On the SPARQL Metamodeling Semantics Entailment Regime for OWL 2 QL ontologies

Gianluca Cima
Sapienza University of Rome
cima@diag.uniroma1.it

Giuseppe De Giacomo
Sapienza University of Rome
degiacomo@diag.uniroma1.it

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

Antonella Poggi
Sapienza University of Rome
antonella.poggi@uniroma1.it

ABSTRACT

OWL 2 QL is the profile of OWL 2 targeted to Ontology-Based Data Access (OBDA) scenarios, where large amount of data are to be accessed, and thus answering conjunctive queries over data is the main task. However, this task is quite restrained wrt the classical KR *Ask-and-Tell* framework based on querying the whole theory, not only facts (data). If we use SPARQL as query language, we get much closer to this ideal. Indeed, SPARQL queries over OWL 2 QL, under the so-called *Direct Semantics Entailment Regime*, may comprise any assertion expressible in the language, i.e., both ABox atoms and TBox atoms, including inequalities expressed by means of DifferentIndividuals. Nevertheless this regime is hampered by the assumption that variables in queries need to be *typed*, meaning that the same variable cannot occur in positions of different types, e.g., both in class and individual position (*punning*). In this paper we dismiss this limiting assumption by resorting to a recent meta modeling semantics and show that query answering in the resulting entailment regime is polynomially compilable into Datalog (and hence PTIME wrt both TBox and ABox).

CCS CONCEPTS

•Information systems → Mediators and data integration; Web Ontology Language (OWL); •Computing methodologies → Description logics; •Theory of computation → *Higher order logic;*

KEYWORDS

Ontology-Based Data Access, OWL 2, Datalog

ACM Reference format:
Gianluca Cima, Giuseppe De Giacomo, Maurizio Lenzerini, and Antonella Poggi. 2017. On the SPARQL Metamodeling Semantics Entailment Regime for OWL 2 QL ontologies. In *Proceedings of WIMS '17, Amantea, Italy, June 19-22, 2017,* 6 pages.
DOI: 10.1145/3102254.3102277

1 INTRODUCTION

DL-Lite is a family of Description Logics specifically designed for achieving tractability in answering conjunctive queries (CQs) expressed over ontologies. In particular, when the ontology is expressed in one of the logics of the family, i.e., *DL-Lite*$_\mathcal{R}$, query answering can be reduced to standard evaluation of first-order queries over a database, and can therefore make use of SQL engines. By virtue of these characteristics, *DL-Lite*$_\mathcal{R}$ is at basis of OWL 2 QL, the OWL 2 profile especially designed for Ontology-Based Data Access (OBDA) applications, where the aim is to use an ontology to access a typically big amount of data residing in external data sources.

SPARQL is the de-facto standard language for expressing queries over OWL 2 ontologies. The key form of SPARQL query is the so-called basic graph pattern, that is a conjunction of atoms, where each atom has the form of an axiom expressible in the ontology. So, any axiom that can appear in the ontology can also appear in the conjunctive query pattern corresponding to the SPARQL query. This is coherent with the classical knowledge representation *Ask-and-Tell* framework [12], which is based on "*ask* anything that can be *told* to a knowledge base". Notice that when the Ask-and-Tell framework is specialized to OBDA, it reflects the idea of *querying whole* OWL 2 QL *theories*, specifying patterns spanning both through the TBox (the intensional knowledge represented in the ontology), and the ABox (the *facts* at the extensional level of the ontology), with no limitation on the use of variables, and therefore with a distinct metamodeling and metaquerying flavor, see, e.g., [11].

Starting from the seminal work [3], there has been a huge amount of work aiming at designing optimized algorithms for query answering in OBDA, and developing systems implementing such algorithms. So, it is natural to ask whether, after such body of work, the problem of querying OWL 2 QL theories is solved. Surprisingly, the answer to this question is negative, for the following reasons.

First, the queries studied in the great majority of the works on OBDA contain only ABox atoms. In other words, the CQs expressible in the current OBDA systems are able essentially to specify patterns in the data, and retrieve individual objects satisfying such patterns. Obviously, the intensional knowledge represented in the ontology (TBox axioms) is taken into account when answering the query, but in most of the results, the assumption is that TBox atoms do not appear in queries.

Second, the syntax and the semantics of SPARQL conjunctive queries over OWL 2 QL ontologies are defined by means of the so-called *Direct Semantics Entailment Regime (DSER)* [4], which interprets the ontology under the *Direct Semantics (DS)*, i.e., as a first-order theory, and defines the solutions to a query as the set of tuples of IRIs occurring in the ontology that, once substituted to the variables within the query, make the resulting set of axioms *logically implied* by the ontology. The fact that DS interprets the ontology as a first order theory has an important implication: although the syntactic rules for defining an ontology allow *punning*, that is the capability of using the same name in positions of different *type* (i.e., in an object and in a predicate position, or in a class and in a property position), the semantics imposes that occurrences of the same name in different positions are treated as if they were occurrences of different names. A direct consequence of this choice is that the use of variables in the queries is limited by the so-called *typing constraint*, by which no variable can appear both in object position (i.e., as an argument of a class or of a property), and in predicate position (i.e., as class or property). The result is that we cannot join variables denoting classes with those denoting individuals, thus preventing the specification of interesting queries related to metamodeling, e.g., the one asking for all classes that are instances of another class [11].

