Answering MetaQueries over RDFS Ontologies under the SPARQL Metamodeling Semantics Entailment Regime

Diego Calvanese[1][0000-0001-5174-9693], Gianluca Cima[2][0000-0003-1783-5605], Julien Corman[1][0000-0003-4386-5218], Roberto Maria Delfino[2][0000-0002-5492-5290], Maurizio Lenzerini[2][0000-0003-2875-6187], Lorenzo Marconi[2][0000-0001-9633-8476], Antonella Poggi[2][0000-0002-4030-3458], and Ognjen Savkovic[1][0000-0002-9141-3008]

[1] Free University of Bozen-Bolzano lastname@inf.unibz.it
[2] Sapienza University of Rome lastname@diag.uniroma1.it

Abstract. We study (unions of) conjunctive metaqueries over ontologies expressed in the Description Logics counterpart of RDFS equipped with metamodeling capabilities. This generalization to metaquerying goes beyond the extensively investigated query answering problem over ontologies, as now queries may also contain metavariables and TBox atoms, aligning with the classical "Ask-and-Tell" paradigm in knowledge representation. For the semantics of queries, we adopt the SPARQL Metamodeling Semantics Entailment Regime, a recently introduced regime that generalizes the commonly used SPARQL Direct Semantics Entailment Regime by relaxing the typing constraints. This new regime supports a more flexible querying mechanism over ontologies involving metaclasses and metaproperties, as query variables can appear in positions of different types (i.e. as individual, class, or property). As a result, it enables the formulation of meaningful metaqueries related to metamodeling. We show that this query answering task can be reduced to the evaluation of a linear Datalog program. This Datalog encoding allows us to establish tight complexity bounds for the associated decision problem, in combined complexity, ontology complexity and data complexity.

Keywords: Metamodeling · Metaquerying · RDFS · SPARQL Entailment Regime · Datalog · Computational Complexity.

1 INTRODUCTION

Description Logic (DL) ontologies enable the structured and symbolic representation of knowledge pertaining to a specific domain of interest. A DL ontology (or simply ontology in what follows) typically consists of two complementary components: an intensional part and an extensional part. The first one, called

TBox, is a set of axioms expressed in some formal language that define the intensional knowledge of the modeled domain. The latter, called ABox, is a set of logical ground atoms that provides the extensional knowledge of the domain.

Query answering is arguably one of the most studied tasks for extracting information from ontologies, where a query is a declarative expression defining the information an end user wants to retrieve. But with very few exceptions (see related work in Section 2), this vast body of literature on ontology-mediated query answering [6] and ontology-based data access [35] does not consider *metamodeling* and *metaquerying*.

As pointed out in [24], metamodeling (also known as *multi-level modeling* [4]) allows to model domain knowledge by using *metaclasses* and *metaproperties* in the ontology, which are, respectively, classes and properties that can predicate over classes and properties (in addition to individuals).

Example 1. According to scientific classification, living beings are categorized in progressively more specific taxonomic ranks, namely domains (the most generic), kingdoms, phyla, classes, orders, families, genera, and finally, species (the most specific). Every species (and the same holds for the other ranks) can be thought of as a set of individuals. Therefore, it is natural to treat a concept like "species", "genus", etc. as a metaclass (in this example, a class whose instances are in turn classes). In the following, we partially model the above domain of interest through the ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$, with TBox $\mathcal{T} = \{\mathsf{Warthog} \sqsubseteq_c \mathsf{Animal}, \mathsf{Lion} \sqsubseteq_c \mathsf{Animal}, \mathsf{predatorOf} \sqsubseteq_p \mathsf{preyOf}^-\}$, where $\sqsubseteq_c$ and $\sqsubseteq_p$ refer to inclusions between classes and properties, respectively, and ABox $\mathcal{A} = \{\mathsf{predatorOf}(\mathsf{Scar}, \mathsf{Pumbaa}), \mathsf{Warthog}(\mathsf{Pumbaa}), \mathsf{Lion}(\mathsf{Scar}), \mathsf{Kingdom}(\mathsf{Animal}), \mathsf{Species}(\mathsf{Warthog}), \mathsf{Species}(\mathsf{Lion})\}$. Note that Lion and Warthog are not only used as predicates (e.g. in $\mathsf{Lion} \sqsubseteq_c \mathsf{Animal}$ or $\mathsf{Lion}(\mathsf{Scar})$), but are also predicated over (e.g. in $\mathsf{Species}(\mathsf{Lion})$). Note also that the statements that predicate over classes (e.g. $\mathsf{Species}(\mathsf{Lion})$) are part of the ABox.

The current W3C Recommendation for representing ontologies on the Semantic Web is `OWL 2`, and the de-facto query language for expressing queries over `OWL 2` ontologies is `SPARQL 1.1` [20] (in what follows, simply `SPARQL`). The core construct of `SPARQL` queries is the *basic graph pattern* (*BGP*), which can be seen as a conjunction of atoms. Each atom in a BGP corresponds to an axiom expressible in the language of the queried ontology, but its terms may include variables other than `IRI`s. This is coherent with the classical knowledge representation *Ask-and-Tell* framework [25], whose guiding principle is to "ask anything that can be told to a knowledge base". As previously noted, this differs from the majority of works in the query answering literature, where queries can typically contain only ABox axioms and variables are restricted to individual positions.

Example 2. In the scenario modeled in Example 1, one might ask which taxa Scar's prey belongs to, limited to the animal kingdom (and sub-taxa thereof):

$$q_1(X) = \exists Y.\mathsf{preyOf}(Y, \mathsf{Scar}) \wedge X(Y) \wedge X \sqsubseteq_c \mathsf{Animal}$$

Interestingly, the syntax of the OWL 2 profiles [28], such as OWL 2 QL, provide support for metamodeling through OWL 2 *punning*, which is the capability of an IRI (*Internationalized Resource Identifier*) to play simultaneously the role of an individual, class, or property. Although such languages allow for metamodeling at least from a syntactic point of view, the official semantics, called *Direct Semantics* [29], imposes that occurrences of the same IRI in different positions are treated as if they were occurrences of different IRIs.

As a result of this choice, the *SPARQL Direct Semantics Entailment Regime* (*DSER*) [17] (the most typical SPARQL Entailment Regime for OWL 2 QL ontologies) enforces the so-called *typing constraints*, which forbid a variable in a SPARQL query to appear both in an individual position (i.e. as an argument of a predicate atom) and in predicate position (i.e. as a class or a property). This means that, under DSER, we cannot join variables denoting classes with those denoting individuals, thus preventing the specification of interesting metaqueries related to metamodeling.

Example 3. According to DSER, the set of answers to $q_1(X)$ of Example 2 over the ontology $\mathcal{O}$ of Example 1 is $\{\mathsf{Warthog}, \mathsf{Animal}\}$. To filter out taxa that may be too generic (like Animal), one can restrict the assignments for X to species:

$$q_2(X) = \exists Y.\mathsf{preyOf}(Y, \mathsf{Scar}) \wedge X(Y) \ \wedge X \sqsubseteq_c \mathsf{Animal} \wedge \mathsf{Species}(X)$$

Unfortunately, such a query is not allowed in DSER, because the typing constraints prohibit X from appearing both in class and in individual position.

