contradiction, that for some i , both $(p_i^{\mathcal{I}})^C$ and $(\bar{p}_i^{\mathcal{I}})^C$ are not disjoint from $(D^{\mathcal{I}})^C$, implying that there exists a, b such that $a \in (p_i^{\mathcal{I}})^C$ and $b \in (\bar{p}_i^{\mathcal{I}})^C$. Now, consider the function h such that $h(x) = G^{\mathcal{I}}$, $h(z_1) = p_i^{\mathcal{I}}$, $h(y_1) = h(y_2) = h(y_3) = H^{\mathcal{I}}$, $h(z_2) = \bar{p}_i^{\mathcal{I}}$, $h(w_1) = a$, and $h(w_2) = b$. It is easy to see that h is a query homomorphism from Q^F to \mathcal{I} , and therefore $\mathcal{I} \models Q^F$, which is a contradiction.

Third, we show that for every clause i in F , there is at least one literal that is true according to M_F . Indeed, consider clause i , and suppose that every literal in i is false in M_F , i.e., every p_j appearing positive in i is false in M_F , and every p_j appearing negative in i is true in M_F . By construction of M_F , this means that for every class p_j corresponding to a letter appearing positive in i , $(p_j^{\mathcal{I}})^C$ is disjoint from $(D^{\mathcal{I}})^C$, and for every class \bar{p}_j corresponding to a letter appearing negative in i , $(p_j^{\mathcal{I}})^C$ is not disjoint from $(D^{\mathcal{I}})^C$, that, as we saw before, implies that $(\bar{p}_j^{\mathcal{I}})^C$ is disjoint from $(D^{\mathcal{I}})^C$. But this in turn implies that the function f such that $f(x) = c_i^{\mathcal{I}}$, $f(z_1) = E_1^{\mathcal{I}}$, $f(z_2) = E_2^{\mathcal{I}}$, $f(w_1) = f(w_2) = B^{\mathcal{I}}$, $f(y_1) = L_{i,1}^{\mathcal{I}}$, $f(y_2) = L_{i,2}$, and $f(y_3) = L_{i,3}^{\mathcal{I}}$, is a query homomorphism from Q^F to \mathcal{I} , and therefore $\mathcal{I} \models Q^F$, which is a contradiction.

Thus, we conclude that M_F is indeed a model of F .

Only-if-part. Let M_F be a model of F , and let $\mathcal{I} = \langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle$ be the interpretation for \mathcal{O}^F defined as follows (where a is in the domain of \mathcal{I}):

- $(D^{\mathcal{I}})^C = \{a\}$
- if p_i is true in M_F , then $(p_i^{\mathcal{I}})^C = \{a\}$, and $(\bar{p}_i^{\mathcal{I}})^C = \emptyset$;
- if p_i is false in M_F , then $(p_i^{\mathcal{I}})^C = \emptyset$, and $(\bar{p}_i^{\mathcal{I}})^C = \{a\}$;
- all other aspects of \mathcal{I} are such that exactly the facts in \mathcal{O}^F are true in \mathcal{I} ; for example, $(E_1^{\mathcal{I}})^C = \{B^{\mathcal{I}}\}$, $(E_2^{\mathcal{I}})^C = \{B^{\mathcal{I}}\}$.

Clearly, \mathcal{I} is a model of \mathcal{O}^F . We show that $\mathcal{I} \not\models Q^F$. Indeed, the only possibility for Q^F to be true in \mathcal{I} is the existence of a query homomorphism h from Q^F

to \mathcal{I} such that $(h(x), h(z_1)) \in (T^{\mathcal{I}})^P$, which corresponds to the following two cases:

1. $h(x) = G^{\mathcal{I}}$, $h(y_1) = h(y_2) = h(y_3) = H^{\mathcal{I}}$, and $h(z_1) = p_j^{\mathcal{I}}$, $h(z_2) = \bar{p}_j^{\mathcal{I}}$ for a certain j . In this case, by construction of \mathcal{I} , either $(h(z_1)^{\mathcal{I}})^C$ or $(h(z_2)^{\mathcal{I}})^C$ is empty, contradicting that h is a query homomorphism from Q^F to \mathcal{I} .
2. $h(x) = c_i^{\mathcal{I}}$ for a certain i , $h(z_1) = E_1^{\mathcal{I}}$, and $h(y_1) = L_{i,1}^{\mathcal{I}}$, $h(y_2) = L_{i,2}^{\mathcal{I}}$, $h(y_3) = L_{i,3}^{\mathcal{I}}$. In this case, since M_F is a model of F , by construction of \mathcal{I} we have that $a^{\mathcal{I}} \in (L_{i,1}^{\mathcal{I}})^C$, $a^{\mathcal{I}} \in (L_{i,2}^{\mathcal{I}})^C$, or $a^{\mathcal{I}} \in (L_{i,3}^{\mathcal{I}})^C$, and therefore one among $(L_{i,1}^{\mathcal{I}})^C$, $(L_{i,2}^{\mathcal{I}})^C$, and $(L_{i,3}^{\mathcal{I}})^C$ is not disjoint from $D^{\mathcal{I}}$ in \mathcal{I} , contradicting that h is a query homomorphism from Q^F to \mathcal{I} .

Thus, we conclude that no query homomorphism exists from Q^F to \mathcal{I} , and therefore $\mathcal{I} \not\models Q^F$. \square

Note that in the above reduction both the ontology and the query contain only TBox axioms of the form $\alpha_1 \sqsubseteq_c \neg\alpha_2$. It is not difficult to see that a similar proof can be obtained under the assumption that only the axioms of the form $\alpha_1 \sqsubseteq_p \neg\alpha_2$, or of the form $\text{Irr}(\alpha)$ appear in the ontology and the query.

From Theorem 23 and from the observation that the size of \mathcal{O}^F is polynomial with respect to the size of F , and the size of Q^F does not depend on F , we immediately derive the following corollary.

Corollary 24. *Query entailment under MS is coNP-hard w.r.t. ontology complexity.*

Next, we prove that query entailment under MS is Π_2^P -hard w.r.t. combined complexity. We consider the problem of checking the satisfiability of a 2-QBF, i.e., a Quantified Boolean Formula of the form $\forall x_1, \dots, x_n \exists y_1, \dots, y_m c_1 \wedge \dots \wedge c_k$, where each c_i is a clause with exactly three literals on the boolean variables $x_1, \dots, x_n, y_1, \dots, y_m$. First, notice that, given a 2-QBF F , we can obtain a new formula F' as follows. For each boolean variable z of F :

1. we introduce a variable \bar{z} ;
2. we substitute every negative literal $\neg z$ in F with \bar{z} ;
3. we add every \bar{z} to the list of existentially quantified variable;

4. we add to F suitable clauses $c_{z,1}$ and $c_{z,2}$ to sanction that exactly one between z , \bar{z} is true, with $c_{z,1} = (z \vee \bar{z})$ and $c_{z,2} = (\neg z \vee \neg \bar{z})$. In order to distinguish these newly introduced clauses, we call them *auxiliary clauses*, while we call *original clauses* the clauses originally belonging to F .

It is easy to see that F' is a 2-QBF that is satisfiable if and only if F is satisfiable, and that the size of F' is linear with respect to the size of F . Thus, in the following, we focus on the 2-QBF formula F' . Also, for clarity purposes, we denote by x_1, \dots, x_n and by y_1, \dots, y_m the boolean variables of F' that are, respectively, universally and existentially quantified. Also, we will use z (possibly with subscript) to refer to boolean variables of F' of any of the two kinds mentioned above.

From F' , we define an OWL 2 QL ontology $\mathcal{O}^{F'}$ and a query $Q^{F'}$ as follows.

- The signature Σ of $\mathcal{O}^{F'}$ is such that:
 - Σ_i contains a ; moreover, it contains for each original clause c_i , with $i \in \{1, \dots, k\}$, an individual c_i , as well as individuals t_{l_1, l_2, l_3}^i and $v_{l_1, l_2, l_3}^{i, h}$, with $l_1, l_2, l_3 \in \{0, 1\}$ and $h \in \{1, 2, 3\}$, such that at least one of the l_h 's is equal to 1; finally, it contains, for each variable z , an individual $c_{z,1}$, as well as individuals t_{l_1, l_2}^z and $v_{l_1, l_2}^{z, h}$, with $l_1, l_2 \in \{0, 1\}$ and $h \in \{1, 2\}$, such that exactly one of the l_h 's is equal to 1;
 - both Σ_i and Σ_c contain D and E ; moreover, for each universally quantified variable x_j , with $j \in \{1, \dots, n\}$, they both contain one class named x_j ;
 - Σ_p contains In, Has, Val ;
 - $\Sigma_d, \Sigma_i, \Sigma_t$, and Σ_{Lit} are empty.
- Intuitively, the axioms of the ontology are used to force $\mathcal{O}^{F'}$ to encode, for each clause of F' , all possible assignments for the variables occurring in the clause, that set a truth value for each existentially quantified variable and keep undefined all universally quantified variables. Note, in particular, that assignments of existentially quantified variables are defined by identifying such variables only on the basis of their position within the clause and by appropriately connecting occurrences of each variables to the empty class E to set it to false and to the non-empty class D to set it to true. On the other hand, assignments of universally quantified variables are kept undefined by representing each such variable x_j through a class x_j such that it is uncertain whether it is empty or not, i.e., the axiom $x_j \sqsubseteq_c \neg x_j$ is uncertain in $\mathcal{O}^{F'}$, in the sense that neither such axiom nor its negation are logically implied by $\mathcal{O}^{F'}$. Then, each completion of $\mathcal{O}^{F'}$ encodes all assignments of truth values to both existentially quantified variables and universally quantified variable, based on the assumption

that for each x_j , with $j \in \{1, \dots, m\}$, an assignment to true is encoded by violating the axiom $x_j \sqsubseteq_c \neg x_j$, while an assignment to false is encoded by introducing the axiom $x_j \sqsubseteq_c \neg x_j$.

To provide a better intuition of the encoding, consider the following example. Let c_i be an original clause having the form $c_i = (x_j \vee y_f \vee y_k)$, where $j \in \{1, \dots, n\}$ and $f, k \in \{1, \dots, m\}$. The set of axioms

$$\begin{aligned} & In(c_i, t_{1,0,1}^i), Has(t_{1,0,1}^i, x_j), Val(x_j, x_j) \\ & Has(t_{1,0,1}^i, v_{1,0,1}^{i,2}), Val(v_{1,0,1}^{i,2}, E) \\ & Has(t_{1,0,1}^i, v_{1,0,1}^{i,3}), Val(v_{1,0,1}^{i,3}, D) \end{aligned}$$

models a truth value assignment for the variables in c_i (represented by the individual $t_{1,0,1}^i$), that sets the existentially quantified variable occurring in second position of c_i to false (by representing such occurrence through the individual $v_{1,0,1}^{1,2}$, and connecting it to the empty class E through the property Val) and the one occurring in third position to true (by representing such occurrence through the individual $v_{1,0,1}^{1,3}$, and connecting it to the non-empty class D through Val). Observe, also, that the assignment encoded by the above set of axioms keeps undefined the truth value assigned to the universally quantified variable x_j , by representing it as a class such that the axiom $x_j \sqsubseteq_c \neg x_j$ is uncertain. Then, by extending the set with the axiom $x_j(e_j)$, where e_j is an entity not occurring anywhere else in the ontology, we obtain a completion of $\mathcal{O}^{F'}$ encoding an assignment that sets to true the universally quantified variable x_j .