Third, although OWL 2 QL allows specifying inequalities between IRIs, by means of axioms of the form DifferentIndividuals(e_1 e_2), imposing that e_1 and e_2 denote distinct objects of the domain, the inequality predicate is completely dismissed in the work on query answering over OWL 2 QL ontologies. Notably, it is a folk theorem that extending the approach to cover the inequality predicate is completely trivial, but we argue in this paper that this is not the case in general, and in particular when querying whole theories. Indeed, inequalities between ontology entities can be logically implied by an OWL 2 QL ontology, as shown in the following simple example. Consider the ontology consisting of the axioms:

```
ClassAssertion(:Male :p). ClassAssertion(:Male :peter).
ClassAssertion(:Female :petra). SubClassOf(:Female :Person).
SubClassOf(:Male :Person). DisjointClasses(:Female :Male)
```

and suppose we want to retrieve all classes that contain at least two distinct instances, by means of the SPARQL query:

```
select $x where {ClassAssertion($x $y).ClassAssertion($x $z).
                                DifferentIndividuals($y $z)}
```

It is easy to see that, since :Male and :Female are disjoint, :petra and :peter denote distinct domain objects in every model. Hence, the answer to the query is {:Person,owl:Thing}. This clearly shows that the presence of inequalities within queries requires forms of reasoning taking into account the whole ontology.

The goal of this paper is to present the first approach to querying OWL 2 QL theories, which unleashes the potentiality of the metamodeling characteristics inherent in OWL 2 QL, and of the metaquerying capabilities of SPARQL. To this aim we introduce a new entailment regime, called *Metamodeling Semantics Entailment Regime (MSER)*, which generalizes DSER by (*i*) adopting the Metamodeling Semantics (MS) for OWL 2 QL, introduced in [11] [1], and (*ii*) relaxing the typing constraint of DSER, thus allowing the same variable to occur in positions of different type, e.g., in object and class position.

[1]In fact, in [11], the authors use 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.

The main contribution of this paper is to show that both checking the consistency of an OWL 2 QL ontology and answering SPARQL queries over an OWL 2 QL ontology under MSER are polynomially compilable into Datalog (and hence PTIME wrt both TBox and ABox). It is easy to verify that querying OWL 2 QL ontologies with SPARQL under DSER (but *without inequality axioms*) can be polynomially reduced to evaluating a Datalog program. Our result shows that the same is true for the more general problem of querying OWL 2 QL theories under MSER, without any restriction.

The paper is organized as follows. In Section 2, we introduce preliminary notions, among which the SPARQL MSER for OWL 2 QL. Then, in Section 3, we show how to reduce consistency checking and query answering under MSER to the evaluation of a Datalog program, and in Section 4, we present experiments which serve as a proof of concept for our query answering technique. In Section 5, we present some related work. Finally, in Section 6, we provide some conclusions and discuss future directions of our work.

2 PRELIMINARIES

OWL 2 QL. Ontology *entities*, such as individuals, classes, object properties, etc., are denoted by *expressions*. *Atomic expressions* correspond to element in the ontology vocabulary. The *vocabulary* V_O of an ontology O is defined as the tuple (V_N, V_C, V_{OP}, V_{DP}, V_{DT}, L_{QL}), where (*i*) V_N is the set of IRIs occurring in O extended with the OWL 2 QL reserved vocabulary, (*ii*) V_C (resp., V_{OP}, V_{DP}), is the subset of V_N consisting of the IRIs that appear in class (resp., object property, data property) positions in O, or are reserved IRIs (*Internationalized Resource Identifiers*) denoting classes (resp., object properties, data properties) (*iii*) V_{DT} is a subset of the datatypes in OWL 2 QL, and (*iv*) L_{QL} is the set of literals occurring in some logical axiom of O. *Complex expressions* are built on the basis of V_O. We denote by Exp^O the *finite* set of *expressions* that can be built on V_O. Thus, for example, if $e \in V_{OP}$, then ObjectInverseOf(e) and ObjectSomeValuesFrom(e_1 e_2) are complex expressions in Exp^O. For the sake of simplicity, in the following, we use the Description Logic (DL) syntax for denoting OWL 2 QL expressions. For example, we respectively denote by e^- and $\exists e_1.e_2$ the two above mentioned expressions. Also, we denote by $\top_C, \top_R, \top_D, \bot_C, \bot_R, \bot_D$, respectively, the OWL 2 QL reserved IRIs: owl:Thing, owl:topObjectProperty,owl:topDataProperty,owl:Nothing,owl:bottomObjectProperty,owl:bottomDataProperty.