To remedy this limitation, [23] introduced a novel semantics, called *Metamodeling Semantics* (*MS*) in [24], which relies on the notion of interpretation structures with a second-order flavor, and is a conservative extension of the Direct Semantics. Building on this new semantics, [11] proposed the *SPARQL Metamodeling Semantics Entailment Regime* (*MSER*), a novel SPARQL Entailment Regime that generalizes DSER by *(i)* adopting the MS for OWL 2 QL ontologies and *(ii)* relaxing the typing constraints.

Example 4. According to MSER, the set of answers to the query $q_2(X)$ of Example 3 over the ontology $\mathcal{O}$ of Example 1 is $\{\mathsf{Warthog}\}$.

In this paper, we focus on the fragment of OWL 2 QL corresponding to the DL *DL-Lite*$_{\text{RDFS}}$ [34] extended with *OWL 2 punning*, which we call *MDL-Lite*$_{\text{RDFS}}$. As an example, the ontology illustrated in Example 1 is a *MDL-Lite*$_{\text{RDFS}}$ ontology. We recall that *DL-Lite*$_{\text{RDFS}}$ corresponds to the DL counterpart of RDFS, and, as already pointed out in [34], it is very similar to the DL *RDFS(DL)* [14]. This popular DL has been the subject of several recent studies, including tasks related to abstraction [10,13], as well as query answering over ontologies formulated in this language [34] and slight extensions of it [12,9].

To the best of our knowledge, however, the topic of metaqueries over ontologies expressed in this language equipped with metamodeling capabilities (i.e. over *MDL-Lite*$_{\text{RDFS}}$ ontologies) is still unexplored. In this paper, we focus on unions of conjunctive metaqueries (UCMQs), which correspond to unions of SPARQL Basic Graph Patterns. Both queries in Example 2 and Example 3 are UCMQs.

Contributions. Our main contributions are twofold. First, we show that the problem of answering UCMQs over *MDL-Lite*$_{\texttt{RDFS}}$ ontologies under MSER can be reduced to the evaluation of a Datalog program. In particular, we provide a fixed set of Datalog rules that simulates inference under MSER and, for an *MDL-Lite*$_{\texttt{RDFS}}$ ontology $\mathcal{O}$ and UCMQ q, we show how to encode $\mathcal{O}$ as a set Datalog facts and q as another set of rules, such that the evaluation of the resulting program returns exactly the answers to q over $\mathcal{O}$.

Second, we exploit this program to provide tight complexity bounds for the decision problem associated to answering a UCMQ over an *MDL-Lite*$_{\texttt{RDFS}}$ ontology. Specifically, we show that this problem is NP-complete in combined and query complexity (which is already the complexity of conjunctive query answering [1]), NLogSpace-complete in ontology and TBox complexity, and in AC^0 in ABox (a.k.a. data) complexity.

We conclude this section by mentioning that a Datalog reduction from the problem of answering UCMQs over `OWL 2 QL` ontologies under MSER was already provided in [11]. So, in principle, since *MDL-Lite*$_{\texttt{RDFS}}$ is a fragment of `OWL 2 QL`, we could use that encoding. However, two observations are in order: *(i)* to the best of our knowledge, there is no proof of completeness for the Datalog program defined in [11]; and *(ii)* tight complexity bounds were not provided in this work.

2 RELATED WORK

Metamodeling techniques have been applied in various domains, including model-driven development [5] and conceptual modeling [7]. In the Knowledge Representation and Reasoning research area, several works [27,30,26,22] investigate the satisfiability problem (and, in some cases, also the subsumption problem) of ontologies expressed in languages that support a higher-order syntax. Notably, inspired by HiLog [8], [27] proposes a metamodeling semantics for `OWL 2 DL` enriched with the metamodeling features of `OWL 2`. However, these works do not consider query answering.

The problem of answering `SPARQL` queries over ontologies expressed in (fragments of) `OWL 2` has been an active research topic in recent years [21,18,19,2,31,3]. However, although these works address the case of metaqueries thanks to `SPARQL`'s native support for them, the ontology languages adopted in these papers lack metamodeling capabilities.

Closer to the present paper are the investigations on answering metaqueries over `OWL 2 QL` ontologies in [16,11,24]. Similarly to [27], the semantics for ontologies considered in [16] is inspired by HiLog and thus differs from the MS [24] adopted in this paper. On the other hand, [11] uses the same semantics for ontologies (MS) and for queries (MSER) adopted in this paper. As already said in the introduction, this work provides a Datalog encoding for answering metaqueries over `OWL 2 QL` ontologies under MSER. Very recently, this encoding has been implemented in various Datalog tools and its practical effectiveness has been evaluated over the LUBM and MODEUS ontologies [33,32]. However, we remind the reader that the exact complexity of the associated query answering

problem remains open, as [11] only establishes upper bounds derived from the complexity of the evaluation of Datalog queries.

Finally, we mention that [24] also addresses the problem of answering UCMQs over OWL 2 QL ontologies. However, the semantics adopted for queries is a variant of MSER that is more closely aligned with classical logic. In particular, under the semantics for queries adopted in [24], a Boolean UCMQ q is true over an OWL 2 QL ontology $\mathcal{O}$ if, for every model $\mathcal{I}$ of $\mathcal{O}$, there exists an assignment for the existential variables of q that makes the resulting quantifier-free sentence true in $\mathcal{I}$. However, this assignment may differ from one model to another. In contrast, under MSER, q is true over $\mathcal{O}$ if a *same* assignment satisfies this condition in every model of $\mathcal{O}$. In other words, the semantics for queries adopted in [24] assigns to the union ($\vee$) and the existential quantification ($\exists$) operators the classical logical meaning, while MSER is closer in spirit to DSER (the official W3C semantics for interpreting queries over OWL 2 ontologies). Indeed, we recall that MSER extends DSER to properly handle both metamodeling (by adopting the MS) and metaquerying (by removing the typing constraints) with SPARQL.

3 PRELIMINARIES

We assume basic knowledge about first-order logic, databases and Description Logics. In what follows, we refer to countably infinite and disjoint sets $\mathbb{V}$ and $\mathbb{I}$ of symbols, used for variables and *Internationalized Resource Identifiers* (*IRIs*), respectively. We also use $\bar{t}$ to denote a tuple of symbols and, when clear from the context, we may use the same notation to refer to the set of symbols occurring in this tuple.

An *MDL-Lite*$_{\texttt{RDFS}}$ ontology is a set of axioms, expressed in *MDL-Lite*$_{\texttt{RDFS}}$, i.e., the language that extends *DL-Lite*$_{\texttt{RDFS}}$ [34] by allowing the same expression to denote an individual, a class, or a property. As usual in DLs, an ontology $\mathcal{O}$ can be seen as a pair $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ and $\mathcal{A}$, respectively called *TBox* and *ABox*, are sets of intensional and extensional assertions called *axioms*, expressing knowledge about elements of the domain of interest.