Based on the above intuition, $\mathcal{O}^{F'}$ consists of the following axioms:

- $E \sqsubseteq_c \neg E$ is in $\mathcal{O}^{F'}$;
- $D(a)$ is in $\mathcal{O}^{F'}$;
- for $i \in \{1, \dots, k\}$, the axiom $In(c_i, t_{l_1, l_2, l_3}^i)$ is in $\mathcal{O}^{F'}$, for every t_{l_1, l_2, l_3}^i ;
- for each variable z of F' , the axiom $In(c_{z,1}, t_{l_1, l_2}^z)$ is in $\mathcal{O}^{F'}$, for every t_{l_1, l_2}^z ;
- for each axiom $In(c_i, t_{l_1, l_2, l_3}^i)$, and for each $h \in \{1, 2, 3\}$, let z_h be the h -th variable of c_i ; if z_h is universally quantified, then $Has(t_{l_1, l_2, l_3}^i, z_h)$ is in $\mathcal{O}^{F'}$, otherwise $Has(t_{l_1, l_2, l_3}^i, v_{l_1, l_2, l_3}^{i, h})$ is in $\mathcal{O}^{F'}$;
- for each axiom $In(c_{z,1}, t_{l_1, l_2}^z)$ in $\mathcal{O}^{F'}$, $Has(t_{l_1, l_2}^z, v_{l_1, l_2}^{z, 2})$ is in $\mathcal{O}^{F'}$; moreover, if z is universally quantified, then $Has(t_{l_1, l_2}^z, z)$ is in $\mathcal{O}^{F'}$, otherwise $Has(t_{l_1, l_2}^z, v_{l_1, l_2}^{z, 1})$ is in $\mathcal{O}^{F'}$;
- for each axiom $Has(t_{l_1, l_2, l_3}^i, v_{l_1, l_2, l_3}^{i, h})$ in $\mathcal{O}^{F'}$, $Val(v_{j_1, j_2, j_3}^{i, h}, D)$ is in $\mathcal{O}^{F'}$ if $l_h = 1$, otherwise $Val(v_{l_1, l_2, l_3}^{i, h}, E)$ is in $\mathcal{O}^{F'}$;
- for each axiom $Has(t_{l_1, l_2}^z, v_{l_1, l_2}^{z, 1})$ in $\mathcal{O}^{F'}$, $Val(v_{l_1, l_2}^{z, h}, D)$ is in $\mathcal{O}^{F'}$ if $l_h = 1$, otherwise $Val(v_{l_1, l_2}^{z, h}, E)$ is in $\mathcal{O}^{F'}$;

- for each universally quantified variable x_j of F' , the axioms $Val(x_j, x_j)$ and $x_j \sqsubseteq_c \top_c$ are in $\mathcal{O}^{F'}$.
- According to the ideas illustrated above, F' is satisfiable if for every completion, there exists a subset of the axioms of $\mathcal{O}^{F'}$ that encodes an assignment for the variables of F' that satisfies all clauses of F' . Thus, we define the conjunctive query $Q^{F'}$ as follows:

$$Q^F \leftarrow \gamma_1 \wedge \gamma_2$$

where γ_1 and γ_2 are conjunctions of atoms that we now define. In the following we use the symbol $a_{i,j}$ to denote a query variable whose name coincides with the name of the j -th variable occurring into the i -th clause of F' . Thus, for example, if the clause c_1 has the form $(x_1 \vee \bar{y}_1 \vee y_2)$, $a_{1,1}$ denotes the variable x_1 , $a_{1,2}$ the variable \bar{y}_1 , and $a_{1,3}$ the variable y_2 .

- γ_1 is the conjunction of atoms containing, for each clause c_i , the following atoms, forming what we call the c_i -subquery:

$$\begin{aligned} & In(c_i, w_i), \\ & Has(w_i, s_{i,1}), \quad Val(s_{i,1}, a_{i,1}), \\ & Has(w_i, s_{i,2}), \quad Val(s_{i,2}, a_{i,2}), \\ & Has(w_i, s_{i,3}), \quad Val(s_{i,3}, a_{i,3}), \\ & Has(w_i, p_i), \quad Val(p_i, y_i), \\ & y_i(u_i) \end{aligned}$$

Intuitively, each c_i -subquery checks for the existence of a set of atoms within every completion of $\mathcal{O}^{F'}$ that encodes an assignment to the variables occurring in c_i that satisfies c_i .

- γ_2 is the conjunction of atoms containing, for each variable z (and then for each auxiliary clause $c_{z,1}$), the following atoms, forming what we call the z -subquery:

$$\begin{aligned} & In(c_{z,1}, w_z), \\ & Has(w_z, s_{z,1}), \quad Val(s_{z,1}, z), \\ & Has(w_z, s_{z,2}), \quad Val(s_{z,2}, \bar{z}), \\ & Has(w_z, p_{z,1}), \quad Val(p_{z,1}, y_{z,1}), \\ & Has(w_z, p_{z,2}), \quad Val(p_{z,2}, y_{z,2}), \\ & y_{z,1}(u_z), \quad y_{z,2} \sqsubseteq_c \neg y_{z,2} \end{aligned}$$

Intuitively, each z -subquery checks for the existence of a set of atoms within every completion of $\mathcal{O}^{F'}$ that encodes an assignment such that exactly one among the variables z and \bar{z} is true.

Theorem 25. F is satisfiable if and only if $\mathcal{O}^{F'} \models Q^{F'}$.

Proof. \Rightarrow Suppose that F' is satisfiable, i.e., for every assignment α to the universally quantified variables of F' , the formula obtained from F' by substituting

each occurrence of the universally quantified variables x_1, \dots, x_n with the value that α assigns to such variables, is satisfiable. We show that the certain answer to $Q^{F'}$ over $\mathcal{O}^{F'}$ is true, by showing that for every complete ontology derived from $\mathcal{O}^{F'}$, the certain answer to the query is true.

Consider any assignment α to the universally quantified variables of F' , and let $\mathcal{O}_{F'}^\alpha$ be the ontology obtained from $\mathcal{O}^{F'}$ by asserting that the class x_j is empty for each universally quantified variable x_j to which α assigns false, and by asserting that x_j is non-empty for each universally quantified variable x_j to which α assigns true. We show that the certain answer to $Q^{F'}$ over $\mathcal{O}_{F'}^\alpha$ is true. Consider any assignment β to the existentially quantified variables of F' such that $\alpha \cup \beta$ makes the body of F' (i.e., the set of clauses constituting F') true, and define the binding B for the variables of $Q^{F'}$ as follows:

- For each clause $c_i = (z_1 \vee z_2 \vee z_3)$ and the associated c_i -subquery of $Q^{F'}$, w_i is bound to the object t_{l_1, l_2, l_3}^i such that l_1, l_2, l_3 is the combination of values that $\alpha \cup \beta$ assigns to the variables z_1, z_2 , and z_3 appearing in clause c_i , whereas $s_{i, h}$ is bound to the variable z_h if z_h is a universally quantified variable and, in this case, also the query variable z_h is bound to z_h ; otherwise $s_{i, h}$ is bound to $v_{l_1, l_2, l_3}^{i, h}$ and the query variable z_h is bound to E or to D according to whether $l_h = 0$ or $l_h = 1$, respectively. Finally, the query variables p_i and y_i are bound to the same values to which are bound any of the query variables $s_{i, h}$ and z_h , respectively, for h such that $l_h = 1$. Note that by construction, there must be at least one such h .
- For each boolean variable z of F' and the associated z -subquery of $Q^{F'}$, w_p is bound to the object t_{j_1, j_2}^z such that j_1, j_2 is the combination of values that $\alpha \cup \beta$ assigns to the variables z and \bar{z} , whereas $s_{z, 1}$ is bound to z if z is universally quantified in F' and, in this case, the query variable z is also bound to z ; otherwise $s_{z, h}$ is bound to $v_{l_1, l_2}^{z, h}$ and the query variables z and \bar{z} are respectively bound to D and E if $l_h = 1$, while they are respectively bound to E and D , otherwise. Finally, $p_{z, 1}$ and $p_{z, 2}$ are bound to z and \bar{z} respectively, if $l_1 = 1$ (i.e., if z is assigned to true by $\alpha \cup \beta$), otherwise they are bound to the \bar{z} and z , respectively.

It is immediate to verify that the binding B makes the answer to $Q^{F'}$ true.

\Leftarrow Suppose that the certain answer to $Q^{F'}$ w.r.t. $\mathcal{O}^{F'}$ is true. We show that F' is satisfiable. If the certain answer to $Q^{F'}$ w.r.t. $\mathcal{O}^{F'}$ is true, then it follows that for every choice we make (empty or non-empty) for each class x_j corresponding to the universally quantified variable x_j , there exists a binding for the variables of $Q^{F'}$ such that $Q^{F'}$ is true. Consider any such binding B . By essentially reasoning in reverse with respect to the other direction of the theorem, one can easily verify that from B we can derive an assignment for the existentially quantified variables of F' that makes F' true. This proves that if

the certain answer to $Q^{F'}$ w.r.t. $\mathcal{O}^{F'}$ is true, then F' is satisfiable. \square

By combining Theorem 25 with the observation that both $\mathcal{O}^{F'}$ and $Q^{F'}$ have a size that is polynomial with respect to the size of F' , we easily derive the following corollary.

Corollary 26. *Query entailment under MS is Π_2^P -hard w.r.t. combined complexity.*

6. Entailment of general queries: upper bounds

In this section we present an algorithm for checking the entailment of general queries under MS that matches the lower bounds described in the previous section. To this aim, we start by observing that the results of the previous section show that the complexity of the problem originates from negative axioms which are uncertain, i.e., they are neither logically implied by the ontology nor violated by it. In other words, the presence of uncertain negative axioms forces us to reason by cases when answering queries. Motivated by this observation, in this section we start by studying query entailment over a class of ontologies, called *TBox-complete*, in which, intuitively, there is complete knowledge about negative axioms. Then we move to unrestricted ontologies, and show that query entailment can be reduced to query entailment over a set of TBox-complete ontologies.

6.1. The case of TBox-complete ontologies

We remind the reader that we implicitly refer to an OWL 2 QL ontology \mathcal{O} over vocabulary V and with signature Σ . In the following, we denote by $\mathcal{N}^{\mathcal{O}}$ the set of all negative axioms that can be built starting from the expressions in Exp .

We next introduce the notion of TBox-complete ontology, which in turn is based on the notion of certain negative axiom. Intuitively, an axiom is certain if it is either logically implied by the ontology, or it is violated in the ontology.

Definition 27. An axiom α in $\mathcal{N}^{\mathcal{O}}$ is *certain* if either $\mathcal{O} \models \alpha$ or $\mathcal{O} \models \neg\alpha$. An axiom that is not certain is called *uncertain*. \mathcal{O} is *TBox-complete* if every axiom in $\mathcal{N}^{\mathcal{O}}$ is certain.

Note that, by the above definition, every unsatisfiable ontology is trivially TBox-complete.

Example 28. The ontology presented in Example 22 is not TBox-complete. Indeed, for example, the axiom $A \sqsubseteq_c \neg B$ is neither logically implied nor violated by it. \square

TBox-complete ontologies enjoy notable properties, which turn out to be crucial for entailment of general queries. In the following, we show that in the case of TBox-complete ontologies, the canonical pseudo-model is crucial not only for the entailment of instance queries, as we showed in Section 4, but also for the entailment of general queries. Towards this goal, following the same line of reasoning used for instance queries, we now define the notion of *extended homomorphism*, which, intuitively, is an extension of the notion of instance-based homomorphism that also considers semantic properties expressed as TBox axioms.