An OWL 2 QL theory, or OWL 2 QL ontology (or simply *ontology* in the following) is a finite set of OWL 2 QL *(logical) axioms*. As we do for expressions, we write axioms using the DL syntax, which we modify as described next. Inclusions axioms are written as $e_1 \sqsubseteq_* e_2$, where the subscript * is C, R, A, or D, depending on whether the inclusion is an inclusion between classes, object properties, data properties, or datatypes, respectively. Similarly, disjointness axioms are written as $e_1 \sqsubseteq_* \neg e_2$. Also, axioms asserting that an object property e is reflexive (irreflexive), are written ref(e), (resp., irr(e)). We classify logical axioms into (*i*) *positive TBox axioms*, i.e., inclusion and reflexivity axioms, (*ii*) *negative TBox axioms*, i.e., disjointness and irreflexivity axioms, and (*iii*) *ABox axioms*, i.e., axioms of the forms $C(a)$, $R(a, b)$, $A(a, b)$, $a \neq b$. Axioms not in the above list can be expressed by appropriate combinations of the ones listed.

OWL 2 Metamodeling Semantics The OWL 2 Metamodeling Semantics (MS) was introduced in [11] and is based on the notion of interpretation structure, which plays the same role as the "interpretation domain" in classical first-order logic. Specifically, an *interpretation structure* is a tuple $\Sigma = \langle \Delta_o, \Delta_v, \cdot^I, \cdot^E, \cdot^R, \cdot^A, \cdot^T \rangle$ where:

- Δ_o, the *object domain*, and Δ_v, the *value domain* are two disjoint nonempty sets, forming the interpretation domain $\Delta = \Delta_o \cup \Delta_v$;
- $\cdot^E : \Delta_o \rightarrow \mathcal{P}(\Delta_o)$ is a partial function;
- $\cdot^R : \Delta_o \rightarrow \mathcal{P}(\Delta_o \times \Delta_o)$ is a partial function;
- $\cdot^A : \Delta_o \rightarrow \mathcal{P}(\Delta_o \times \Delta_v)$ is a partial function;
- $\cdot^T : \Delta_o \rightarrow \mathcal{P}(\Delta_v)$ is a partial function;
- $\cdot^I : \Delta_o \rightarrow \{\mathsf{T}, \mathsf{F}\}$ is a total function s.t. for each $d \in \Delta_o$, if $\cdot^E, \cdot^R, \cdot^A, \cdot^T$ are undefined for d, then $d^I = \mathsf{T}$.

Thus, the interpretation structure is not simply a set, but a mathematical representation of a world made up by elements which have complex inter-relationships, where such inter-relationships are represented by the various functions constituting Σ. Also, an *interpretation* $\mathcal{I}$ for O is a pair, $\langle \Sigma, \mathcal{I}_o \rangle$, where Σ is an interpretation structure and $\mathcal{I}_o$ is the *interpretation function* for $\mathcal{I}$, i.e., a function that maps every expression in Exp^O into an object in Δ_o, and every literal in L_{QL} into a value in Δ_v, according to an appropriate set of conditions. For example, one condition imposes that $\perp_C$ is interpreted as an object associated through $\cdot^E$ to an empty set. As another example, if an entity e is interpreted as an object regarded as a relation R_e, then the expression e^- is interpreted as the inverse of R_e. For the full list of conditions, please refer to [11]. Finally, to define the semantics of logical axioms, MS resorts to the usual notion of satisfaction of an axiom with respect to an interpretation $\mathcal{I}$. Thus, for example, $\mathcal{I} \models (e_1 \sqsubseteq_C e_2)$ if $(e_1^{\mathcal{I}_o})^E$ and $(e_2^{\mathcal{I}_o})^E$ are defined, and $(e_1^{\mathcal{I}_o})^E \subseteq (e_2^{\mathcal{I}_o})^E$ where e_1, e_2 are expressions. As another notable example, $\mathcal{I} \models e_1 \neq e_2$ if $e_1^{\mathcal{I}_o} \neq e_2^{\mathcal{I}_o}$.

SPARQL. We concentrate on conjunctive queries (simply called *queries* in the following), expressed using SPARQL. Let O be an ontology and $\mathcal{V}$ a set of variables. We start by introducing the notion of query atom. A *query atom over* O (simply called *atom* in the following) has the same form of an axiom with the difference that its arguments belong to the set of *terms over* O *and* $\mathcal{V}$. The set of terms, denoted $Exp^O_{\mathcal{V}}$, is defined similarly to Exp^O, with the difference that its base set is $V^O_N \cup \mathcal{V}$, rather than simply V^O_N. Note that, similarly to axioms, atoms can be classified into TBox atoms and ABox atoms.

A conjunctive query q over an ontology is an expression of the form

select $x_1 \ldots x_n$ **where** $\{ B \}$,

where $n \geq 1$, $x_1, \ldots, x_n$ are variables, called *distinguished variables*, n is the *arity* of the query, and B, called *body of the conjunctive query* q and denoted by $body(q)$, is a non-empty conjunction of atoms. In the following we will use the notation $q(\vec{x}, \vec{y}) : \vec{x} \leftarrow B$, or simply $q(\vec{x}, \vec{y})$, to denote a query of the form above, where $\vec{x} = (x_1, \ldots, x_n)$ and $\vec{y}$ is the tuple of variables that occur in B and do not belong to $\vec{x}$. Note that in SPARQL jargon, a conjunction of atoms is a *basic graph pattern*, i.e., a conjunction of RDF triples involving variables. However, for the sake of clarity and without loss of generality, instead of using the RDF syntax, we use here the DL syntax.