Syntactically, the set of *well-formed expressions over* $\mathbb{I}$ $Exp(\mathbb{I})$ is the union of the set of *role expressions* $Exp_p(\mathbb{I}) = \mathbb{I} \cup \{e^- \mid e \in \mathbb{I}\}$, and the set of *concept expressions* $Exp_c(\mathbb{I}) = \mathbb{I} \cup \{\exists e \mid e \in Exp_p(\mathbb{I})\}$. An *MDL-Lite*$_{\texttt{RDFS}}$ *TBox* $\mathcal{T}$ is a finite set of axioms of the following form:

$$b \sqsubseteq_c a \quad \text{(concept inclusion)} \qquad\qquad p_1 \sqsubseteq_p p_2 \quad \text{(role inclusion)}$$

where $a \in \mathbb{I}$ is an IRI, $b \in Exp_c(\mathbb{I})$ is a concept expression, and $p_1, p_2 \in Exp_p(\mathbb{I})$ are role expressions. An *MDL-Lite*$_{\texttt{RDFS}}$ *ABox* $\mathcal{A}$ is a finite set of axioms of the form $a(e)$ or $r(e_1, e_2)$, with $e, e_1, e_2, a, r \in \mathbb{I}$. We use $\Sigma(\mathcal{O}) \subset \mathbb{I}$ to denote the set of IRIs occurring in and ontology $\mathcal{O}$ (also called the *signature* of $\mathcal{O}$).

The semantics of *MDL-Lite*$_{\texttt{RDFS}}$ ontologies is based on the so-called *Metamodeling Semantics* (*MS*) [24], which we tailor here for this language. An *MS-interpretation* (or simply *interpretation*) $\mathcal{I}$ is a tuple $\mathcal{I} = \langle \Delta, \cdot^C, \cdot^P, \cdot^{\mathcal{I}} \rangle$, where:

- Δ is a non-empty set of *objects*, called the *domain* of $\mathcal{I}$;
- $\cdot^C \colon \Delta \to \mathcal{P}(\Delta)$ is a partial function mapping an object o to a set o^C of objects;
- $\cdot^P \colon \Delta \to \mathcal{P}(\Delta \times \Delta)$ is another partial function mapping an object o to a binary relation o^P over Δ;
- $\cdot^{\mathcal{I}} \colon \mathit{Exp}(\mathbb{I}) \to \Delta$ is a total function, called *interpretation function*, mapping each expression in $\mathit{Exp}(\mathbb{I})$ to an object $o^{\mathcal{I}} \in \Delta$ such that *(i)* for every $p \in \mathbb{I}$, if $\cdot^P$ is defined for $p^{\mathcal{I}}$, then $((p^-)^{\mathcal{I}})^P = \{(o', o) \mid (o, o') \in (p^{\mathcal{I}})^P\}$; *(ii)* for every $p \in \mathit{Exp}_p(\mathbb{I})$, if $\cdot^P$ is defined for $p^{\mathcal{I}}$, then $((\exists p)^{\mathcal{I}})^C = \{o \mid \exists o'.(o, o') \in (p^{\mathcal{I}})^P\}$.

In the above definition, $\cdot^C$ (resp. $\cdot^P$) applied to a concept o forms the extension of o as a concept (resp. role). If the function is undefined for o, then o is intuitively not a concept (resp. role) in the world represented by $\mathcal{I}$.

We now provide the notion of satisfaction of an *MDL-Lite*$_{\texttt{RDFS}}$ axiom ϕ with respect to an MS-interpretation $\mathcal{I} = \langle \Delta, \cdot^C, \cdot^P, \cdot^{\mathcal{I}} \rangle$ (denoted by $\mathcal{I} \models \phi$):

- if $\phi = b \sqsubseteq_c a$, then $\mathcal{I} \models \phi$ if $\cdot^C$ is defined for $b^{\mathcal{I}}$ and $a^{\mathcal{I}}$, and $(b^{\mathcal{I}})^C \subseteq (a^{\mathcal{I}})^C$;
- if $\phi = p \sqsubseteq_p r$, then $\mathcal{I} \models \phi$ if $\cdot^P$ is defined for $p^{\mathcal{I}}$ and $r^{\mathcal{I}}$, and $(p^{\mathcal{I}})^P \subseteq (r^{\mathcal{I}})^P$;
- if $\phi = a(e)$, then $\mathcal{I} \models \phi$ if $\cdot^C$ is defined for $a^{\mathcal{I}}$, and $e^{\mathcal{I}} \in (a^{\mathcal{I}})^C$;
- if $\phi = r(e_1, e_2)$, then $\mathcal{I} \models \phi$ if $\cdot^P$ is defined for $r^{\mathcal{I}}$, and $(e_1^{\mathcal{I}}, e_2^{\mathcal{I}}) \in (r^{\mathcal{I}})^P$.

We use $\mathcal{I} \models \mathcal{T}$ (resp. $\mathcal{I} \models \mathcal{A}$) to denote the fact that $\mathcal{I}$ satisfies all the axioms occurring in a TBox $\mathcal{T}$ (resp. ABox $\mathcal{A}$). We say that $\mathcal{I}$ is a *model* of an *MDL-Lite*$_{\texttt{RDFS}}$ ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ (denoted by $\mathcal{I} \models \mathcal{O}$) if $\mathcal{I} \models \mathcal{T}$ and $\mathcal{I} \models \mathcal{A}$. We may also write $\mathcal{O} \models \phi$ to indicate that $\mathcal{I} \models \phi$ holds for every model $\mathcal{I}$ of $\mathcal{O}$.

A *query atom* is an expression having the form of an *MDL-Lite*$_{\texttt{RDFS}}$ axiom, with the difference that variables may also occur among its terms. A *Conjunctive MetaQuery (CMQ)* $q(\bar{X}; \bar{Y})$ over an ontology $\mathcal{O}$ is an expression of the form $\exists \bar{Y}.\varphi$, where $\bar{X}$ and $\bar{Y}$ are tuples of variables in $\mathbb{V}$, and φ is a conjunction of atoms with variables in $\bar{X} \cup \bar{Y}$. When convenient, we simply write $q(\bar{X})$ or q, and we call *arity* of $q(\bar{X})$ the arity of $\bar{X}$. A *Union of CMQs (UCMQ)* over $\mathcal{O}$ is a finite disjunction of CMQs over $\mathcal{O}$ with the same arity, called its disjuncts. We say that a query is *Boolean* if it has arity 0.

For the semantics of (U)CMQs, we adopt the ***SPARQL*** *Metamodeling Semantics Entailment Regime* (*MSER*) [11], which defines the answers to a query similarly to the `SPARQL` Direct Semantics Entailment Regime (DSER) [17], except that it uses MS for logical entailment.

Let $q(\bar{X}) = \exists \bar{Y}.\alpha_1 \wedge \cdots \wedge \alpha_n$ be a CMQ, and let $\bar{d}$ and $\bar{e}$ be two tuples of `IRI`s of the same arity as $\bar{X}$ and $\bar{Y}$, respectively. We denote by $q(\bar{X}/\bar{d}; \bar{Y}/\bar{e})$ the conjunction of ground atoms $\phi_1, \ldots, \phi_n$, where each ϕ_i is obtained from α_i by replacing each variable $X \in \bar{X}$ and $Y \in \bar{Y}$ with the corresponding `IRI` in $\bar{d}$ and $\bar{e}$, respectively. Note that, by construction, every ϕ_i in $q(\bar{X}/\bar{d}; \bar{Y}/\bar{e})$ is an *MDL-Lite*$_{\texttt{RDFS}}$ axiom. Then, we say that a tuple $\bar{d}_a$ of `IRI`s is an *answer to* $q(\bar{X})$ *over* $\mathcal{O}$ if there is a tuple $\bar{e}_a$ of `IRI`s such that $\mathcal{O} \models \phi_i$ holds for every $\phi_i \in q(\bar{X}/\bar{d}_a; \bar{Y}/\bar{e}_a)$. If q is a UCMQ, then $\bar{d}_a$ is an answer to q over $\mathcal{O}$ if it is an

answer to any disjunct of q. For a Boolean UCMQ $q()$, we say that q *is entailed by* $\mathcal{O}$, written $\mathcal{O} \models q$, if the empty tuple is an answer to q over $\mathcal{O}$.