Definition 29. Let $\mathcal{I} = \langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle$ be an interpretation for \mathcal{O} . An *extended homomorphism* from $Can(\mathcal{O})$ to \mathcal{I} is a function Ψ from $\bar{\Delta}^o$ to Δ^o and from $\bar{\Delta}^v$ to Δ^v such that:

- Ψ is an instance-based homomorphism from $Can(\mathcal{O})$ to \mathcal{I} , and
- Ψ further preserves in \mathcal{W} the following intensional properties of $\mathcal{W}_{Can(\mathcal{O})}$:
 5. for every $e_1, e_2 \in \bar{\Delta}^o$, if $e_1^{\bar{C}} \cap e_2^{\bar{C}} = \emptyset$, then $\Psi(e_1)^C \cap \Psi(e_2)^C = \emptyset$;
 6. for every $e_1, e_2 \in \bar{\Delta}^o$, if $e_1^{\bar{P}} \cap e_2^{\bar{P}} = \emptyset$, then $\Psi(e_1)^P \cap \Psi(e_2)^P = \emptyset$;
 7. for every $e_1, e_2 \in \bar{\Delta}^o$, if $e_1^{\bar{D}} \cap e_2^{\bar{D}} = \emptyset$, then $\Psi(e_1)^D \cap \Psi(e_2)^D = \emptyset$;
 8. for every $e \in \bar{\Delta}^o$, if for every $e' \in \bar{\Delta}^o$, $(e', e') \notin e^{\bar{P}}$, then for every $e'' \in \Delta^o$, $(e'', e'') \notin \Psi(e)^P$;
 9. for every $e_1, e_2 \in \bar{\Delta}^o$, if $e_1^{\bar{C}} \subseteq e_2^{\bar{C}}$, then $\Psi(e_1)^C \subseteq \Psi(e_2)^C$;
 10. for every $e \in \bar{\Delta}^o$ and every $t \in \mathbb{D}$, if $\{e' \mid (e'', e') \in e^{\bar{D}} \text{ for some } e''\} \subseteq t^{\bar{T}}$, then $\{v \mid (o, v) \in \Psi(e)^D \text{ for some } o\} \subseteq \Psi(t)^T$;
 11. for every $e_1, e_2 \in \bar{\Delta}^o$, if $e_1^{\bar{P}} \subseteq e_2^{\bar{P}}$, then $\Psi(e_1)^P \subseteq \Psi(e_2)^P$;
 12. for every $e_1, e_2 \in \bar{\Delta}^o$, if $e_1^{\bar{D}} \subseteq e_2^{\bar{D}}$, then $\Psi(e_1)^D \subseteq \Psi(e_2)^D$;
 13. for every $e \in \bar{\Delta}^o$, if for every $e' \in \bar{\Delta}^o$ $(e', e') \in e^{\bar{P}}$, then for every $e'' \in \Delta^o$, $(e'', e'') \in \Psi(e)^P$.

The following proposition shows that if \mathcal{O} is TBox-complete, then besides representing all models of \mathcal{O} with respect to instance checking, $Can(\mathcal{O})$ represents all the models of \mathcal{O} with respect to logical implication of TBox axioms.

Proposition 30. *If \mathcal{O} is TBox-complete, then for every model $\mathcal{I} \in Mod_{MS}(\mathcal{O})$ there exists an extended homomorphism from $Can(\mathcal{O})$ to \mathcal{I} .*

Proof. If $Mod_{MS}(\mathcal{O}) = \emptyset$, then the statement is trivially true. Therefore, in what follows we assume $Mod_{MS}(\mathcal{O}) \neq \emptyset$, and we let $\mathcal{I} \in Mod_{MS}(\mathcal{O})$ be $\langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle$.

We assume that \mathcal{O} is TBox-complete and refer to the function Ψ defined in the proof of Proposition 16, where we showed that Ψ is an instance-based homomorphism from $Can(\mathcal{O})$ to \mathcal{I} . In the following we show that Ψ is indeed an extended homomorphism from $Can(\mathcal{O})$ to \mathcal{I} , i.e., that it further satisfies properties 5-13 of Definition 29, thus proving that an extended homomorphism from $Can(\mathcal{O})$ to \mathcal{I} always exists.

We remind the reader that, by definition, for every k , the k -portion of $Can(\mathcal{O})$, denoted $Can_k(\mathcal{O})$, is such that the functions $\cdot^{\bar{C}_k}$, $\cdot^{\bar{P}_k}$, $\cdot^{\bar{D}_k}$, and $\cdot^{\bar{T}_k}$ are all undefined for $s \in \mathcal{S}_o$ and for $w \in \mathcal{S}_v$. Therefore, in what follows, we check whether Ψ satisfies properties 5-13 of Definition 29 only for e, e_1, e_2 in Exp .

- Consider Property 5. Let $e_1, e_2 \in \bar{\Delta}^o$ be such that $e_1^{\bar{C}} \cap e_2^{\bar{C}} = \emptyset$, and suppose by contradiction that there exists $e \in \Delta^o$ such that $e \in \Psi(e_1)^C$, and $e \in \Psi(e_2)^C$. Since from the definition of Ψ , we know that for every $e \in Exp$, $\Psi(e) = e^{\mathcal{I}}$, we have that $e \in (e_1^{\bar{C}})^C \cap (e_2^{\bar{C}})^C$. Also, since \mathcal{I} is a model of \mathcal{O} , and \mathcal{O} is TBox-complete, we have that $\mathcal{O} \cup \{e_1 \sqsubseteq \neg_c e_2\}$ is unsatisfiable, or, equivalently, $\mathcal{O} \models \exists x.e_1(x) \wedge e_2(x)$. But then, by Proposition 11, there exists a query homomorphism from the query $\exists x.e_1(x) \wedge e_2(x)$ to $Can(\mathcal{O})$, which clearly contradicts $e_1^{\bar{C}} \cap e_2^{\bar{C}} = \emptyset$. Similarly, we can prove that Ψ satisfies Properties 6 and 7.
- Consider Property 8. Let $e \in \bar{\Delta}^o$ be such that for every $e' \in \bar{\Delta}^o$, $(e', e') \notin e^{\bar{P}}$, and suppose by contradiction that there exists $e'' \in \Delta^o$ such that $(e'', e'') \in \Psi(e)^{\bar{P}}$. By virtue of how Ψ is defined, this implies $(e'', e'') \in (e^{\mathcal{I}})^{\bar{P}}$. Since \mathcal{I} is a model of \mathcal{O} and \mathcal{O} is TBox-complete, we have that $\mathcal{O} \cup \{\text{Irr}(e)\}$ is unsatisfiable, or, equivalently, $\mathcal{O} \models \exists x.e(x, x)$. But then, by Proposition 11, there exists a query homomorphism from the query $\exists x.e(x, x)$ to $Can(\mathcal{O})$, which clearly contradicts that for every $e' \in \bar{\Delta}^o$, $(e', e') \notin e^{\bar{P}}$.
- Consider Property 9. We recall that Ψ is defined by induction on k , where k is the number of rule applications which leads to $Chase_k(\mathcal{O})$. In particular, Ψ_k is a function on $\bar{\Delta}_k^o \cup \bar{\Delta}_k^v$, such that $\Psi = \bigcup_{j \in \mathbb{N}} \Psi_j$, where $\bar{\Delta}_k^o$ and $\bar{\Delta}_k^v$ are respectively the object and value domain of $Can_k(\mathcal{O})$. We then show, by induction on k , that, for every $k \geq 0$, Ψ_k satisfies Property 9, and we will do it by preventively showing that

(i) for every $e_1, e_2 \in Exp$, if $\alpha_{e_1} \in e_2^{\bar{C}_k}$, then $\mathcal{O} \models e_1 \sqsubseteq_c e_2$.

Base step: $k = 0$. We show that (i) holds. Since by construction of $Chase_0(\mathcal{O})$, $\alpha_{e_1} \in e_2^{\bar{C}_0}$ implies $e_1 = e_2$, and since obviously $\mathcal{O} \models e_1 \sqsubseteq_c e_1$,

we conclude that property (i) is satisfied.

We now show that Ψ_0 satisfies Property 9. The only non-trivial case is $e_1 \neq e_2$. Notice that, if $e_1^{\bar{C}_0} \neq \emptyset$, then by construction of $Chase_0(\mathcal{O})$, there exists α_{e_1} in \mathcal{S}_o^0 such that $\alpha_{e_1} \in e_1^{\bar{C}_0}$ and $\alpha_{e_1} \notin e^{\bar{C}_0}$ for every $e \neq e_1$. In particular, $\alpha_{e_1} \notin e_2^{\bar{C}_0}$, and therefore $e_1^{\bar{C}_0} \subseteq e_2^{\bar{C}_0}$ implies $e_1^{\bar{C}_0} = \emptyset$. which contradicts our hypothesis. Hence, it must be the case that $e_1^{\bar{C}_0} = \emptyset$. Thus, since \mathcal{O} is TBox-complete, it follows that $\mathcal{O} \models e_1 \sqsubseteq_c \neg e_1$ and, therefore, $\mathcal{O} \models e_1 \sqsubseteq_c e_2$ for every $e_2 \in Exp_c$. Since \mathcal{I} is a model of \mathcal{O} , we conclude that $\Psi_0(e_1)^C \subseteq \Psi_0(e_2)^C$, which proves that Ψ_0 satisfies property 9.

Inductive step: $k \geq 0$. We assume that (i) holds for k and we prove that it holds for $k+1$. Suppose that $\alpha_{e_1} \in e_2^{\bar{C}_{k+1}}$. Clearly, the only non trivial case is when $\alpha_{e_1} \notin e_2^{\bar{C}_k}$. It is easy to see that this can happen only if $Chase_{k+1}(\mathcal{O})$ is obtained from $Chase_k(\mathcal{O})$ by the application of rule 1 on the basis of $e_3(\alpha_{e_1})$, $e_3 \sqsubseteq_c e_2 \in Chase_k(\mathcal{O})$. Then, by inductive hypothesis, $\mathcal{O} \models e_1 \sqsubseteq_c e_3$, which combined with $e_3 \sqsubseteq_c e_2$, proves that $\mathcal{O} \models e_1 \sqsubseteq_c e_2$.

Let us now assume that Ψ_k satisfies Property 9, and let us show that Ψ_{k+1} also satisfies Property 9. Suppose that $e_1^{\bar{C}_{k+1}} \subseteq e_2^{\bar{C}_{k+1}}$. The only non trivial case is when $\mathcal{O} \models \neg(e_1 \sqsubseteq_c \neg e_1)$. Then, $\alpha_{e_1} \in e_1^{\bar{C}_0}$ and $\alpha_{e_1} \in e_1^{\bar{C}_{k+1}}$, which by implies that $\alpha_{e_1} \in e_2^{\bar{C}_{k+1}}$. Thus, it follows from property (i) that $\mathcal{O} \models e_1 \sqsubseteq_c e_2$, and since \mathcal{I} is a model of \mathcal{O} , we have $\Psi_{k+1}(e_1)^C \subseteq \Psi_{k+1}(e_2)^C$, which means that Ψ_{k+1} satisfies Property 9.

- The proof for Property 10, 11 and 12, is analogous to the one for Property 9. In particular, they are respectively based to the following properties, analogous to (i):

(ii) for every $t \in \mathbb{D}$, if $w_e \in t^{\bar{T}_k}$, then $\mathcal{O} \models \rho(e) \sqsubseteq_t t$;

(iii) for every $e_1, e_2 \in Exp$, if $(\beta_{e_1}^1, \beta_{e_1}^2) \in e_2^{\bar{F}_k}$, then $\mathcal{O} \models e_1 \sqsubseteq_p e_2$;

(iv) for every $e_1, e_2 \in Exp$, if $(\gamma_{e_1}, w_{e_1}) \in e_2^{\bar{D}_k}$, then $\mathcal{O} \models e_1 \sqsubseteq_d e_2$.

- Consider Property 13. Similarly to the other cases, we show, by induction on k , that, for every $k \geq 0$, Ψ_k satisfies Property 13, and we will do it by preventively showing that

(v) for every $e \in Exp$, if $(\beta_e^1, \beta_e^1) \in e^{\bar{P}_k}$, then $\mathcal{O} \models \text{Ref}(e)$.