Queries are interpreted by relying on the notion of SPARQL *entailment regime*, as defined in the SPARQL 1.1. W3C standard specification[2]. Specifically, a SPARQL entailment regime defines:

(*i*) the syntax and the semantics of axioms constituting the queried ontology,
(*ii*) the syntax of conjunctive queries considered legal for the regime, and
(*iii*) the semantics of such queries, i.e., what are the answers to a query.

The most typical SPARQL Entailment Regime for OWL 2 QL ontologies is the Direct Semantics Entalment Regime (DSER). However, based on its limitations discussed in the introduction, we introduce here a new entailment regime, called *Metamodeling Semantics Entailment Regime (MSER)*, which generalizes DSER as described in the following. As for (*i*), MSER assumes to deal with OWL 2 QL ontologies interpreted according to MS, which generalizes the Direct Semantics adopted by DSER. As for (*ii*), it considers as legal the whole set of queries of the form specified above, while DSER restricts to queries where variables can occur only in positions of the same type. Thus, for example, in DSER, a variable occurring in object position cannot also occur in class position (e.g. in $\sqsubseteq_C$ axioms). As for (*iii*), MSER defines the answers to a query similarly to DSER, except that it uses MS for logical entailment. Specifically, given a tuple of variables $\vec{z} = (z_1, \ldots, z_n)$, a tuple of IRIs $\vec{w} = (w_1, \ldots, w_n)$, and a conjunction of atoms B, we denote by $\sigma[\vec{z} \rightarrow \vec{w}](B)$ the conjunction of atoms obtained from B by substituting each z_i in $\vec{z}$ with w_i in $\vec{w}$, for $i \in \{1, \ldots, n\}$. Now, let O be an ontology and $q(\vec{x}, \vec{y})$ a conjunctive query. An n-tuple of IRIs $\vec{t}$ is an *answer to* $q(\vec{x}, \vec{y})$ *over* O *under MSER* if there exists an m-tuple of IRIs $\vec{v}$ such that

$$O \models \sigma[(\vec{x}, \vec{y}) \rightarrow (\vec{t}, \vec{v})](body(q)).$$

Also, $Ans(q, O)$ denotes the set of answers to q over O under MSER. It is worth noting that in MSER, similarly to DSER and in contrast to classical logic, existential variables $\vec{y}$, although projected out from the answer, are required to be *bound* to the same m-tuple of constants $\vec{v}$, in every model of O.

3 REDUCING CONSISTENCY CHECKING AND QUERY ANSWERING TO DATALOG EVALUATION

In this section, we show that consistency checking and query answering over OWL 2 QL ontologies under MSER can be reduced to the evaluation of a Datalog program. Note that, for the sake of simplicity, from now on we implicitly assume to deal with ontologies that do not contain data properties. But our results can be immediately extended to ontologies containing data properties as well.

Intuitively, the key ideas of our approach are the following. First, we define a finite number of inference rules which capture reasoning in OWL 2 QL, i.e. which are sound and complete with respect to logical implication under MSER. Second, we define a translation function τ from the set of legal OWL 2 QL axioms to the set of instances of a database schema S^{ql}, where S^{ql} comprises a distinct relation for each distinct *form* of axiom, whose arity is the number of atomic expressions that occur in axioms of that form. Then, for example, inclusion axioms of the form $c_1 \sqsubseteq_C \exists r_2^-.c_2$, where