Hereafter, we will use the terms *class* and *property* in place of the DL notions of "concept" and "role", respectively, as these best align with the knowledge graph terminology.

Throughout this paper, we will also use Datalog. A Datalog *atom* may have constants and/or variables as arguments. For instance, the atom $\texttt{subClass}(a, X)$ contains a constant a and a variable X. A Datalog *fact* is an atom without variables, while a Datalog *rule* is an expression of the form $\alpha_0 \leftarrow \alpha_1, \ldots, \alpha_m$, where $m \geq 1$, each α_i is an atom and all variables in α_0 appear in some α_j, for $j \geq 1$. The *head* of the rule is α_0 and $\alpha_1, \ldots, \alpha_m$ is its *body*. A Datalog *program* is a finite set of facts and rules. In a Datalog program, a predicate is called *intensional* if it appears in the head of some rule(s), and *extensional* otherwise. A Datalog program is called *linear* if the body of each rule contains at most one intensional predicate. We use a standard characterization of the evaluation of a Datalog program Π (see, e.g., [1]), based on the notion of *least model* Π^* of Π, i.e. the least (w.r.t. set inclusion) set of facts that contains all facts in Π and satisfies all its rules.

4 DATALOG ENCODING

In this section, we introduce a linear Datalog program Π that encodes an ontology $\mathcal{O}$ and has linear size. We show that the least model Π^* of Π coincides (modulo some simple encoding of DL axioms as Datalog facts) with the set of formulas that are logically implied by $\mathcal{O}$. This will allow us to derive results about the complexity of query answering, in Section 5.

Encoding axioms. Before defining the program Π, we need to clarify how we encode DL axioms as Datalog facts (i.e. grounded atoms). First, to each expression $\eta \in \mathit{Exp}(\mathbb{I}) \setminus \mathbb{I}$ (meaning that η is a composite role or concept), we associate a Datalog fact $\mathsf{expr}(\eta)$ that specifies how η is constructed. Precisely, we define $\mathsf{expr}(r^-)$ and $\mathsf{expr}(\exists r)$ as the Datalog facts $\texttt{inverse}(r^-, r)$ and $\texttt{domain}(\exists r, r)$, respectively. Then, a *MDL-Lite*$_{\texttt{RDFS}}$ axiom α is encoded as the set

$$\mathsf{enc}(\alpha) = \{\mathsf{atom}(\alpha)\} \cup \{\mathsf{expr}(\eta) \mid \eta \text{ appears in } \alpha\}$$

of Datalog facts, where $\mathsf{atom}(\alpha)$ is defined as follows:

MDL-Lite$_{\texttt{RDFS}}$ axiom α	Datalog fact $\mathsf{atom}(\alpha)$
$a(e)$	$\texttt{classInst}(e, a)$
$r(e_1, e_2)$	$\texttt{propInst}(e_1, e_2, r)$
$a \sqsubseteq_c b$	$\texttt{subClass}(a, b)$
$r \sqsubseteq_p s$	$\texttt{subProp}(r, s)$

Soundness and completeness. With this encoding, we can clarify what it means for our program Π to be correct, namely

$$\mathcal{O} \models \alpha \quad \text{iff} \quad \mathsf{enc}(\alpha) \subseteq \Pi^*,$$

for any *MDL-Lite*$_{\texttt{RDFS}}$ axiom α over $\Sigma(\mathcal{O})$, where Π^* is the least model of Π.

The rules of our program Π (which we will introduce below) all have an intuitive meaning. For instance, the first rule of Figure 1 states that if a is a subclass of b and e is an instance a, then e is also an instance of b. In other words, these rules only derive (atoms representing) axioms that are indeed implied by $\mathcal{O}$. As a consequence, correctness (as we just defined it) trivially holds in one direction (*soundness*), namely

$$\mathsf{enc}(\alpha) \subseteq \Pi^* \text{ implies } \mathcal{O} \models \alpha$$

The proof of the converse property (*completeness*) is more involved:

$$\mathcal{O} \models \alpha \text{ implies } \mathsf{enc}(\alpha) \subseteq \Pi^*$$

This property means that every *MDL-Lite*$_{\texttt{RDFS}}$ axiom (semantically) implied by $\mathcal{O}$ can be derived via the Datalog program Π. So throughout this section, while introducing Π, we will also prove that it is complete.

ABox axioms. We start by introducing a sub-program Π^{ABox} of Π, in charge of inferring all ABox axioms implied by $\mathcal{O}$. In order to ensure that Π is linear, we use two predicates `subClassExt` and `subPropExt` to encode the TBox axioms that are (syntactically) present in $\mathcal{T}$. Precisely, for an axiom α, we use $\mathsf{encExt}(\alpha)$ for the set of atoms identical to $\mathsf{enc}(\alpha)$, but where the predicate `subClass` (resp. `subProp`), if present, is replaced with `subClassExt` (resp. `subPropExt`).

Our subprogram Π^{ABox} includes the set $\mathcal{D}_\mathcal{O}$ of Datalog facts that encode the axioms of $\mathcal{O}$, i.e.:

$$\mathcal{D}_\mathcal{O} = \bigcup_{\alpha \in \mathcal{O}} \mathsf{encExt}(\alpha) \subseteq \Pi^{ABox}$$

In addition to $\mathcal{D}_\mathcal{O}$, the program Π^{ABox} contains the 7 rules listed in Figure 1.

Next, we observe that $\mathsf{enc}(\alpha) = \{\mathsf{atom}(\alpha)\}$ for any ABox axiom α, so that completeness for this fragment can be reformulated as

$$\mathcal{O} \models \alpha \text{ implies } \mathsf{atom}(\alpha) \in (\Pi^{ABox})^* \tag{1}$$

To show that this property holds, we focus on the set $(\Pi^{ABox})^*_{\mathsf{facts}}$ of Datalog atoms that represent ABox axioms (a.k.a. "facts") in the least model of this program, i.e.:

$$(\Pi^{ABox})^*_{\mathsf{facts}} = \{\alpha \mid \mathsf{atom}(\alpha) \in (\Pi^{ABox})^* \text{ and } \alpha \text{ is an ABox axiom}\}$$

$$\begin{aligned}
\texttt{classInst}(E, B) &\leftarrow \texttt{classInst}(E, A), \texttt{subClassExt}(A, B)\\
\texttt{classInst}(E_1, A) &\leftarrow \texttt{propInst}(E_1, E_2, R),\ \texttt{domain}(W, R),\ \texttt{subClassExt}(W, A)\\
\texttt{classInst}(E_2, A) &\leftarrow \texttt{propInst}(E_1, E_2, R),\ \texttt{inverse}(S, R), \texttt{domain}(W, S),\ \texttt{subClassExt}(W, A)\\
\texttt{propInst}(E_1, E_2, S) &\leftarrow \texttt{propInst}(E_1, E_2, R),\ \texttt{subPropExt}(R, S)\\
\texttt{propInst}(E_2, E_1, S) &\leftarrow \texttt{propInst}(E_1, E_2, R),\ \texttt{inverse}(R', R),\ \texttt{subPropExt}(R', S)\\
\texttt{propInst}(E_2, E_1, S) &\leftarrow \texttt{propInst}(E_1, E_2, R),\ \texttt{inverse}(S', S),\ \texttt{subPropExt}(R, S')\\
\texttt{propInst}(E_1, E_2, S) &\leftarrow \texttt{propInst}(E_1, E_2, R),\ \texttt{inverse}(R', R),\ \texttt{inverse}(S', S),\ \texttt{subPropExt}(R', S')
\end{aligned}$$