Base step: $k = 0$. Since by construction of $Chase_0(\mathcal{O})$, for no $e \in Exp_p$ it holds that $(\beta_e^1, \beta_e^1) \in e^{\bar{P}_0}$, we have that property (v) is satisfied.

We next show that Ψ_0 satisfies Property 13. Notice that, by construction, $Chase_0(\mathcal{O})$ contains at least the axiom $\top_c(\alpha_{\top_c})$ since $\top_c \in Exp_c$ and $\mathcal{O} \cup \top_c \sqsubseteq_c \neg \top_c$ is unsatisfiable. Hence, at least the object α_{\top_c} is in $\bar{\Delta}_0^o$, but for every $e \in Exp_p$, $(\alpha_{\top_c}, \alpha_{\top_c}) \notin e^{\bar{P}_0}$. Therefore, the premise of

Property 13 never holds and the Property is trivially satisfied.

Inductive step: $k \geq 0$. We assume that (v) holds for k and we prove that it holds for $k+1$. Suppose that $(\beta_e^1, \beta_e^1) \in e^{P_{k+1}^-}$. Clearly, the only non trivial case is when $(\beta_e^1, \beta_e^1) \notin e^{P_k}$. It is easy to see that this can happen only if $Chase_{k+1}(\mathcal{O})$ is obtained from $Chase_k(\mathcal{O})$ by the application of one of the following rules:

- rule **2**, on the basis of $e'(\beta_e^1, \beta_e^1)$, $e' \sqsubseteq_p e \in Chase_k(\mathcal{O})$; but then, by inductive hypothesis, $\mathcal{O} \models \text{Ref}(e')$, which combined with $e' \sqsubseteq_p e$, proves that $\mathcal{O} \models \text{Ref}(e)$;
- rule **5**, on the basis of $e(\beta_e^1, e')$, $\text{Ref}(e) \in Chase_k(\mathcal{O})$; then, clearly, $\mathcal{O} \models \text{Ref}(e)$;
- rule **10**, on the basis of $e^-(\beta_e^1, \beta_e^1) \in Chase_k(\mathcal{O})$; then, by inductive hypothesis, $\mathcal{O} \models \text{Ref}(e^-)$, which proves that $\mathcal{O} \models \text{Ref}(e)$.

Let us now assume that Ψ_k satisfies Property 13, and let us show that Ψ_{k+1} also satisfies Property 13. Suppose that for every $e' \in \Delta_{k+1}^{\bar{o}}$ ($e', e' \in e^{P_{k+1}^-}$).

Since e is non empty, then $\mathcal{O} \models \neg(e \sqsubseteq_p \neg e)$, and $(\beta_e^1, \beta_e^2) \in e^{\bar{P}_0}$. But then $\beta_e^1 \in \bar{\Delta}_0^{\bar{o}}$ and $\beta_e^2 \in \Delta_{k+1}^{\bar{o}}$, which implies that $(\beta_e^1, \beta_e^2) \in e^{P_{k+1}^-}$. Thus, it follows from (v) that $\mathcal{O} \models \text{Ref}(e)$, and since \mathcal{I} is a model of \mathcal{O} , we have that for every $e' \in \Delta^{\bar{o}}$ ($e', e' \in \Psi(e)^P$).

□

We are now ready to generalize Proposition 17, and show that, if \mathcal{O} is satisfiable and TBox-complete, then $Can(\mathcal{O})$ is representative of all its models with respect to entailment of general queries.

Proposition 31. *If \mathcal{O} is satisfiable and TBox-complete, and $Q = q_1 \cup \dots \cup q_n$ is a general query over \mathcal{O} , then $\mathcal{O} \models Q$ if and only if there exists a query homomorphism from Q to $Can(\mathcal{O})$.*

Proof. The proof is based on Proposition 30, and follows the same line of reasoning of Proposition 17. □

The following theorem is the basis for reducing entailment of general queries to entailment of metaground queries in the context of TBox-complete ontologies.

Theorem 32. *If \mathcal{O} is TBox-complete, and $Q = q_1 \cup \dots \cup q_n$ is a general query over \mathcal{O} , then $\mathcal{O} \models Q$ if and only if there exists $i \in \{1, \dots, n\}$ and a conjunctive query $q' \in MG(q_i, \mathcal{O})$ such that $\mathcal{O} \models q'$.*

Proof. If-part. If there exists $i \in \{1, \dots, n\}$ and a conjunctive query $q' \in MG(q_i, \mathcal{O})$ such that $\mathcal{O} \models q'$, then there is a query homomorphism from q' to $Can(\mathcal{O})$. Let μ' be the instantiation of q_i that led to q' , and let μ be obtained by extending μ' with the assignment of the variables of q' given by the query homomorphism from q' to $Can(\mathcal{O})$. It is easy to see that μ is a query homomorphism from q_i to $Can(\mathcal{O})$, and therefore also a query homomorphism from Q to $Can(\mathcal{O})$, and which implies that $\mathcal{O} \models Q$.

Only-if-part. If $\mathcal{O} \models Q$, then there is a query homomorphism from Q to $Can(\mathcal{O})$, and therefore there is a query homomorphism h from q_i to $Can(\mathcal{O})$, for some $i \in \{1, \dots, n\}$. By construction of $Chase(\mathcal{O})$, all elements in \mathcal{S}_o and \mathcal{S}_v are individuals or values in $Chase(\mathcal{O})$, respectively, and therefore by the definition of $Can(\mathcal{O})$, the metavariables of q_i are mapped to elements in Exp by h . It follows that there is an instantiation q' of q_i such that there is a query homomorphism from q' to $Can(\mathcal{O})$, which implies that $\mathcal{O} \models q'$. \square

Combined with Proposition 11, the above theorem provides us with an algorithm for entailment of general queries over OWL 2 QL TBox-complete ontologies under MS. The algorithm simply checks whether there is a disjunct of the query for which there exists a metagrounding such that both the conjunction of its TBox atoms and the metaground instance query constituted by its ABox atoms are entailed by the ontology. Obviously, the algorithm can be implemented using current off-the-shelf OWL 2 QL reasoners.

It is easy to see that this leads to the following complexity results.

Theorem 33. *Entailment of general queries over TBox-complete ontologies under MS is in AC^0 w.r.t. ABox complexity, PTIME w.r.t. ontology complexity, and NP-complete w.r.t. combined complexity.*

The above theorem states that query entailment of general queries under TBox-complete ontologies has the same computational property as query entailment of metaground instance queries, i.e., usual first-order conjunctive queries, under unrestricted ontologies. Hence, it confirms the intuition that the complexity of answering general queries over unrestricted ontologies originates from the presence of negative axioms that are neither logically implied nor violated.

At the same time, the results above have a practical interest. Indeed, even though one might think that TBox-completeness is a strong assumption, limiting the applicability of the above result in practice, we argue that TBox-complete ontologies are neither unreasonable, nor expensive to be devised in practice. The reasons are twofold. On one hand, many ontologies derived from Entity-Relationship schemas, Object-oriented schemas or UML diagrams are actually TBox-complete. On the other hand, it is easy to see that, given an ontology

\mathcal{O} that is not TBox-complete and with a set S of negative axioms that are uncertain, an automated, rather inexpensive procedure can be devised, that adds to \mathcal{O} suitable facts to make the negative axioms in S violated. One can show that, in several interesting cases, such an addition achieves TBox-completeness without changing the intended meaning of the ontology \mathcal{O} .

6.2. The case of unrestricted ontologies

Let us now come back to the case of unrestricted ontologies, i.e., ontology that are not TBox-complete, and therefore may contain a nonempty set of negative axioms in $\mathcal{N}^{\mathcal{O}}$ that are uncertain. In order to solve the issue highlighted by Example 22 for such ontologies, we now introduce the notion of *completion* of an ontology, that is crucial for characterizing the sets of models of an ontology under MS. Such notion relies on the notion of *violation set* of a negative axiom. Intuitively, given an axiom α in $\mathcal{N}^{\mathcal{O}}$, the violation set of α , denoted by \mathcal{F}^{α} , is the set containing the ABox axioms that are necessary and sufficient to violate α in the ontology $\mathcal{O} \cup \mathcal{F}^{\alpha}$. The violation sets \mathcal{F}^{α} for the different types of possible axioms in $\mathcal{N}^{\mathcal{O}}$ are depicted in Table 5, where we adopt the following naming scheme (possibly including subscripts): a , r , d , and t denote, respectively, an atomic class, an atomic object property, a data property, and a datatype. $s_{\alpha}, s'_{\alpha}, s''_{\alpha}$ denote distinct entities, specific for α , that are not contained in Σ , and w_{α} denotes any literal that does not belong to Σ_{Lit} .

Negative axiom α	Violation set of α
$a_1 \sqsubseteq_c \neg a_2$	$\{a_1(s_{\alpha}), a_2(s_{\alpha})\}$
$a \sqsubseteq_c \neg \exists r$	$\{a(s_{\alpha}), r(s_{\alpha}, s'_{\alpha})\}$
$a \sqsubseteq_c \neg \exists r^{-}$	$\{a(s_{\alpha}), r(s'_{\alpha}, s_{\alpha})\}$
$a \sqsubseteq_c \neg \delta(d).t$	$\{a(s_{\alpha}), d(s_{\alpha}, w_{\alpha})\}$ with w_{α} s.t. $w_{\alpha}^{LS} \in t^{DT}$
$\exists r_1 \sqsubseteq_c \neg \exists r_2$	$\{r_1(s_{\alpha}, s'_{\alpha}), r_2(s_{\alpha}, s''_{\alpha})\}$
$\exists r_1^{-} \sqsubseteq_c \neg \exists r_2^{-}$	$\{r_1(s'_{\alpha}, s_{\alpha}), r_2(s''_{\alpha}, s_{\alpha})\}$
$\exists r_1^{-} \sqsubseteq_c \neg \exists r_2$	$\{r_1(s'_{\alpha}, s_{\alpha}), r_2(s_{\alpha}, s''_{\alpha})\}$
$\exists r \sqsubseteq_c \neg \delta(d).t$	$\{r(s_{\alpha}, s'_{\alpha}), d(s_{\alpha}, w_{\alpha})\}$ with w_{α} s.t. $w_{\alpha}^{LS} \in t^{DT}$
$\exists r^{-} \sqsubseteq_c \neg \delta(d).t$	$\{r(s'_{\alpha}, s_{\alpha}), d(s_{\alpha}, w_{\alpha})\}$ with w_{α} s.t. $w_{\alpha}^{LS} \in t^{DT}$
$r_1 \sqsubseteq_p \neg r_2$	$\{r_1(s_{\alpha}, s'_{\alpha}), r_2(s_{\alpha}, s'_{\alpha})\}$
$r_1^{-} \sqsubseteq_p \neg r_2$	$\{r_1(s'_{\alpha}, s_{\alpha}), r_2(s_{\alpha}, s'_{\alpha})\}$
$d_1 \sqsubseteq_d \neg d_2$	$\{d_1(s_{\alpha}, w_{\alpha}), d_2(s_{\alpha}, w_{\alpha})\}$
$\text{Irr}(r)$	$\{r(s_{\alpha}, s_{\alpha})\}$
$\text{Irr}(r^{-})$	$\{r(s_{\alpha}, s_{\alpha})\}$

Table 5: Violation sets of negative axioms

Definition 34. Let $\mathcal{U}^{\mathcal{O}} \subseteq \mathcal{N}^{\mathcal{O}}$ be the set of uncertain negative axioms in \mathcal{O} , and let $\sigma \subseteq \mathcal{U}^{\mathcal{O}}$. The *completion of \mathcal{O} with respect to σ* is the ontology, denoted as \mathcal{O}^{σ} , defined as follows:

$$\mathcal{O}^{\sigma} = \mathcal{O} \cup \sigma \cup \mathcal{C}^{\mathcal{U}^{\mathcal{O}} \setminus \sigma}$$

where $\mathcal{C}^{\mathcal{U}^{\mathcal{O}} \setminus \sigma}$ is the union of the violation sets of all the negative axioms in $\mathcal{U}^{\mathcal{O}}$ that are not in σ .