[2]www.w3.org/TR/sparql11-entailment/

	Inference rule	Datalog rule
$\mathcal{R}^{ql}_{\mathcal{T}}$	$\mathcal{O} \models c_1 \sqsubseteq_C c_3,\ \mathcal{O} \models c_3 \sqsubseteq_C \exists r_2.c_2 \ \Rightarrow\ \mathcal{O} \models c_1 \sqsubseteq_C \exists r_2.c_2$	$\texttt{isacCR}(c_1,r_2,c_2)\texttt{:-isacCC}(c_1,c_3)\texttt{,isacCR}(c_3,r_2,c_2)$
$\mathcal{R}^{ql}_{\mathcal{T}}$	$\mathcal{O} \models r_1 \sqsubseteq_R r_2,\ \mathcal{O} \models irr(r_2) \ \Rightarrow\ \mathcal{O} \models irr(r_1)$	$\texttt{irref}(r_1)\texttt{:-isarRI}(r_1,r_2)\texttt{,irref}(r_2)$
$\mathcal{R}^{ql}_{\mathcal{A}}$	$\mathcal{O} \models r_2(y,x),\ \mathcal{O} \models r_2 \sqsubseteq_R r_1^- \ \Rightarrow\ \mathcal{O} \models r_1(x,y)$	$\texttt{instr}(r_1,x,y)\texttt{:-instr}(r_2,y,x)\texttt{,isarRI}(r_2,r_1)$
$\mathcal{R}^{ql}_{\mathcal{A}}$	$\mathcal{O} \models \exists y\, r_1(x,y),\ \mathcal{O} \models \exists r1 \sqsubseteq_C c2 \ \Rightarrow\ \mathcal{O} \models c_2(x)$	$\texttt{instc}(c_2,x)\texttt{:-existR}(r_1,x)\texttt{,isacRC}(r_1,c_2)$
$\mathcal{R}^{ql}_{\neq}$	$\mathcal{O} \models r_1 \sqsubseteq_R \neg r_2,\ \mathcal{O} \models r_2(x,z),\ \mathcal{O} \models r_1(y,z) \ \Rightarrow\ \mathcal{O} \models x \neq y$	$\texttt{diff}(x,y)\texttt{:-disjrRR}(r_1,r_2)\texttt{,instr}(r_1,x,z)\texttt{,instr}(r_2,y,z)$
$\mathcal{R}^{ql}_{\neq}$	$\mathcal{O} \models irr(r),\ \mathcal{O} \models r(x,y) \ \Rightarrow\ \mathcal{O} \models x \neq y$	$\texttt{diff}(x,y)\texttt{:-instr}(r,x,y)\texttt{,irref}(r)$
$\mathcal{R}^{ql}_{\exists}$	$\mathcal{O} \models r(x,y) \ \Rightarrow\ \mathcal{O} \models \exists z\, r(x,z)$	$\texttt{existR}(r,x)\texttt{:-instr}(r,x,y)$
$\mathcal{R}^{ql}_{\exists}$	$\mathcal{O} \models c_1(x), c_1 \sqsubseteq_C \exists r_2^-.c_2 \ \Rightarrow\ \mathcal{O} \models \exists z\, r_2(z,x)$	$\texttt{existI}(r_2,x)\texttt{:-instc}(c_1,x)\texttt{, isacCI}(c_1,r_2,c_2)$

Table 1: Examples of inference rules in R^{ql}, and corresponding Datalog rules in $\mathcal{P}^{ql}$

c_1, c_2 are atomic classes, and r_2 is an atomic object property, are translated into tuples (c_1, r_2, c_2) of the relation `isacCI`. Third, we use τ to translate each inference rule to a Datalog rule over the predicates of S^{ql} and obtain a set of Datalog rules $\mathcal{P}^{ql}$.

Because of the lack of space, we cannot report on the whole set $\mathcal{R}^{ql}$ of OWL 2 QL inference rules, but we next provide an overview on the main sets of rules it consists of. In particular, $\mathcal{R}^{ql}$ consists of the sets of rules $\mathcal{R}^{ql}_{\mathcal{T}}$, $\mathcal{R}^{ql}_{\mathcal{A}}$, $\mathcal{R}^{ql}_{\neq}$, and $\mathcal{R}^{ql}_{\exists}$. The former allow deriving logically implied TBox axioms, ground ABox axioms, and inequalities, respectively, while the latter allow deriving first-order assertions that are not expressible in OWL 2 QL but can be logically implied by $\mathcal{O}$, such as assertions of the form $\exists y \mid r(e, y)$, or $\exists y \mid r(y, e)$. In the left column of Table 1, we provide examples of OWL 2 QL rules belonging to each of the sets mentioned above.

Then, we define, in Table 2, the translation function τ from the set of OWL 2 QL axioms and atoms to the set of atoms over the schema $\mathcal{S}^{ql}$, where c, c_1, c_2, r_1, r_2 denote elements of V_N, x, y variables in $\mathcal{V}$ and S^{ql} consists of the predicates occurring in the columns labeled with $\tau(\alpha)$ and of the binary predicates `existR` and `existI`.

α	$\tau(\alpha)$	α	$\tau(\alpha)$
$c_1 \sqsubseteq_C c_2$	$\texttt{isacCC}(c_1, c_2)$	$c_1 \sqsubseteq_C \neg c_2$	$\texttt{disjcCC}(c_1, c_2)$
$c_1 \sqsubseteq_C \exists r_2.c_2$	$\texttt{isacCR}(c_1, r_2, c_2)$	$c_1 \sqsubseteq_C \neg\exists r_2$	$\texttt{disjcCR}(c_1, r_2)$
$c_1 \sqsubseteq_C \exists r_2^-.c_2$	$\texttt{isacCI}(c_1, r_2, c_2)$	$c_1 \sqsubseteq_C \neg\exists r_2^-$	$\texttt{disjcCI}(c_1, r_2)$
$\exists r_1 \sqsubseteq_C c_2$	$\texttt{isacRC}(r_1\ \ c_2)$	$\exists r_1 \sqsubseteq_C \neg c_2$	$\texttt{disjcRC}(r_1, c_2)$
$\exists r_1 \sqsubseteq_C \exists r_2.c_2$	$\texttt{isacRR}(r_1, r_2, c_2)$	$\exists r_1 \sqsubseteq_C \neg\exists r_2$	$\texttt{disjcRR}(r_1, r_2)$
$\exists r_1 \sqsubseteq_C \exists r_2^-.c_2$	$\texttt{isacRI}(r_1, r_2, c_2)$	$\exists r_1 \sqsubseteq_C \neg\exists r_2^-$	$\texttt{disjcRI}(r_1, r_2)$
$\exists r_1^- \sqsubseteq_C c_2$	$\texttt{isacIC}(r_1\ \ c_2)$	$\exists r_1^- \sqsubseteq_C \neg c_2$	$\texttt{disjcIC}(r_1, c_2)$
$\exists r_1^- \sqsubseteq_C \exists r_2.c_2$	$\texttt{isacIR}(r_1, r_2, c_2)$	$\exists r_1^- \sqsubseteq_C \neg\exists r_2$	$\texttt{disjcIR}(r_1, r_2)$
$\exists r_1^- \sqsubseteq_C \exists r_2^-.c_2$	$\texttt{isacII}(r_1, r_2, c_2)$	$\exists r_1^- \sqsubseteq_C \neg\exists r_2^-$	$\texttt{disjcII}(r_1, r_2)$
$r_1 \sqsubseteq_R r_2$	$\texttt{isarRR}(r_1, r_2)$	$r_1 \sqsubseteq_R r_2^-$	$\texttt{isarRI}(r_1, r_2)$
$r_1 \sqsubseteq_R \neg r_2$	$\texttt{disjrRR}(r_1, r_2)$	$r_1 \sqsubseteq_R \neg r_2^-$	$\texttt{disjrRI}(r_1, r_2)$
$refl(r)$	$\texttt{refl}(r)$	$irr(r)$	$\texttt{irref}(r)$
$c(x)$	$\texttt{instc}(c, x)$	$r(x, y)$	$\texttt{instr}(r, x, y)$
$x \neq y$	$\texttt{diff}(x, y)$		