Fig. 1. Rules of the program Π^{ABox}, in charge of inferring ABox axioms

This set $(\Pi^{ABox})^*_{\mathsf{facts}}$ can intuitively be viewed as a first-order structure. Or more formally, under the MS, as any interpretation $\mathcal{I} = \langle \Delta, \cdot^C, \cdot^P, \cdot^{\mathcal{I}} \rangle$ for $\Sigma(\mathcal{O})$ such that *(i)* $\cdot^{\mathcal{I}}$ is injective, *(ii)* $o \in (a^{\mathcal{I}})^C$ iff $a(e) \in (\Pi)^*_{\mathsf{facts}}$ and $e^{\mathcal{I}} = o$, and *(iii)* $(o_1, o_2) \in (r^{\mathcal{I}})^P$ iff $r(e_1, r_2) \in (\Pi)^*_{\mathsf{facts}}, e_1^{\mathcal{I}} = o_1$ and $e_2^{\mathcal{I}} = o_2$.

By construction, for an ABox axiom α,

$$\mathcal{I} \models \alpha \text{ implies } \mathsf{atom}(\alpha) \in (\Pi^{ABox})^* \tag{2}$$

In addition, we will show below that $\mathcal{I}$ is a model of $\mathcal{O}$. These two properties are sufficient to prove completeness (a.k.a. Property (1)) for our fragment. Indeed, if $\mathcal{O} \models \alpha$, then $\mathcal{I} \models \alpha$ (since $\mathcal{I}$ is a model of $\mathcal{O}$). Together with (2), this implies that $\mathsf{atom}(\alpha) \in (\Pi^{ABox})^*$.

So to complete the proof, we only need to show that $\mathcal{I}$ is a model of $\mathcal{O}$, i.e. that it satisfies every axiom $\beta \in \mathcal{O}$. Due to space limitations, we focus on one type of axiom as an illustration, namely the case where β is of the form $\exists r \sqsubseteq_c a$ with $a, r \in \mathbb{I}$ (the proof for the other types of axioms is either simpler or analogous). By construction, $\texttt{subClassExt}(\exists r, a) \in \mathcal{D}_{\mathcal{O}}$, and $\mathsf{expr}(\exists r) = \{\texttt{domain}(\exists r, r)\} \subseteq \mathcal{D}_{\mathcal{O}}$. So from the second rule of Figure 1, $\texttt{propInst}(e_1, e_2, r) \in (\Pi^{ABox})^*_{\mathsf{facts}}$ implies $\texttt{classInst}(e_1, a) \in (\Pi^{ABox})^*_{\mathsf{facts}}$ for any e_1, e_2. Viewing $(\Pi^{ABox})^*_{\mathsf{facts}}$ as the interpretation $\mathcal{I}$, this implication translates to $(\exists r^{\mathcal{I}})^C \subseteq (a^{\mathcal{I}})^C$, i.e. $\mathcal{I} \models \beta$.

TBox Axioms. Next, we extend the program Π^{ABox} to a program Π that also derives TBox axioms, which gives us a complete system for *MDL-Lite*$_{\texttt{RDFS}}$. We use a technique that reduces the inference of such axioms to the inference of some ABox atoms, thus relying on the results of the previous section.

For each IRI $e \in \Sigma(\mathcal{T})$, we introduce three fresh constants (not in $\mathbb{I}$) that we call *witnesses* for e, denoted with w_e, w_e^s and w_e^o (for "subject" and "object"). Then we extend our program with the Datalog facts $\texttt{classWit}(w_e, e)$ and $\texttt{classInst}(w_e, e)$, which indicate that w_e is the class witness for e and an instance of e. We proceed similarly with the other two witnesses, adding to Π the atoms $\texttt{subjWit}(w_e^s, e)$, $\texttt{objWit}(w_e^o, e)$, $\texttt{propWit}(w_e^s, w_e^o, e)$ and $\texttt{propInst}(w_e^s, w_e^o, e)$. We also add to Π the set of inference rules listed in Figure 2, which rely on witnesses to infer TBox axioms.

We can now extend our proof of completeness to *MDL-Lite*$_{\texttt{RDFS}}$. We saw in the previous section that Π^{ABox} is complete for ABox axioms. Since $\Pi^{ABox} \subseteq \Pi$,

$$
\begin{aligned}
\texttt{subClass}(A, B) &\leftarrow \texttt{classWit}(E, A), \texttt{classInst}(E, B)\\
\texttt{subClass}(W, A) &\leftarrow \texttt{domain}(W, R), \texttt{subjWit}(E, R), \texttt{classInst}(E, A)\\
\texttt{subClass}(W, A) &\leftarrow \texttt{inverse}(R, S), \texttt{domain}(W, S), \texttt{objWit}(E, R), \texttt{classInst}(E, A)\\
\texttt{subProp}(R, S) &\leftarrow \texttt{subjWit}(E_1, R), \texttt{objWit}(E_2, R), \texttt{propInst}(E_1, E_2, S)\\
\texttt{subProp}(R, S') &\leftarrow \texttt{subjWit}(E_1, R), \texttt{objWit}(E_2, R), \texttt{inverse}(S', S), \texttt{propInst}(E_2, E_1, S)\\
\texttt{subProp}(R', S) &\leftarrow \texttt{subjWit}(E_1, R), \texttt{objWit}(E_2, R), \texttt{inverse}(R', R), \texttt{propInst}(E_2, E_1, S)\\
\texttt{subProp}(R', S') &\leftarrow \texttt{subjWit}(E_1, R), \texttt{objWit}(E_2, R), \texttt{inverse}(R', R), \texttt{inverse}(S', S),\\
&\qquad \texttt{propInst}(E_1, E_2, S)
\end{aligned}
$$

Fig. 2. Rules in charge of inferring TBox axioms

this property trivially extends to Π. So we can focus on proving that Π is complete for TBox axioms.

First, we note that for any TBox axiom α over $\Sigma(\mathcal{O})$, $\mathcal{D}_\mathcal{O} \subseteq \Pi$ contains the set $\{\mathsf{expr}(\eta) \mid \eta \text{ appears in } \alpha\}$. So, again, completeness can be reformulated as

$$\mathcal{O} \models \alpha \text{ implies } \mathsf{atom}(\alpha) \in \Pi^* \tag{3}$$

As above, we use $(\Pi)^*_{\mathsf{facts}}$ to refer to the set of Datalog facts that encode ABox axioms and belong to the least model Π^* of Π, and we leverage the fact that $(\Pi)^*_{\mathsf{facts}}$ can be viewed as an interpretation $\mathcal{I}$. We will also write $(\mathcal{D}_\mathcal{O})_{\mathsf{facts}}$ for the set of ABox atoms in $\mathcal{D}_\mathcal{O}$, i.e. all $\mathsf{atom}(\beta) \in \mathcal{D}_\mathcal{O}$ s.t. β is an ABox axiom.