Intuitively, the completion of \mathcal{O} with respect to σ is the ontology that is obtained by asserting that all axioms in σ are satisfied by \mathcal{O} and that all axioms in $\mathcal{U}^{\mathcal{O}}$ that are not in σ are violated by \mathcal{O} . Note that \mathcal{O}^{σ} may be unsatisfiable even if \mathcal{O} is satisfiable.

It is straightforward to see that, by definition, for every subset σ of $\mathcal{U}^{\mathcal{O}}$, we have that \mathcal{O}^{σ} is TBox-complete.

The following example shows examples of completions of an ontology with respect to various sets of axioms.

Example 35. Consider the ontology introduced in Example 22. One can easily verify that $\mathcal{U}^{\mathcal{O}} = \{A \sqsubseteq_c \neg B, B \sqsubseteq_c \neg A\}$, as all other axioms in $\mathcal{N}^{\mathcal{O}}$ are either implied or violated by \mathcal{O} .

Let $\sigma_0 = \emptyset$, $\sigma_1 = \{A \sqsubseteq_c \neg B\}$, $\sigma_2 = \{B \sqsubseteq_c \neg A\}$, and $\sigma_3 = \{A \sqsubseteq_c \neg B, B \sqsubseteq_c \neg A\}$. The following are examples of completions of \mathcal{O} :

$$\begin{aligned} \mathcal{O}^{\sigma_0} &= \mathcal{O} \cup \{A(s), B(s), B(s'), A(s')\} \\ \mathcal{O}^{\sigma_1} &= \mathcal{O} \cup \{A \sqsubseteq_c \neg B, B(s), A(s)\} \\ \mathcal{O}^{\sigma_2} &= \mathcal{O} \cup \{B \sqsubseteq_c \neg A, A(s), B(s)\} \\ \mathcal{O}^{\sigma_3} &= \mathcal{O} \cup \{B \sqsubseteq_c \neg A, A \sqsubseteq_c \neg B\} \end{aligned}$$

where s, s' denote entities not occurring in \mathcal{O} . Note, in particular, that all completions are TBox-complete, and \mathcal{O}^{σ_1} and \mathcal{O}^{σ_2} are unsatisfiable. \square

From the definition of completion of an ontology with respect to a set of negative axioms we can easily derive the following theorem.

Theorem 36. *If Q is a general query over \mathcal{O} , then $\mathcal{O} \models Q$ if and only if $\mathcal{O}^{\sigma} \models Q$ for every $\sigma \subseteq \mathcal{U}^{\mathcal{O}}$.*

Proof. The proof is based on showing that $\text{Mod}_{MS}(\mathcal{O}) = \bigcup_{\sigma \subseteq \mathcal{U}^{\mathcal{O}}} \text{Mod}_{MS}(\mathcal{O}^{\sigma})$.

Consider $\mathcal{I} \in \text{Mod}_{MS}(\mathcal{O})$. For every $\alpha \in \mathcal{U}^{\mathcal{O}}$, either $\mathcal{I} \models \alpha$ or $\mathcal{I} \models \neg\alpha$. Let $\sigma_{\mathcal{I}}$ be the subset of axioms of $\mathcal{U}^{\mathcal{O}}$ that are satisfied in \mathcal{I} . By definition of $\sigma_{\mathcal{I}}$, $\mathcal{I} \models \mathcal{O}^{\sigma_{\mathcal{I}}}$, and therefore, $\mathcal{I} \in \text{Mod}_{MS}(\mathcal{O}^{\sigma_{\mathcal{I}}})$, thus proving that $\text{Mod}_{MS}(\mathcal{O}) \subseteq \bigcup_{\sigma \subseteq \mathcal{U}^{\mathcal{O}}} \text{Mod}_{MS}(\mathcal{O}^{\sigma})$.

For the converse, since for every $\sigma \subseteq \mathcal{U}^{\mathcal{O}}$, we have that $\mathcal{O} \subseteq \mathcal{O}^{\sigma}$, it is obvious that $\bigcup_{\sigma \subseteq \mathcal{U}^{\mathcal{O}}} \text{Mod}_{MS}(\mathcal{O}^{\sigma}) \subseteq \text{Mod}_{MS}(\mathcal{O})$. \square

Theorem 36 directly provides us with an algorithm for entailment of general queries over unrestricted ontologies. We illustrate it in the following example.

Example 37. Consider the ontology and the query discussed in Example 22, as well as the set of axioms σ_i , for $i = 1, \dots, 4$, introduced in Example 35. It is easy to see that $\mathcal{O}^{\sigma_0} \models q$ and $\mathcal{O}^{\sigma_3} \models q$. Since \mathcal{O}^{σ_1} and \mathcal{O}^{σ_2} are unsatisfiable, we also have that $\mathcal{O}^{\sigma_1} \models Q$ and $\mathcal{O}^{\sigma_2} \models q$. Since $\sigma_1, \sigma_2, \sigma_3$, and σ_4 are all possible subsets of $\mathcal{U}^{\mathcal{O}}$, we conclude that $\mathcal{O} \models q$.

The results above lead us to the following complexity results for answering general queries over unrestricted ontologies.

Theorem 38. *Query entailment under MS is in AC^0 w.r.t. ABox complexity, in coNP w.r.t. ontology complexity, and in Π_2^P w.r.t. combined complexity.*

Proof. By Theorem 33 we know that given a general query Q and a TBox-complete ontology, the problem of checking whether $\mathcal{O} \models Q$ is AC^0 with respect to ABox complexity. Clearly, since the computation of \mathcal{O}^{σ} for every subset σ of $\mathcal{U}^{\mathcal{O}}$ does not depend on the size of ABox, the ABox complexity of query answering is AC^0 also for unrestricted ontologies. As far as ontology complexity is concerned, it is easy to see that the problem is in coNP. Indeed, a nondeterministic version of the algorithm for the complement of the problem is the following: to check non-entailment of Q from \mathcal{O} , we guess a set $\sigma \subseteq \mathcal{U}^{\mathcal{O}}$, and then check whether the TBox-complete ontology \mathcal{O}^{σ} does not entails Q , by using the algorithm described in the previous subsection. Hence, once we have guessed the set $\sigma \subseteq \mathcal{U}^{\mathcal{O}}$, we perform a task that is clearly in PTIME with respect to the size of the ontology. So, there is an algorithm for the complement of the problem whose complexity is in NP, and therefore there is a coNP algorithm for checking whether $\mathcal{O}^{\sigma} \models Q$. As far as combined complexity is concerned, in the above algorithm, once we have guessed the set $\sigma \subseteq \mathcal{U}^{\mathcal{O}}$, we can use a coNP oracle for checking $\mathcal{O}^{\sigma} \not\models Q$. Since the algorithm that we have described is for the complement of the problem, it follow that there is an algorithm for checking $\mathcal{O}^{\sigma} \models Q$ whose combined complexity is coNP^{NP} . \square

By combining Corollary 24 and Corollary 26 with Theorem 38, we obtain the following.

Corollary 39. *Query entailment under MS is coNP-complete w.r.t. ontology complexity and Π_2^P -complete w.r.t. combined complexity.*

7. Conclusions

We have presented a new semantics for the OWL 2 QL ontology language, named *Metamodeling Semantics* (in short, MS), specifically tailored to capture basic features of metamodeling. In particular, MS provides the ability of interpreting metaclasses and metaproperties, i.e., classes and properties admitting classes or properties as instances, in a way that seriously takes into account the syntactic capability offered by OWL 2 of allowing a *same* entity to simultaneously play *distinct roles*, such as, for example, the role of individual and the role of class. Note that such an ability is in contrast to the standard W3C Direct Semantics for OWL 2, which resorts to interpret occurrences of the same entity name in positions of different types as denoting distinct entities.

We have studied the computational properties of reasoning under MS over ontologies expressed in OWL 2 QL and showed that while the worst-case complexity of entailment of unions of conjunctive queries is tractable for queries including only ABox atoms, the entailment of general unions of conjunctive queries is Π_2^P -complete. Also, we identified a class of ontologies, called *TBox-complete*, for which entailment of general queries is tractable.

Our work can be continued along several research directions. First, here, we did not consider inequalities, while the standard OWL 2 QL allows one to express them by means of `DifferentIndividuals`. As already mentioned, we know that inequalities may in general lead to undecidability, but we believe that by carefully restricting their occurrences within queries, entailment of general queries over TBox-complete ontologies is still tractable under MS.

Second, we studied the computational properties of reasoning under MS, focusing on OWL 2 QL, which is one of the three tractable profiles of OWL 2. It would be interesting to extend our study to the other two tractable profiles, i.e. OWL 2 EL and OWL 2 RL. In particular, considering the OWL 2 EL profile poses new challenges, since the set of expressions that can be built over the entities of an OWL 2 EL ontology is in general infinite, which implies that, in principle, relying on the metagrounding technique is not possible.

Third, general metamodeling requires to overcome the syntactic restrictions of OWL 2 QL. In particular, we are currently studying to which extent it is possible to enrich the metamodeling capabilities of the ontology language, keeping the same computational cost of reasoning. Possible extensions concern mechanisms to assert that an entity can play only some specific role, e.g. “Maurizio” is not a

class nor a property”, or to define a property as a specialization of the *instance of* relation (i.e., `rdf:type`) or the *subclass of* relation (i.e., `owl:subclassof`).

Acknowledgements

We thank Riccardo Rosati per fruitful discussions on the complexity lower bounds discussed in this paper. We gratefully acknowledge support by MIUR under the SIR project “MODEUS”, grant n. RBSI14TQHQ, and under the PRIN project “HOPE”, grant n. 2017MMJJRE, and by the H2020-EU.2.1.1 project TAILOR, grant id. 952215.

References

References

- [1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. F. Patel-Schneider (Eds.), *The Description Logic Handbook: Theory, Implementation and Applications*, Cambridge University Press, 2003.
- [2] G. Xiao, D. Calvanese, R. Kontchakov, D. Lembo, A. Poggi, R. Rosati, M. Zakharyashev, *Ontology-based data access: A survey*, in: *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, 2018, pp. 5511–5519.
- [3] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodríguez-Muro, R. Rosati, *Ontologies and databases: The DL-Lite approach*, in: S. Tessaris, E. Franconi (Eds.), *Reasoning Web. Semantic Technologies for Informations Systems – 5th Int. Summer School Tutorial Lectures (RW)*, Vol. 5689 of *Lecture Notes in Computer Science*, Springer, 2009, pp. 255–356.
- [4] B. Motik, B. C. Grau, I. Horrocks, Z. Wu, A. Fokoue, C. Lutz, et al., *Owl 2 web ontology language profiles*, W3C recommendation 27 (2009) 61.
- [5] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, R. Rosati, *Tractable reasoning and efficient query answering in description logics: The DL-Lite family*, *J. of Automated Reasoning* 39 (3) (2007) 385–429.
- [6] D. Allemang, J. Hendler, *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL*, Elsevier, 2011.