Table 2: Function τ

Finally, $\mathcal{P}^{ql}$ consists of the set of Datalog rules obtained by using τ to translate each OWL 2 QL inference rule. In particular, we use the predicates `existR` and `existI` to encode, respectively, assertions of the form $\exists y \mid r(e, y)$ and $\exists y \mid r(y, e)$, which can be logically implied by OWL 2 QL ontologies but cannot be expressed in OWL 2 QL. The key point is that the use of such predicates allows us to avoid introducing existential variables within the head of the rule. Thus, the above mentioned assertions are translated into the tuples $\texttt{existR}(r, e)$ and $\texttt{existI}(r, e)$. Note that this, together with the fact that we consider inequalities and we interpret $\mathcal{O}$ under a more general semantics, significantly distinguishes our reduction to Datalog from the one proposed in [5]. In Table 1, we provide the translation by τ of examples of OWL 2 QL inference rules belonging to each of the sets $\mathcal{R}^{ql}_{\mathcal{T}}$, $\mathcal{R}^{ql}_{\mathcal{A}}$, $\mathcal{R}^{ql}_{\neq}$, and $\mathcal{R}^{ql}_{\exists}$.

We are now ready to show how both the problems of checking the consistency of $\mathcal{O}$ under MS and of answering q over $\mathcal{O}$ under MSER can be reduced to the evaluation a Datalog program. Note that, as customary in Datalog, we assume to deal with programs including a "special" intensional predicate, called *answer predicate*, here denoted Ans, and we assume that, given an instance D of a schema $\mathcal{S}$, the answer to a program Π over D, denoted $\Pi(D)$, is the extension of the answer predicate within the instance D' of $\mathcal{S}$ that results from the evaluation of Π over D.

Thus, let $D^{\mathcal{O}}$ be the set of facts computed by applying τ to every axiom in $\mathcal{O}$. Obviously, by construction, $D^{\mathcal{O}}$ is an instance of S^{ql} over elements of V_N. Also, let $\mathcal{P}^{inc}$ consist of the following rules:

```
Ans():-instc(x,z),instc(y,z),disjc(x,y),
Ans():-instr(x,z,v),instc(y,z,v),disjr(x,y),
Ans():-instr(x,z,z),irref(x),
Ans():-⊥C(x),
Ans():-⊥R(x)
```

Intuitively, the rules of $\mathcal{P}^{inc}$ check for the existence of specific *violation patterns* in $D^{\mathcal{O}}$. Then, we have the following.

Theorem 3.1. *An ontology $\mathcal{O}$ is consistent under MS iff $(\mathcal{P}^{ql} \cup \mathcal{P}^{inc})(D^{\mathcal{O}}) = \emptyset$.*

Now, let q be a query over $\mathcal{O}$ and let r^q be the rule

$$\texttt{Ans}(\vec{x})\texttt{:-}\tau(body(q)),$$

where, with a little abuse of notation, we denote by $\tau(B)$ the conjunction of the assertions obtained by applying τ to each atom in B. It is easy to see that, by construction, $\tau(B)$ is expressed over the alphabet of S^{ql} and involve only elements of V_N and $\mathcal{V}$. Also, let $\Pi^q = \mathcal{P}^{ql} \cup \mathcal{P}^q$.

Theorem 3.2. *Let $\mathcal{O}$ be a consistent ontology and q a conjunctive query. Then,*

$$Ans(q, \mathcal{O}) = \Pi^q(D^{\mathcal{O}}).$$

In a nutshell, the correctness of the two theorems relies on (*i*) the soundness and completeness of the set of OWL 2 QL inference rules, (*ii*) the semantics of MSER which treats existential variables as if they were bound, and, finally, (*iii*) the property that for every tuple t of elements of V_N, that is an instance of a predicate in $S^{ql} \setminus \{\texttt{existR}, \texttt{existI}\}$, t belongs to the minimum model of the Datalog program if and only if either $\mathcal{P}^{inc} \neq \emptyset$ and O is unsatisfiable, or t can be translated, via the inverse function of τ, into an axiom that belongs to every model of the ontology.