As we saw in the previous section, in order to prove (3), it is sufficient to show that $\mathcal{I}$ is a model of $\mathcal{O}$, and that for any TBox axiom α,

$$(\Pi)^*_{\mathsf{facts}} \models \alpha \text{ implies } \mathsf{atom}(\alpha) \in \Pi^* \tag{4}$$

The proof that $\mathcal{I}$ is a model of $\mathcal{O}$ is identical to the one above, due to the fact that $\Pi^{ABox} \subseteq \Pi$. As for Property (4), for space limitations, we once again focus on one syntactic type of TBox axioms as an illustration, namely the case where α is of the form $\exists r \sqsubseteq_c a$ for $r, a \in \mathbb{I}$ (the proof for the other cases is either simpler or analogous).

Since $\mathcal{O} \models \alpha$ (by assumption) and $\mathcal{I}$ is a model of $\mathcal{O}$, we know that $\mathcal{I} \models \alpha$, or in other words, that every $(s, o) \in (r^\mathcal{I})^P$ satisfies $s \in (a^\mathcal{I})^C$. In Datalog terms (leveraging the correspondence between $\mathcal{I}$ and $(\Pi)^*_{\mathsf{facts}}$), this means that $\texttt{propInst}(s, o, r) \in (\Pi)^*_{\mathsf{facts}}$ implies $\texttt{classInst}(s, a) \in (\Pi)^*_{\mathsf{facts}}$. In particular, since $\texttt{propInst}(w^s_r, w^o_r, r) \in (\mathcal{D}_\mathcal{O})_{\mathsf{facts}} \subseteq (\Pi)^*_{\mathsf{facts}}$, this implies

$$\texttt{classInst}(w^s_r, a) \in (\Pi)^*_{\mathsf{facts}} \subseteq \Pi^* \tag{5}$$

Besides, by construction, the atoms $\texttt{subjWit}(w^s_r, a)$ and $\texttt{domain}(\exists r, r)$ are both in $\Pi \subseteq \Pi^*$. Together with (5) and the second rule of Figure 2, this implies

$$\texttt{subClass}(\exists r, a) = \mathsf{atom}(\alpha) \in \Pi^*,$$

which is what we wanted to prove.

This result, together with the structure of the program Π, allows us to derive an important (albeit unsurprising) property of our logic. For a TBox axiom α, the atom $\mathsf{atom}(\alpha)$ belongs to Π^* iff $\mathsf{atom}(\alpha)$ has been derived by means of one of the rules of Figure 2. Then, a simple syntactic analysis of our program shows that the ABox $\mathcal{A}$ cannot play any role in the derivation of such atoms (in particular, in Figure 2, atoms with predicate `classInst` or `propInst` can only be instantiated with witnesses). As a consequence, all TBox axioms implied by $\mathcal{O}$ follow from the TBox $\mathcal{T}$ alone, i.e.:

Lemma 1. *For a MDL-Lite$_{\texttt{RDFS}}$ ontology $\mathcal{O} = \langle \mathcal{T}, \mathcal{A} \rangle$ and TBox axiom α,*

$$\mathcal{O} \models \alpha \text{ implies } \mathcal{T} \models \alpha$$

Filtering out Auxiliary Constants. The Datalog program Π that we introduced is complete (in the sense defined above), but not sound, because Π^* may contain atoms of the form $\texttt{classInst}(w_{e_1}, e_2)$ and $\texttt{propInst}(w^s_{e_1}, w^o_{e_1}, e_2)$, which are technically not implied by $\mathcal{O}$, since witnesses are fresh. We made this choice to simplify our presentation and proof of completeness. However, soundness can easily be recovered by eliminating such atoms from Π^*.

To do this, we first modify the definition of $\mathsf{atom}(\alpha)$, in such a way that it now uses a fresh predicate $\texttt{classInst}'$ (resp. $\texttt{propInst}'$, $\texttt{subClass}'$, $\texttt{subProp}'$) in place of `classInst` (resp. `propInst`, `subClass`, `subProp`). For instance the definition of $\mathsf{atom}(a(e))$ becomes $\texttt{classInst}'(e, a)$, instead of $\texttt{classInst}(e, a)$.

Then we introduce a predicate `IRI`, in charge of collecting the signature of $\mathcal{O}$, and we use $\mathcal{D}_\Sigma$ for the set $\{\texttt{IRI}(e) \mid e \in \Sigma(\mathcal{O})\}$. To complete the construction, we add to Π this set of atoms $\mathcal{D}_\Sigma$, as well as four additional rules, one for each of the fresh predicates, which retains only constants from $\Sigma(\mathcal{O})$, thanks to the predicate `IRI`. For instance, for $\texttt{classInst}'$, the rule is

$$\texttt{classInst}'(E, A) \leftarrow \texttt{classInst}(E, A), \texttt{IRI}(E), \texttt{IRI}(A)$$

5 QUERY ANSWERING

In this section, we show how answering an UCMQ $q(\bar{X})$ over an *MDL-Lite*$_{\texttt{RDFS}}$ ontology $\mathcal{O}$ under MSER amounts to evaluating a Datalog query over the program defined in Section 4. Then we study the complexity of this problem.

Query answering. We first generalize the definition of $\mathsf{atom}(\alpha)$ to atoms that may contain variables (the definition is identical otherwise). For instance, if $\alpha = a \sqsubseteq X$, then $\mathsf{atom}(\alpha) = \texttt{subClass}(a, X)$. Next, to each disjunct $q'(\bar{X}) = \exists \bar{Y}.\alpha_1 \wedge \cdots \wedge \alpha_n$ of q, we associate the Datalog rule with head $\mathtt{q}'(\bar{X})$ and body $\bigcup_{i \in [1..n]} \mathsf{enc}(\alpha_i)$. We extend Π with each of these rules, as well as one rule for q, with head $\mathtt{q}(\bar{X})$ and body $\{\mathtt{q}'(\bar{X}) \mid q' \text{ is a disjunct of } q\}$. Let $\Pi_{\mathcal{O},q}$ be the resulting program. Because Π is complete, a valuation $\bar{d}$ for $\bar{X}$ is an answer to q over $\mathcal{O}$ iff the atom $\mathtt{q}(\bar{d})$ is in $(\Pi_{\mathcal{O},q})^*$.

Decision problem. As is conventional in the database literature, in order to study the complexity of query answering, we focus on Boolean queries, i.e. we investigate the following decision problem:

QueryAnswering $(q, \mathcal{O})$
Input: Boolean UCMQ q, *MDL-Lite*$_{\texttt{RDFS}}$ ontology $\mathcal{O}$
Decide: $\mathcal{O} \models q$

Ontology Complexity, Membership. Let n denote the size of (the encoding of) $\mathcal{O}$. Because our program Π is sound and complete (in the sense of Section 4), one can use the following procedure to decide $\mathcal{O} \models q$. First, compute Π out of $\mathcal{O}$, in $O(n)$. Then for each disjunct $q'(\bar{X}; \bar{Y})$ of q, and for each valuation $(\bar{d}, \bar{e})$ for $(\bar{X}, \bar{Y})$, decide whether $\mathsf{enc}(\phi) \subseteq \Pi^*$ holds for every ground atom $\phi \in q(\bar{X}/\bar{d}; \bar{Y}/\bar{e})$.

The amount of extra space needed to iterate over all such valuations does not depend on $\mathcal{O}$ (but only on $\bar{X}$ and $\bar{Y}$). Then we observe that the size of the set $\mathsf{enc}(\phi)$ is fixed (it contains up to three atoms). Deciding whether a fact is in the least model of a linear Datalog program is known to be in NLogSpace (see [15], Theorem 6.8). Since Π is linear (and has size in $O(n)$), this allows us to conclude that QueryAnswering is in NLogSpace in the size of $\mathcal{O}$.