- [7] C. Atkinson, T. Kühne, The essence of multilevel metamodeling, in: International Conference on the Unified Modeling Language, Springer, 2001, pp. 19–33.
- [8] V. A. de Carvalho, J. P. A. Almeida, C. M. Fonseca, G. Guizzardi, Multi-level ontology-based conceptual modeling, *Data and Knowledge Engineering* 109 (2017) 3–24.
- [9] C. Atkinson, T. Kuhne, Model-driven development: a metamodeling foundation, *IEEE software* 20 (5) (2003) 36–41.
- [10] G. G. Wang, S. Shan, Review of metamodeling techniques in support of engineering design optimization, *Journal of Mechanical design* 129 (4) (2007) 370–380.
- [11] F. Angiulli, R. Ben-Eliyahu-Zohary, G. Ianni, L. Palopoli, Computational properties of metaquerying problems, *ACM Trans. on Computational Logic* 4 (2) (2003) 149–180.
- [12] R. Jin, W. Chen, T. W. Simpson, Comparative studies of metamodelling techniques under multiple modelling criteria, *Structural and multidisciplinary optimization* 23 (1) (2001) 1–13.
- [13] G. De Giacomo, M. Lenzerini, R. Rosati, Higher-order description logics for domain metamodeling, in: Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011, 2011.
- [14] M. Lenzerini, L. Lepore, A. Poggi, A higher-order semantics for metaquerying in OWL 2 QL, in: Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016., 2016, pp. 577–580.
- [15] M. Lenzerini, L. Lepore, A. Poggi, Answering metaqueries over hi (OWL 2 QL) ontologies, in: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016, 2016, pp. 1174–1180.
- [16] W. Chen, M. Kifer, D. S. Warren, HILOG: A foundation for higher-order logic programming, *J. Log. Program.* 15 (3) (1993) 187–230.
- [17] R. Motz, E. Rohrer, P. Severi, The description logic SHIQ with a flexible meta-modelling hierarchy, *J. Web Sem.* 35 (2015) 214–234.
- [18] M. Martinez, E. Rohrer, P. Severi, Complexity of the description logic ALCM, in: Proc. of the 15th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR), 2016, pp. 585–588.
- [19] B. Motik, On the properties of metamodeling in OWL, *J. Log. Comput.* 17 (4) (2007) 617–637.

- [20] P. Kubincová, J. Kluka, M. Homola, Expressive description logic with instantiation metamodelling, in: *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016*, Cape Town, South Africa, April 25-29, 2016., 2016, pp. 569–572.
- [21] R. Kontchakov, M. Rezk, M. Rodriguez-Muro, G. Xiao, M. Zakharyashev, Answering SPARQL queries over databases under OWL 2 QL entailment regime, in: *Proc. of the 13th Int. Semantic Web Conf. (ISWC)*, 2014, pp. 552–567.
- [22] I. Kollia, B. Glimm, Optimizing SPARQL query answering over OWL ontologies, *J. Artif. Intell. Res. (JAIR)* 48 (2013) 253–303.
- [23] B. Glimm, Y. Kazakov, I. Kollia, G. B. Stamou, Lower and upper bounds for SPARQL queries over OWL ontologies, in: *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, January 25-30, 2015, Austin, Texas, USA., 2015, pp. 109–115.
- [24] S. Bischof, M. Krötzsch, A. Polleres, S. Rudolph, Schema-agnostic query rewriting in SPARQL 1.1, in: *The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference*, Riva del Garda, Italy, October 19-23, 2014. *Proceedings, Part I*, 2014, pp. 584–600.
- [25] M. Arenas, G. Gottlob, A. Pieris, Expressive languages for querying the semantic web, in: *Proc. of the 33rd ACM SIGACT SIGMOD SIGAI Symp. on Principles of Database Systems (PODS)*, 2014, pp. 14–26.
- [26] G. Gottlob, A. Pieris, Beyond SPARQL under OWL 2 QL entailment regime: Rules to the rescue, in: *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015*, Buenos Aires, Argentina, July 25-31, 2015, 2015, pp. 2999–3007.
- [27] C. Gutierrez, D. Hernández, A. Hogan, A. Polleres, Certain answers for sparql?, in: *Proceedings of the 10th Alberto Mendelzon International Workshop on Foundations of Data Management*, Panama City, Panama, May 8-10, 2016, 2016.
- [28] S. Ahmetaj, W. Fischl, R. Pichler, M. Simkus, S. Skritek, Towards reconciling SPARQL and certain answers, in: *Proceedings of the 24th International Conference on World Wide Web, WWW 2015*, Florence, Italy, May 18-22, 2015, 2015, pp. 23–33.
- [29] V. Gutiérrez-Basulto, Y. A. Ibáñez-García, R. Kontchakov, E. V. Kostylev, Conjunctive queries with negation over dl-lite: A closer look, in: *RR-13*, 2013, pp. 109–122.
- [30] A. K. Chandra, P. M. Merlin, Optimal implementation of conjunctive queries in relational data bases, in: *Proc. of the 9th ACM Symp. on Theory of Computing (STOC)*, 1977, pp. 77–90.

- [31] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodriguez-Muro, R. Rosati, M. Ruzzi, D. F. Savo, The Mastro system for ontology-based data access, *Semantic Web J.* 2 (1) (2011) 43–53.
- [32] M. Rodriguez-Muro, D. Calvanese, Quest, an OWL 2 QL reasoner for ontology-based data access, in: *Proc. of the 9th Int. Workshop on OWL: Experiences and Directions (OWLED)*, Vol. 849 of *CEUR Electronic Workshop Proceedings*, <http://ceur-ws.org/>, 2012.

Journal Pre-proof

Appendix A. Complete proof of Proposition 16

Proof. Let $\mathcal{I} = \langle \mathcal{W}, \cdot^{\mathcal{I}} \rangle \in \text{Mod}_{MS}(\mathcal{O})$ be a model of \mathcal{O} . We define a function Ψ from $\bar{\Delta}^o$ to Δ^o and from $\bar{\Delta}^v$ to Δ^v , and we show that Ψ is an instance-based homomorphism from $\text{Can}(\mathcal{O})$ to \mathcal{I} .

Specifically, we define Ψ by induction on k , where k is the number of rules whose application leads to $\text{Chase}_k(\mathcal{O})$, by setting Ψ_k as a function on $\bar{\Delta}_k^o \cup \bar{\Delta}_k^v$, and letting $\Psi = \bigcup_{j \in \mathbb{N}} \Psi_j$. Simultaneously, we show, by induction on k , that Ψ_k is an instance-based homomorphism from $\text{Can}_k(\mathcal{O})$ to \mathcal{I} .

Base step. Let $k = 0$. First of all, we set $\Psi_0(e) = e^{\mathcal{I}}$ for every $e \in \text{Exp}$, and $\Psi_0(v) = v$, for every $v \in \Delta^v$. As for every element that is not in $\text{Exp} \cup \Delta^v$, and therefore is in $\mathcal{S}_o^0 \cup \mathcal{S}_v^0$, we observe that it appears exactly once in $\text{Chase}_0(\mathcal{O})$. Based on this observation, for every $s_1, s_2 \in \mathcal{S}_o^0$ and for every $s \in \mathcal{S}_v^0$ we set Ψ_0 as follows:

- if $s_1 = \alpha_c$, then $\Psi_0(s_1) = o$, where o is any object in $(c^{\mathcal{I}})^C$. Note that at least one such object exists in Δ^o , since by construction $\mathcal{O} \cup \{c \sqsubseteq_c \neg c\}$ is unsatisfiable;
- if $s_1 = \beta_p^1$ and $s_2 = \beta_p^2$, then $\Psi_0(s_1) = o_1$, and $\Psi_0(s_2) = o_2$, where o_1, o_2 is any pair of objects such that $\langle o_1, o_2 \rangle \in (p^{\mathcal{I}})^P$. Note that such pair must exist, since by construction $\mathcal{O} \cup \{p \sqsubseteq_p \neg p\}$ is unsatisfiable;
- if $s_1 = \gamma_d$ and $s = w_d$, then $\Psi_0(s_1) = o$ and $\Psi_0(s) = v$, where o, v is any pair such that $\langle o, v \rangle \in (d^{\mathcal{I}})^D$. Note that such pair must exist since by construction $\mathcal{O} \cup \{d \sqsubseteq_d \neg d\}$ is unsatisfiable;
- if $s_1 = \beta_p^{\text{irr}}$, then $\Psi_0(s_1) = o$, where o is any object such that $\langle o, o \rangle \in (p^{\mathcal{I}})^P$. Note that at least one such object exists in Δ^o , since by construction $\mathcal{O} \cup \{\text{Irr}(p)\}$ is unsatisfiable.

We next show that Ψ_0 is an instance-based homomorphism from $\text{Can}_0(\mathcal{O})$ to \mathcal{I} , by showing it satisfies all properties of Definition 15.

- Property (1): let e_1, e_2 be two objects in $\bar{\Delta}_0^o$ such that $e_2 \in e_1^{\bar{C}_0}$. We want to show that $\Psi_0(e_2) \in \Psi_0(e_1)^C$. Since $e_2 \in e_1^{\bar{C}_0}$, $e_1(e_2) \in \text{Chase}_0(\mathcal{O})$, where e_1 is a class expression belonging to Exp_c . Thus, by definition, $\Psi_0(e_1) = e_1^{\mathcal{I}}$. Also, by construction of $\text{Chase}_0(\mathcal{O})$, $e_1(e_2)$ either belongs to \mathcal{O} or it was added in one of the steps 1, 2(a), 2(b), 2(c), or 3 of its construction.