Clearly, the above theorems provide algorithms that are PTIME in the size of the ontology, and can be readily used by exploiting any off-the-shelf Datalog engine.

4 EXPERIMENTS

As a proof of concept, we report on a series of experiments on two implementations of the query answering technique presented in the previous section, using the Datalog engines RDFox, and LogicBlox.

RDFox[3] is a state-of-the-art triple store working in main memory which exploits parallelism and shared memory for Datalog reasoning using the Skolem chase. Since it does not allow for *n*-ary predicates, following a trick suggested by the developers [13], we use "reified" such predicates: i.e., a rule `R(x,y,z), S(z,u,v) => T(z,y,w)` is rewritten as `R1(t1,x), R2(t1,y), R3(t1,z), S1(t2,z), S2(t2,u), S3(t2,v), BIND(SKOLEM("f",z,y,w) AS t3)) => T1(t3,z), T2(t3,y), T3(t3,w)`, where the atom `BIND(SKOLEM("f",z,y,w) AS t3` is used to assign to `t3` a new value that uniquely depends on `z`, `y`, and `w`, by exploiting the `SKOLEM()` built-in function which simulates function symbols in a limited way (it does not "reconcile" variables, guaranteeing termination).

LogicBlox[4], instead, is a state-of-the-art Datalog engine that allows for n-ary predicates, uses a second-memory database, and incorporates fundamental advances in Datalog query optimization.

We experimented the Datalog translation using the TBox provided by the Lehigh University Benchmark (LUBM)[5] with the removal of axioms not allowed in OWL 2 QL, and ABoxes of different sizes (based on the number of universities) generated through the data generator (UBA1.7). As queries, we considered the standard LUBM benchmark queries, except for the queries number 4 and number 8 including data properties, which for the sake of simplicity, we did not include in our proof of concept. We also considered the metaqueries q_1,q_2, q_4, and q_{10} of the paper [8], used also in [9]. For example, the metaquery q_{10} selects the type of faculty (i.e., professor, associate professor, etc.) to which belongs a person and the type of work-relationship existing between such person and some organization to which belongs at least one of her graduate students; namely, q_{10} is the following:

select ?c ?p **where**{ ?c(?y). ?c $\sqsubseteq_C$ Faculty.
?p $\sqsubseteq_R$ worksFor. ?p(?y, ?w). advisor(?x, ?y).
GraduateStudent(?x). memberOf(?x, ?w) }

We ran all experiments on a standard laptop with Intel i5 5200U @2.20Ghz processor with 8Gb of RAM. In Table 3 are reported the results for the two systems, for two different sizes of the ABox, corresponding to 1 and 9 universities. Times are in milliseconds and concern, on one hand, the construction of the model for the Datalog program and, on the other hand, the evaluation of each query over the model. The experiments show that it is indeed "feasable" to solve query answering under MSER following our approach, in the sense, that the query that takes longer to be evaluated, namely mq_1, takes up to 20 seconds using LogicBlox, which is not bad if one considers that we did not make any attempt to customize the Datalog program to exploit at best the optimization strategies of any of the two Datalog engines. Also, since RDFox works in main-memory, in general, it performs better than LogicBlox, with the exception of queries involving several *n*-ary predicates (see, e.g., the evaluation time of q_2), whose evaluation suffers from the use of reification. On the other hand, using RDFox, we had significant limits in increasing the size of the ABox.

In order to measure the impact of conjunctive query answering under MSER, we carried two other series of experiments. On one hand, we compared our times with those required by state-of-the-art OWL 2 QL reasoners, such as Mastro [2] and Ontop [1], to evaluate the LUBM benchmark queries under DSER. To this aim, since Mastro and Ontop interpret queries under the standard first-order semantics, we modified the queries by making distinguished all variables occurring in the body. Unsurprisingly, our experiments showed that current OWL 2 QL reasoners perform better. Obviously, the comparison could concern only the LUBM queries, since they are first-order, while our implementation allows for significantly extending the set of legal queries, by handling metaqueries without typing constraints and inequalities.

Second, in order to measure the impact of the usage of the `DifferentIndividuals` atoms in queries, as well as of the usage of metamodeling capabilities associated with the query language that extract information spanning the various levels of an ontology (including the possibility of joining variable in different position), we extended the LUBM ontology with 9 Universities with the new entity name `:TypeOfProfessor` and the following sets of axioms: (*i*) a set of ABox axioms stating that `:FullProfessor`, `:AssociateProfessor`, and `:AssistantProfessor` are instances of `:TypeOfProfessor`, and (*ii*) a set of TBox axioms stating the pairwise disjointness of the classes `:FullProfessor`, `:AssociateProfessor`, and `:AssistantProfessor`. Then, we considered the query asking for all (x, y) such that x is a professor, y is a type of professor, and x is an instance of y. Note that this is a metaquery violating the typing constraint. Finally, we considered a query involving an inequality, asking for all pairs of different professors working in a given department. The experiments confirmed that the usage of `DifferentIndividuals` atoms in queries and of metamodeling capabilities has really no negative impact at all in terms of time for the construction of the model and for the evaluation of the queries. Indeed, the time for the construction of the model required only few milliseconds more, and the times of the query evaluation for the two queries were in line with those in Table 3 (e.g., for the RDFox implementation the times are 317 ms for the first query, and 514 ms for the second query).