ABox (a.k.a. Data) Complexity, Membership. We now use n to denote the size of (the encoding of) $\mathcal{A}$, and rely on the same procedure as in the previous section to provide an upper bound on the cost of QueryAnswering in terms of n.

The amount of time needed to iterate over all valuations for $(\bar{X}, \bar{Y})$ is in $O(n)$, and the amount of time needed to iterate over each disjunct q' of q, each atom $\phi \in q(\bar{X}/\bar{d}; \bar{Y}/\bar{e})$ and each atom $\alpha \in \mathsf{enc}(\phi)$ is in $O(1)$. If α is an extensional atom, then $\alpha \in \Pi^*$ can be decided in $O(n)$, by iterating over Π. If α is intensional with predicate $\texttt{subClass}'$ or $\texttt{subProp}'$, then from Lemma 1, $\alpha \in \Pi^*$ can be decided in $O(1)$. The last possible case is if α has predicate $\texttt{classInst}'$ or $\texttt{propInst}'$. We show that in this case, $\alpha \in \Pi^*$ can be decided by answering a Boolean First-Order (FO) query over the encoding $\mathcal{D}_\mathcal{O}$ of $\mathcal{O}$, and that this query can be computed in $O(1)$ (and therefore also has size in $O(1)$). This implies that QueryAnswering is in AC^0 in ABox complexity.

Let β be the atom identical to α, but where $\texttt{classInst}'$ (resp. $\texttt{propInst}'$) is replaced with $\texttt{classInst}$ (resp. $\texttt{propInst}$). Then by construction, $\alpha \in \Pi^*$ iff $\beta \in \Pi^*$. And from (1), we know that $\beta \in \Pi^*$ iff $\beta \in (\Pi^{ABox})^*$. So we can focus on the subprogram Π^{ABox}, whose rules are in Figure 1. These rules can be partially grounded (with constants) for all variables but E, E_1, and E_2, by inspecting only atoms of $\mathcal{D}_\mathcal{O}$ with predicate $\texttt{inverse}$, $\texttt{domain}$, $\texttt{subClassExt}$ or $\texttt{subPropExt}$. The number of such atoms is independent of $\mathcal{A}$ (it only depends on $\mathcal{T}$), so this partial grounding can be performed in $O(1)$. For instance, a partial grounding of the first rule may be $\texttt{classInst}(E, b) \leftarrow \texttt{classInst}(E, a), \texttt{subClassExt}(a, b)$, where a and b are constants.

Next, each rule in the resulting set Ψ of partially grounded rules can be finitely "unfolded", meaning that recursively replacing each intensional atom

ABox complexity	in AC^0
TBox complexity	NLogSpace-complete
ontology complexity	NLogSpace-complete
query complexity	NP-complete
combined complexity	NP-complete

Fig. 3. Complexity of QueryAnswering

by its definitions in Ψ (and renaming variables accordingly) yields a finite set of non-recursive rules. This can be easily seen by observing that *(i)* the only rules in Ψ whose body introduces variables (i.e. not present in the head) are instantiations of the second and third rules of Figure 1, *(ii)* the only intensional predicate in these rules is `propInst` and *(iii)* the rules of Ψ with `propInst` as predicate do not introduce variables (their bodies only use E_1 and E_2).

For $\psi \in \Psi$, let $\mathsf{unfold}(\psi)$ denote the unfolding of ψ. If β is of the form $\texttt{classInst}(e, a)$, then $\beta \in (\Pi^{ABox})^*$ iff $\beta \in (\mathcal{D}_\mathcal{O} \cup \Psi_\beta)^*$, where $\Psi_\beta = \bigcup\{\mathsf{unfold}(\psi) \mid \psi \in \Psi$ and ψ has head predicate $\texttt{classInst}(E, a)\}$, and similarly for $\beta = \texttt{propInst}(e_1, e_2, r)$ (with head predicate $\texttt{propInst}(E_1, E_2, r)$). Because Ψ_β is non-recursive, deciding $\beta \in (\mathcal{D}_\mathcal{O} \cup \Psi_\beta)^*$ is equivalent to answering a Boolean FO query (precisely, a union of conjunctive queries) over $\mathcal{D}_\mathcal{O}$, and this query only depends Ψ and β, which are independent of $\mathcal{A}$.

TBox Complexity, Hardness. We provide a reduction from the *graph reachability problem*, which is known to be NLogSpace-complete, to QueryAnswering. Given a directed graph G with nodes V and edges E, and a pair (s, t) of nodes in V, graph reachability consists in deciding whether there exists a path from s to t in G. Viewing E as a binary relation over V, this is equivalent to deciding whether (s, t) is in the transitive closure E^* of E. W.l.o.g., we can assume that E is a superset of the identity relation over V.

We represent each node $v \in V$ as a distinct atomic concept c_v, and leverage the fact that the relation $\sqsubseteq_c$ is (semantically) transitive. Precisely, given $G = \langle V, E\rangle$ and $s, t \in V$ as input, we define the UCMQ $q() = c_s \sqsubseteq_c c_t$ and the ontology $\mathcal{O}^G = \langle \mathcal{T}^G, \emptyset\rangle$, where $\mathcal{T}^G = \{c_{v_1} \sqsubseteq_c c_{v_2} \mid (v_1, v_2) \in E\}$.

Now let $R = \{(c_{v_1}, c_{v_2}) \mid c_{v_1} \sqsubseteq_c c_{v_2} \in \mathcal{T}^G\}$, and let R^* denote the transitive closure of R. Immediately from the semantics of *MDL-Lite*$_{\texttt{RDFS}}$, $\mathcal{O} \models c_{v_1} \sqsubseteq_c c_{v_2}$ iff $(c_{v_1}, c_{v_2}) \in R^*$, for any $v_1, v_2 \in V$. And by construction, $(c_{v_1}, c_{v_2}) \in R^*$ iff $(v_1, v_2) \in E^*$. Therefore $\mathcal{O} \models q$ iff $(s, t) \in E^*$.

This reduction requires a constant amount of extra space, and the sizes of q and $\mathcal{A}$ are fixed (i.e. independent of the size of G). This allows us to conclude that QueryAnswering is NLogSpace-hard in the size of $\mathcal{T}$.

Query and Combined Complexity. Finally, we show that QueryAnswering is NP-complete in both query and combined complexity. NP-hardness for query (and hence combined) complexity is inherited from the well-known complexity of Boolean CQ entailment over an ABox. Membership in combined (and hence

query) complexity in NP is also straightforward: we can use a non-deterministic version of the algorithm used above for NLogSpace membership (in the size of $\mathcal{O}$): guess a disjunct $q'(\bar{X};\bar{Y})$ of q, and a valuation $(\bar{d},\bar{e})$ for $(\bar{X},\bar{Y})$ such that $\mathsf{enc}(\phi) \subseteq \Pi^*$ holds for every ground atom $\phi \in q(\bar{X}/\bar{d};\bar{Y}/\bar{e})$. Then check that this holds for each such ϕ. The number of such checks is linear in the size of q'. and the cost of each of them is independent on the size of q, and, as we have seen above, (less than) polynomial in the size of $\mathcal{T}$ and $\mathcal{A}$.