- Suppose that $e_1(e_2) \in \mathcal{O}$. Then, $e_2 \in \Sigma_i$ and hence $\Psi_0(e_2) = e_2^{\mathcal{I}}$. But then, since \mathcal{I} is a model of \mathcal{O} , $e_2^{\mathcal{I}} \in (e_1^{\mathcal{I}})^C$ and therefore $\Psi_0(e_2) \in \Psi_0(e_1)^C$.
 - Suppose that $e_1(e_2)$ was added in step 1. Then $e_1 = \top_c$ and $e_2 \in \Sigma_i$. It follows that $\Psi_0(e_2) = e_2^{\mathcal{I}}$. Also, since \mathcal{I} is a model of \mathcal{O} , $(\top_c^{\mathcal{I}})^C = \{o \mid o^{\mathcal{I}} = \mathbf{true}\}$ and $(e_2^{\mathcal{I}})^{\mathcal{I}} = \mathbf{true}$ (because $e_2 \in \Sigma_i$). Therefore, $\Psi_0(e_2) \in \Psi_0(e_1)^C$.
 - Suppose that $e_1(e_2)$ was added in step 2(a). Then, $e_2 = \alpha_c$, where $c \in Exp_c$, $\mathcal{O} \cup \{c \sqsubseteq_c \neg c\}$ is unsatisfiable, $\alpha_c \in \mathcal{S}_o^0$, and either $e_1 = c$ or $e_1 = \top_c$. In both cases, by definition of Ψ_0 , $\Psi_0(e_2) = o$ where $o \in (c^{\mathcal{I}})^C$ and $o^{\mathcal{I}} = \mathbf{true}$. Hence, in both cases, $\Psi_0(e_2) \in \Psi_0(e_1)^C$.
 - Suppose that $e_1(e_2)$ was added in step 2(b). Then, $e_1 = \top_c$ and either $e_2 = \beta_p^1$ or $e_2 = \beta_p^2$, where $p \in Exp_p$, $\mathcal{O} \cup \{p \sqsubseteq_c \neg p\}$ is unsatisfiable, and $\beta_p^1, \beta_p^2 \in \mathcal{S}_o^0$. Suppose that $e_2 = \beta_p^1$ (resp., $e_2 = \beta_p^2$), then by definition of Ψ_0 , $\Psi_0(e_2) = o_1$ where $\langle o_1, o_2 \rangle \in (p^{\mathcal{I}})^P$ (resp. $\langle o_2, o_1 \rangle \in (p^{\mathcal{I}})^P$), for some $o_2 \in \Delta_o$, and $o_1^{\mathcal{I}} = \mathbf{true}$. Hence, $\Psi_0(e_2) \in \Psi_0(e_1)^C$.
 - By following the same line of reasoning of the previous case, we can prove that if $e_1(e_2)$ was added in step 2(c) or step 3, then $\Psi_0(e_2) \in \Psi_0(e_1)^C$.
- Property (2): let e_1, e_2, e_3 be three objects in $\bar{\Delta}_o^0$ such that $\langle e_2, e_3 \rangle \in e_1^{\bar{P}_0}$. This means that $e_1(e_2, e_3) \in Chase_0(\mathcal{O})$, where e_1 is an object property expression belonging to Exp_p . Thus, by construction, $e_1(e_2)$ either belongs to \mathcal{O} or it was added in one of the steps 1, 2(b), or 3 of its construction. It is easy to see that, by following the same line of reasoning used for Property (1), one can show that $\langle \Psi_0(e_2), \Psi_0(e_3) \rangle \in \Psi_0(e_1)^P$.
 - Property (3): let e_1, e_2 be two objects in $\bar{\Delta}_o^0$ and v a value in $\bar{\Delta}_v^0$ such that $\langle e_2, v \rangle \in e_1^{\bar{D}_0}$. This means that $e_1(e_2, e_3) \in Chase_0(\mathcal{O})$, where e_1 is an object property expression belonging to Exp_d and e_3 is either a literal in Σ_{Lit} such that $v = e_3^{LS}$ or $e_3 = v$ is a variable in \mathcal{S}_v^0 . Thus, by definition, $\Psi_0(e_1) = e_1^{\mathcal{I}}$. Now, notice that, by construction, $e_1(e_2, e_3)$ either belongs to \mathcal{O} or it was added in one of the steps 1 or 2(c) of its construction.
 - Suppose that $e_1(e_2, e_3) \in \mathcal{O}$. Then, $e_2 \in \Sigma_i$ and $e_3 \in \Sigma_{Lit}$. Hence, $\Psi_0(e_2) = e_2^{\mathcal{I}}$ and $\Psi_0(e_3) = e_3^{LS} = v$. But then, since \mathcal{I} is a model of \mathcal{O} , $(\Psi_0(e_2), \Psi_0(e_3)) \in \Psi_0(e_1)^D$.
 - Suppose that $e_1(e_2, e_3)$ was added in step 1. Then, $e_1 = \top_d$, $e_2 \in \Sigma_i$, and $e_3 \in \Sigma_{Lit}$, and by definition of Ψ_0 , $\Psi_0(e_2) = e_2^{\mathcal{I}}$ and $\Psi_0(e_3) = e_3^{LS} = v$. But then, since \mathcal{I} is a model of \mathcal{O} , $(\top_d^{\mathcal{I}})^D = \{o \mid o^{\mathcal{I}} = \mathbf{true}\} \times \Delta_v$ and $(e_2^{\mathcal{I}})^{\mathcal{I}} = \mathbf{true}$, which proves that $\langle \Psi_0(e_2), \Psi_0(e_3) \rangle \in \Psi_0(e_1)^D$.
 - Suppose that $e_1(e_2, e_3)$ was added in step 2(c). Then, $e_1 = d$, $e_2 = \gamma_d$, and $e_3 = v = wd$, where $e_1 \in \Sigma_d$ and $\gamma_d \in \mathcal{S}_o^0$, $w_d \in \mathcal{S}_v^0$, and $\mathcal{O} \cup \{d \sqsubseteq_c \neg d\}$ is unsatisfiable. Then, by definition of Ψ_0 ,

$\Psi_0(e_2) = o$, and $\Psi_0(e_3) = w$, where $\langle o, w \rangle \in (d^{\mathcal{I}})^D$, which proves that $\langle \Psi_0(e_2), \Psi_0(e_3) \rangle \in \Psi_0(e_1)^D$.

- Property (4): let e be an object of $\bar{\Delta}_o^0$ and v a value in $\bar{\Delta}_v^0$ such that $v \in e^{\bar{\mathcal{I}}_0}$. This means that $e(e_2) \in \text{Chase}_0(\mathcal{O})$, where e_1 is a datatype expression belonging to Exp_t . In fact, by construction, $e(e_2)$ can only be added to $\text{Chase}_0(\mathcal{O})$ in step 2(c), where $e = \text{rdfs:Literal}$ and $e_2 = v = \gamma_d$, where $d \in \Sigma_d$, $\mathcal{O} \cup \{d \sqsubseteq_c \neg d\}$ is unsatisfiable, and $\gamma_d \in \mathcal{S}_o^0$. Then by definition of Ψ_0 , $\Psi_0(e_2) = w$ where $\langle o, w \rangle \in (d^{\mathcal{I}})^D$, for some $o \in \Delta_o$. Also, since \mathcal{I} is a model of \mathcal{O} , $(\text{rdfs:Literal}^{\mathcal{I}})^T = \Delta_v$ and $w \in \Delta_v$, which proves that $\Psi_0(e) \in \Psi_0(e)^T$.

Inductive step. Let k be the number of rules whose application leads to $\text{Chase}_k(\mathcal{O})$ and suppose that Ψ_k is an instance-based homomorphism from $\text{Can}_k(\mathcal{O})$ to \mathcal{I} . Then, suppose that a new chase rule is applied to get $\text{Chase}_{k+1}(\mathcal{O})$. We have to show how to define Ψ_{k+1} and prove that it is an instance-based homomorphism from $\text{Can}_{k+1}(\mathcal{O})$ to \mathcal{I} .

We define $\Psi_{k+1} = \Psi_k \cup \Psi'$, where Ψ' is the part of Ψ_{k+1} taking care of the new variables possibly introduced in $\mathcal{S}_o^{k+1} \cup \mathcal{S}_v^{k+1}$. Note that, since by applying rules **(1-5,7,9-11)** no new variable is introduced, $\Psi' \neq \emptyset$ only if the $(k+1)$ -th rule is either **(6)** or **(8)**.

- Suppose that the $(k+1)$ -th rule is **(1)**. In particular, suppose that $c_1(i_1) \in \text{Chase}_k(\mathcal{O})$, $c_1 \sqsubseteq_c c_2 \in \text{Chase}_k(\mathcal{O})$, and $c_2(i_1) \notin \text{Chase}_k(\mathcal{O})$. Then, by applying rule **(1)**, $c_2(i_1)$ is introduced in $\text{Chase}_{k+1}(\mathcal{O})$, and hence the only difference between Can_k and Can_{k+1} is that $i_1 \notin c_2^{\bar{C}^k}$, whereas $i_1 \in c_2^{\bar{C}^{k+1}}$.

Let us now consider Ψ_{k+1} . Since no new variable is introduced, $\Psi_{k+1} = \Psi_k$. Hence, on one hand, since $c_1(i_1) \in \text{Chase}_k(\mathcal{O})$ and since, by inductive hypothesis, Ψ_{k+1} is an instance-based homomorphism from Can_k to \mathcal{I} , $\Psi_{k+1}(i_1) \in (\Psi_{k+1}(c_1))^C$. On the other hand, since \mathcal{I} is a model of \mathcal{O} , we know that $\mathcal{I} \models c_1 \sqsubseteq_c c_2$, implying that $\Psi_{k+1}(i_1) \in (\Psi_{k+1}(c_2))^C$, and, hence, that Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

- Suppose that the $(k+1)$ -th rule is **(2)**. In particular, suppose that $p_1(i_1, i_2) \in \text{Chase}_k(\mathcal{O})$, $p_1 \sqsubseteq_p p_2 \in \text{Chase}_k(\mathcal{O})$, and $p_2(i_1, i_2) \notin \text{Chase}_k(\mathcal{O})$. Then, by applying rule **(2)**, $p_2(i_1, i_2)$ is introduced in $\text{Chase}_{k+1}(\mathcal{O})$, and hence the only difference between Can_k and Can_{k+1} is that $\langle i_1, i_2 \rangle \notin p_2^{\bar{P}^k}$, whereas $\langle i_1, i_2 \rangle \in p_2^{\bar{P}^{k+1}}$.

Let us now consider Ψ_{k+1} . Since no new variable is introduced, $\Psi_{k+1} = \Psi_k$. Hence, on one hand, since $p_1(i_1, i_2) \in \text{Chase}_k(\mathcal{O})$, and since, by inductive hypothesis, Ψ_k is an instance-based homomorphism from Can_k to \mathcal{I} , $\langle \Psi_{k+1}(i_1), \Psi_k(i_2) \rangle \in (\Psi_{k+1}(p_1))^P$. On the other hand, since \mathcal{I} is a model

of \mathcal{O} , we know that $\mathcal{I} \models p_1 \sqsubseteq_p p_2$, implying that $\langle \Psi_{k+1}(i_1), \Psi_{k+1}(i_2) \rangle \in (\Psi_{k+1}(p_2))^P$, and, hence, Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

- Suppose that the $(k+1)$ -th rule is **(3)**. In particular, suppose that $d_1(i_1, l) \in Chase_k(\mathcal{O})$, $d_1 \sqsubseteq_d d_2 \in Chase_k(\mathcal{O})$, and $d_2(i_1, l) \notin Chase_k(\mathcal{O})$. Then, by applying rule **(3)**, $d_2(i_1, l)$ is introduced in $Chase_{k+1}(\mathcal{O})$, and hence the Can_k and Can_{k+1} differ for the following. Let $v = l^{LS}$ if $l \in \Sigma_{Lit}$ and $v = l$, otherwise. Then, $\langle i_1, v \rangle \notin d_2^{\bar{D}^k}$ and $v \notin \rho(d_2)^{\bar{T}^k}$, whereas $\langle i_1, v \rangle \in d_2^{\bar{D}^{k+1}}$ and $l \in \rho(d_2)^{\bar{T}^{k+1}}$.

Let us now consider Ψ_{k+1} . Since no new variable is introduced, $\Psi_{k+1} = \Psi_k$. Hence, on one hand, since, by inductive hypothesis, Ψ_k is an instance-based homomorphism from Can_k to \mathcal{I} , $\langle \Psi_{k+1}(i_1), \Psi_{k+1}(v) \rangle \in (\Psi_{k+1}(d_1))^D$. On the other hand, since \mathcal{I} is a model of \mathcal{O} , we know that $\mathcal{I} \models d_1 \sqsubseteq_d d_2$, implying that $\langle \Psi_{k+1}(i_1), \Psi_{k+1}(v) \rangle \in (\Psi_k(d_2))^D$, and $\Psi_{k+1}(v) \in (\Psi_{k+1}(\rho(d_2)))^T$, and, hence, Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

- Suppose that the $(k+1)$ -th rule is **(4)**. In particular, suppose that $d(i, l) \in Chase_k(\mathcal{O})$, $\rho(d) \sqsubseteq_t t \in Chase_k(\mathcal{O})$, and $t(l) \notin Chase_k(\mathcal{O})$. Then, by applying rule **(4)**, $t(l)$ is introduced in $Chase_{k+1}(\mathcal{O})$, and hence the only difference between Can_k and Can_{k+1} is the following. Let $v = l^{LS}$ if $l \in \Sigma_{Lit}$ and $v = l$, otherwise. Then, $v \notin t^{\bar{T}^k}$, whereas $v \in t^{\bar{T}^{k+1}}$.

Let us now consider Ψ_{k+1} . Since no new variable is introduced, $\Psi_{k+1} = \Psi_k$. Hence, on one hand, since, by inductive hypothesis, Ψ_k is an instance-based homomorphism from Can_k to \mathcal{I} , $\Psi_{k+1}(v) \in (\Psi_{k+1}(\rho(d)))^T$. On the other hand, since \mathcal{I} is a model of \mathcal{O} , we know that $\mathcal{I} \models \rho(d) \sqsubseteq_t t$, implying that $\Psi_{k+1}(v) \in (\Psi_{k+1}(t))^T$, and, hence, Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

- Suppose that the $(k+1)$ -th rule is **(5)**. In particular, suppose that $\text{Ref}(p) \in Chase_k(\mathcal{O})$ and that there exists $i \in V_i \cup \mathcal{S}_o^k$ such that $p(i, i) \notin Chase_k(\mathcal{O})$. Then, by applying rule **(5)**, $p(i, i)$ is introduced in $Chase_{k+1}(\mathcal{O})$, and hence the only difference between Can_k and Can_{k+1} is that $\langle i, i \rangle \notin p^{\bar{P}^k}$, whereas $\langle i, i \rangle \in p^{\bar{P}^{k+1}}$.

Let us now consider Ψ_{k+1} . Since no new variable is introduced, $\Psi_{k+1} = \Psi_k$. Hence, on one hand, by inductive hypothesis, Ψ_{k+1} is an instance-based homomorphism from Can_k to \mathcal{I} . On the other hand, since \mathcal{I} is a model of \mathcal{O} , we know that $\mathcal{I} \models \text{Ref}(p)$, implying that $\langle \Psi_{k+1}(i), \Psi_{k+1}(i) \rangle \in (\Psi_{k+1}(p))^P$, which proves that Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

- Suppose that the $(k+1)$ -th rule is **(6)**. In particular, suppose that $\exists p.c(i_1) \in Chase_k(\mathcal{O})$, and that no i exists such that $c(i) \in Chase_k(\mathcal{O})$ and $p(i_1, i) \in Chase_k(\mathcal{O})$. Then, by applying rule **(6)**, $c(s)$ and $p(i_1, s)$ are introduced in $Chase_{k+1}(\mathcal{O})$, where s is a new object variable introduced in $Chase_{k+1}(\mathcal{O})$ such that $\mathcal{S}_o^{k+1} = \mathcal{S}_o^k \cup \{s\}$ and $\mathcal{S}_v^{k+1} = \mathcal{S}_v^k$.

Let us now consider Ψ_{k+1} . We define it by extending Ψ_k , choosing an appropriate object for $\Psi_{k+1}(s)$, and we have to prove that all the instance-based properties of Ψ_{k+1} are preserved from Can_{k+1} to \mathcal{I} . Since $\exists p.c(i_1) \in Chase_k(\mathcal{O})$, we know that $i_1 \in (\exists p.c)^{\bar{C}^k}$ and, by inductive hypothesis, $\Psi_k(i_1) \in (\Psi_k(\exists p.c))^C$. Moreover, since \mathcal{I} is a model of \mathcal{O} , there exists at least one object o in Δ^o such that:

$$\langle \Psi_k(i_1), o \rangle \in (p^{\mathcal{I}})^P \text{ and } o \in (c^{\mathcal{I}})^C \quad (*).$$

Choose one such o , and set $\Psi_{k+1}(s) = o$. Since $c(s) \in Chase_{k+1}(\mathcal{O})$, we have $s \in c^{\bar{C}^{k+1}}$, and since $p(i_1, s) \in Chase_{k+1}(\mathcal{O})$, we have $\langle i_1, s \rangle \in p^{\bar{P}^{k+1}}$. But then, from $\Psi_{k+1}(s) = o$, $\Psi_{k+1}(c) = c^{\mathcal{I}}$, $\Psi_{k+1}(p) = p^{\mathcal{I}}$, and $(*)$, it immediately follows $\Psi_{k+1}(s) \in (\Psi_{k+1}(c))^C$ and $\langle \Psi_{k+1}(i_1), \Psi_{k+1}(s) \rangle \in (\Psi_{k+1}(p))^P$, which is enough to prove that Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

- Suppose that the $(k+1)$ -th rule is **(7)**. In particular, suppose that $p(i_1, i_2) \in Chase_k(\mathcal{O})$, $c(i_2) \in Chase_k(\mathcal{O})$, and $\exists p.c(i_1) \notin Chase_k(\mathcal{O})$. Then, by applying rule **(7)**, $\exists p.c(i_1)$ is introduced in $Chase_{k+1}(\mathcal{O})$, and hence the only difference between Can_k and Can_{k+1} is that $i_1 \notin (\exists p.c)^{\bar{C}^k}$, whereas $i_1 \in (\exists p.c)^{\bar{C}^{k+1}}$.

Let us now consider Ψ_{k+1} . Since no new variable is introduced, $\Psi_{k+1} = \Psi_k$. Hence, on one hand, by inductive hypothesis, Ψ_{k+1} is an instance-based homomorphism from Can_k to \mathcal{I} . On the other hand, since \mathcal{I} is a model of \mathcal{O} , we know that $\mathcal{I} \models p(i_1, i_2)$, and $\mathcal{I} \models c(i_2)$, thus implying that $\Psi_k(i_1) \in (\Psi_k(\exists p.c))^C$, which proves that Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

- Suppose that the $(k+1)$ -th rule is **(8)**. In particular, suppose that $\delta(d).t(i) \in Chase_k(\mathcal{O})$, and that there exists no l such that $t(l) \in Chase_k(\mathcal{O})$, and $d(i, l) \in Chase_k(\mathcal{O})$. Then, by applying rule **(8)**, $t(w)$, and $d(i, w)$ are introduced in $Chase_{k+1}(\mathcal{O})$, where w is a new value variable introduced in $Chase_{k+1}(\mathcal{O})$ such that $\mathcal{S}_o^{k+1} = \mathcal{S}_o^k$ and $\mathcal{S}_v^{k+1} = \mathcal{S}_v^k \cup \{w\}$.

Let us now consider Ψ_{k+1} . We define it by extending Ψ_k , choosing an appropriate value for $\Psi_{k+1}(w)$, and we have to prove that all the instance-based properties of Ψ_{k+1} are preserved from Can_{k+1} to \mathcal{I} . Since $\delta(d).t(i) \in Chase_k(\mathcal{O})$, we know that $i \in (\delta(d).t)^{\bar{C}^k}$ and, by inductive hypothesis, $\Psi_k(i) \in (\Psi_k(\delta(d).t))^C$. Also, since \mathcal{I} is a model of \mathcal{O} , there exists at least one value v in Δ^v such that:

$$\langle \Psi_k(i), v \rangle \in (d^{\mathcal{I}})^D \text{ and } v \in (t^{\mathcal{I}})^T \quad (**).$$

Choose one such v , and set $\Psi_{k+1}(w) = v$. To show that Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} we have to show that all instance-based properties of Can_{k+1} that involve w are preserved in \mathcal{I} . By construction of $Chase_{k+1}(\mathcal{O})$, since $t(w) \in Chase_{k+1}(\mathcal{O})$, we have $w \in t^{\bar{T}^{k+1}}$, and since $d(i, w) \in Chase_{k+1}(\mathcal{O})$, we have $\langle i, w \rangle \in d^{\bar{D}^{k+1}}$.

But then, from $\Psi_{k+1}(w) = v$, $\Psi_{k+1}(t) = t^{\mathcal{I}}$, $\Psi_{k+1}(d) = d^{\mathcal{I}}$, and (**), it immediately follows $\Psi_{k+1}(w) \in (\Psi_{k+1}(t))^T$ and $\langle \Psi_{k+1}(i), \Psi_{k+1}(w) \rangle \in (\Psi_{k+1}(d))^D$, which is enough to prove that Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

- Suppose that the $(k+1)$ -th rule is **(9)**. In particular, suppose that $d(i, l) \in Chase_k(\mathcal{O})$, $t(l) \in Chase_k(\mathcal{O})$, and $\delta(d).t(i) \notin Chase_k(\mathcal{O})$. Then, by applying rule **(9)**, $p.c(i_1)$ is introduced in $Chase_{k+1}(\mathcal{O})$, and hence the only difference between Can_k and Can_{k+1} is that $i \notin (\delta(d).t)^{\bar{C}_k}$, whereas $i \in (\delta(d).t)^{\bar{C}_{k+1}}$.

Let us now consider Ψ_{k+1} . Since no new variable is introduced, $\Psi_{k+1} = \Psi_k$. Hence, on one hand, by inductive hypothesis, Ψ_{k+1} is an instance-based homomorphism from Can_k to \mathcal{I} . On the other hand, since \mathcal{I} is a model of \mathcal{O} , we know that $\mathcal{I} \models d(i, l)$, and $\mathcal{I} \models t(l)$, thus implying that $\Psi_k(i) \in (\Psi_k(\delta(d).t))^C$, which proves that Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

- Suppose that the $(k+1)$ -th rule is **(10)**. In particular, suppose that $p^-(i_1, i_2) \in Chase_k(\mathcal{O})$ and $p(i_2, i_1) \notin Chase_k(\mathcal{O})$. Then, by applying rule **(10)**, $p(i_2, i_1)$ is introduced in $Chase_{k+1}(\mathcal{O})$, and hence the only difference between Can_k and Can_{k+1} is that $\langle i_2, i_1 \rangle \notin p^{\bar{P}_k}$, whereas $\langle i_2, i_1 \rangle \in p^{\bar{P}_{k+1}}$.

Let us now consider Ψ_{k+1} . Since no new variable is introduced, $\Psi_{k+1} = \Psi_k$. Hence, on one hand, since, by inductive hypothesis, Ψ_k is an instance-based homomorphism from Can_k to \mathcal{I} , $\langle \Psi_k(i_1), \Psi_k(i_2) \rangle \in (\Psi_k(p_1^-))^P$. On the other hand, since \mathcal{I} is a model of \mathcal{O} , we know that $((p^-)^{\mathcal{I}})^P = ((p^{\mathcal{I}})^P)^{-1}$, implying that $\langle \Psi_k(i_2), \Psi_k(i_1) \rangle \in (\Psi_k(p))^P$, and, hence, Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

- Suppose that the $(k+1)$ -th rule is **(11)**. In particular, suppose that $p(i_1, i_2) \in Chase_k(\mathcal{O})$ and $p^-(i_2, i_1) \notin Chase_k(\mathcal{O})$. Then, by applying rule **(11)**, $p^-(i_2, i_1)$ is introduced in $Chase_{k+1}(\mathcal{O})$, and hence the only difference between Can_k and Can_{k+1} is that $\langle i_2, i_1 \rangle \notin (p^-)^{\bar{P}_k}$, whereas $\langle i_2, i_1 \rangle \in (p^-)^{\bar{P}_{k+1}}$.

Let us now consider Ψ_{k+1} . Since no new variable is introduced, $\Psi_{k+1} = \Psi_k$. Hence, on one hand, since, by inductive hypothesis, Ψ_k is an instance-based homomorphism from Can_k to \mathcal{I} , $\langle \Psi_k(i_1), \Psi_k(i_2) \rangle \in (\Psi_k(p_1))^P$. On the other hand, since \mathcal{I} is a model of \mathcal{O} , we know that $((p^-)^{\mathcal{I}})^P = ((p^{\mathcal{I}})^P)^{-1}$, implying that $\langle \Psi_k(i_2), \Psi_k(i_1) \rangle \in (\Psi_k(p^-))^P$, and, hence, Ψ_{k+1} is an instance-based homomorphism from Can_{k+1} to \mathcal{I} .

□

AUTHOR DECLARATION

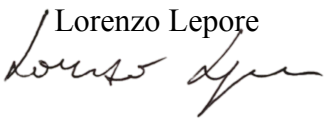
We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

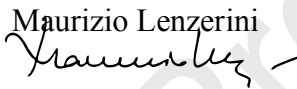
We confirm that the manuscript has been read and approved by all named authors and that there are no other persons who satisfied the criteria for authorship but are not listed. We further confirm that the order of authors listed in the manuscript has been approved by all of us. We confirm that we have given due consideration to the protection of intellectual property associated with this work and that there are no impediments to publication, including the timing of publication, with respect to intellectual property. In so doing we confirm that we have followed the regulations of our institutions concerning intellectual property.

We understand that the Corresponding Author is the sole contact for the Editorial process (including Editorial Manager and direct communications with the office). He/she is responsible for communicating with the other authors about progress, submissions of revisions and final approval of proofs. We confirm that we have provided a current, correct email address which is accessible by the Corresponding Author and which has been configured to accept email from (antonella.poggi@uniroma1.it).

Rome, 05/11/2019.

Signed by all authors as follows:

Lorenzo Lepore


Maurizio Lenzerini


Antonella Poggi