6 CONCLUSION

In this paper, we addressed metaquerying on *MDL-Lite*$_{\texttt{RDFS}}$ (the metamodeling counterpart of *DL-Lite*$_{\texttt{RDFS}}$) ontologies under MSER semantics. Specifically, we focused on unions of conjunctive metaqueries, and we provided a reduction from the metaquery answering problem to the evaluation of a Datalog query over a (positive) Datalog program that encodes in a sound and complete way the knowledge represented by the ontology. Besides, the rules of this program are independent of the input ontology. The fact that this program is linear allowed us to established a tight NLogSpace bound for TBox and ontology complexity, while showing that data complexity remains in AC^0, and that query and combined complexity are NP-complete. Our approach also allows immediate implementations using off-the-shelf Datalog engines.

We plan to extend our work in several directions, possibly simultaneously. First, an interesting extension would be to consider the first-order logic semantics (rather than the SPARQL semantics) for query answering. Second, both the ontology and the query language could be enriched by allowing, e.g., concept and role disjointnesses, inequalities, negated atoms, and concept (resp. role) expressions in the right-hand side of concept (resp. role) inclusions.

Acknowledgments. This work has been supported by the PNRR MUR project PE0000013-FAIR.

References

1. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley (1995)
2. Ahmetaj, S., Fischl, W., Pichler, R., Simkus, M., Skritek, S.: Towards reconciling SPARQL and certain answers. In: WWW 2015. pp. 23–33 (2015)
3. Arenas, M., Gottlob, G., Pieris, A.: Expressive languages for querying the semantic web. ACM Transactions on Database Systems **43**(3), 13:1–13:45 (2018)
4. Atkinson, C., Kühne, T.: The essence of multilevel metamodeling. In: UML 2001. Lecture Notes in Computer Science, vol. 2185, pp. 19–33 (2001)
5. Atkinson, C., Kühne, T.: Model-driven development: A metamodeling foundation. IEEE Software **20**(5), 36–41 (2003)
6. Bienvenu, M.: Ontology-mediated query answering: Harnessing knowledge to get more from data. In: IJCAI 2016. pp. 4058–4061 (2016)

7. de Carvalho, V.A., Almeida, J.P.A., Fonseca, C.M., Guizzardi, G.: Multi-level ontology-based conceptual modeling. Data Knowl. Eng. **109**, 3–24 (2017)
8. Chen, W., Kifer, M., Warren, D.S.: HILOG: A foundation for higher-order logic programming. J. of Logic Programming **15**(3), 187–230 (1993)
9. Cima, G., Console, M., Delfino, R., Lenzerini, M., Poggi, A.: Answering conjunctive queries with safe negation and inequalities over RDFS knowledge bases. In: AAAI 2025. pp. 14824–14831 (2025)
10. Cima, G., Console, M., Lenzerini, M., Poggi, A.: Monotone abstractions in ontology-based data management. In: AAAI 2022. pp. 5556–5563 (2022)
11. Cima, G., De Giacomo, G., Lenzerini, M., Poggi, A.: On the SPARQL metamodeling semantics entailment regime for OWL 2 QL ontologies. In: WIMS 2017. pp. 10:1–10:6 (2017)
12. Cima, G., Lenzerini, M., Poggi, A.: Answering conjunctive queries with inequalities in *DL-Lite*$_{\mathscr{R}}$. In: AAAI 2020. pp. 2782–2789 (2020)
13. Cima, G., Poggi, A., Lenzerini, M.: The notion of abstraction in ontology-based data management. Artif. Intell. **323**, 103976 (2023)
14. Cuenca Grau, B.: A possible simplification of the semantic web architecture. In: WWW 2004. pp. 704–713 (2004)
15. Dantsin, E., Eiter, T., Gottlob, G., Voronkov, A.: Complexity and expressive power of logic programming. ACM Comput. Surv. **33**(3), 374–425 (Sep 2001)
16. De Giacomo, G., Lenzerini, M., Rosati, R.: Higher-order description logics for domain metamodeling. In: AAAI 2011. pp. 183–188 (2011)
17. Glimm, B.: Using SPARQL with RDFS and OWL entailment. In: RW 2011. pp. 137–201. Lecture Notes in Computer Science, Springer (2011)
18. Glimm, B., Kazakov, Y., Kollia, I., Stamou, G.B.: Lower and upper bounds for SPARQL queries over OWL ontologies. In: AAAI 2015. pp. 109–115 (2015)
19. Gottlob, G., Pieris, A.: Beyond SPARQL under OWL 2 QL entailment regime: Rules to the rescue. In: IJCAI 2015. pp. 2999–3007 (2015)
20. Harris, S., Seaborne, A.: SPARQL 1.1 query language. W3C Recommendation (2013), available at https://www.w3.org/TR/sparql11-query/
21. Kontchakov, R., Rezk, M., Rodriguez-Muro, M., Xiao, G., Zakharyaschev, M.: Answering SPARQL queries over databases under OWL 2 QL entailment regime. In: ISWC 2014. Lecture Notes in Computer Science, vol. 8796, pp. 552–567 (2014)
22. Kubincová, P., Kluka, J., Homola, M.: Expressive description logic with instantiation metamodelling. In: KR 2016. pp. 569–572 (2016)
23. Lenzerini, M., Lepore, L., Poggi, A.: A higher-order semantics for metaquerying in OWL 2 QL. In: KR 2016. pp. 577–580 (2016)
24. Lenzerini, M., Lepore, L., Poggi, A.: Metamodeling and metaquerying in OWL 2 QL. Artif. Intell. **292**, 103432 (2021)
25. Levesque, H.J.: Foundations of a functional approach to knowledge representation. Artif. Intell. **23**, 155–212 (1984)
26. Martinez, M., Rohrer, E., Severi, P.: Complexity of the description logic ALCM. In: KR 2016. pp. 585–588 (2016)
27. Motik, B.: On the properties of metamodeling in OWL. J. of Logic and Computation **17**(4), 617–637 (2007)
28. Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: OWL 2 Web Ontology Language profiles (second edition). W3C Recommendation (2012), available at http://www.w3.org/TR/owl2-profiles/
29. Motik, B., Patel-Schneider, P.F., Cuenca Grau, B.: OWL 2 Web Ontology Language direct semantics (second edition). W3C Recommendation (2012), available at https://www.w3.org/TR/owl2-direct-semantics/

30. Motz, R., Rohrer, E., Severi, P.: The description logic SHIQ with a flexible meta-modelling hierarchy. J. Web Sem. **35**, 214–234 (2015)
31. Poggi, A.: On the SPARQL direct semantics entailment regime for OWL 2 QL. In: DL 2016. CEUR Workshop Proc. vol. 1577 (2016)
32. Qureshi, H.M., Faber, W.: Efficient OWL2QL meta-reasoning using asp-based hybrid knowledge bases. In: ICLP 2024. EPTCS, vol. 416, pp. 188–200 (2024)
33. Qureshi, H.M., Faber, W.: Evaluating datalog tools for meta-reasoning over OWL 2 QL. Theory and Practice of Logic Programming **24**(2), 368–393 (2024)
34. Rosati, R.: The limits of querying ontologies. In: ICDT 2007. Lecture Notes in Computer Science, vol. 4353, pp. 164–178. Springer (2007)
35. Xiao, G., Calvanese, D., Kontchakov, R., Lembo, D., Poggi, A., Rosati, R., Zakharyaschev, M.: Ontology-based data access: A survey. In: IJCAI 2018. pp. 5511–5519 (2018)