5 RELATED WORK.

Related to our work are results on query answering in *DL-Lite*$_{\mathcal{R}}$ [3, 7], i.e., the logic underpinning OWL 2 QL, and, in particular, results

[3]https://www.cs.ox.ac.uk/isg/tools/RDFox/
[4]http://www.logicblox.com/
[5]http://swat.cse.lehigh.edu/onto/univ-bench.owl

	Model	q_1	q_2	q_3	q_5	q_6	q_7	q_9	q_{10}	q_{11}	q_{12}	q_{13}	q_{14}	mq_1	mq_2	mq_4	mq_{10}
LUBM(1)$_{\text{RDFox}}$	45607	10	3191	4	26	15	227	41	7	0	0	2	139	773	6	19	0
LUBM(9)$_{\text{RDFox}}$	466142	79	14499	33	62	2931	1203	384	69	0	0	0	3066	9621	69	231	69
LUBM(1)$_{\text{LogicBlox}}$	12961	240	275	237	250	388	250	299	239	230	236	226	392	1901	254	249	341
LUBM(9)$_{\text{LogicBlox}}$	14678427	235	3048	226	242	2354	4127	819	228	250	246	221	2364	20545	241	319	878

Table 3: Times in milliseconds for RDFox and LogicBlox implementations

of [6] showing that answering conjunctive queries with inequalities in *DL-Lite*$_{\mathcal{R}}$ is in general undecidable. We point out, however, that such a negative result is due to the fact that existential variables are assigned the standard logical meaning, which is not the case in SPARQL queries over OWL 2 theories, interpreted under DSER, where existential variables are to be bound to IRIs occurring within the OWL 2 theory in oder to give rise to solution mappings, and thus to query answers. Recent works [5, 9–11] have investigated the problem of answering SPARQL queries over OWL 2 QL theories under DSER. However, none of such works considers queries possibly containing inequalities. Moreover, since all such works interpret ontologies according to DS, they do not consider queries that violate the typing constraint. In fact, it was shown in [14], that answering queries with inequalities over OWL 2 QL ontologies under DSER is decidable and can be reduced to the evaluation of a Datalog program. Note, however, that [14] does not consider metamodeling.

Very relevant to our study are the works [10, 11], where the authors tackle the problem of querying OWL 2 QL theories comprising metaclasses and metaproperties, as allowed by the OWL 2 standard, and, hence, to overcome the limitations of DS, introduce and use MS. In particular, their study show that the problem of answering untyped queries is in general intractable (in data complexity), even in the absence of inequalities. However, they consider answering unions of conjunctive queries over OWL 2 QL ontologies under a variant of MSER, called Metamodeling Semantics Entailment Regime with existential variables (MSER$_\exists$), which differs from DSER not only because of the semantics used to interpret the queried ontology and the removal of the typing constraints (as it is the case for MSER), but also because it assigns to existential variables and union the classical logical meaning. Note in particular that, a union of conjunctive queries is true under MSER$_\exists$ if for every model, there exists at least one query in the union that is true, while it is true under MSER if there there exists at least one query in the union that is true in every model. Thus, while MSER$_\exists$ allows for more expressive queries, MSERis therefore closer in spirit to DSER, which is the official W3C semantics for interpreting queries over OWL 2 ontologies. Actually, MSER is a generalization of DSER that extends it in order to properly handle both metamodeling (by adopting MS) and metaquerying (by removing the typing constraint).

6 CONCLUSIONS

In this work we have investigated the problem of querying OWL 2 QL theories with SPARQL without any restriction, thus, using TBox and ABox atoms (including inequalities) possibly violating the typing constraint imposed by DSER. We showed that the problem can be reduced to the evaluation of a Datalog program, thus providing the first "ready-to-use" algorithm for querying OWL 2 QL theories with SPARQL without any restriction. Note that, as an aside, our approach provides the first algorithm allowing to answer queries with inequalities over OWL 2 QL ontologies under DSER as well.

We plan to continue our work by performing more experiments to evaluate our approach, e.g., by comparing the performances of our algorithm, ran over OWL 2 QL ontologies including meta-level axioms, with the performances of the algorithm proposed in [10] for metaquerying in OWL 2 QL under MSER$_\exists$.

ACKNOWLEDGEMENTS

We gratefully acknowledge support by MIUR under the SIR project "MODEUS", grant n. RBSI14TQHQ, and by Sapienza under the project PRE-O-PRE.

REFERENCES

[1] Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao. 2017. Ontop: Answering SPARQL Queries over Relational Databases. *Semantic Web Journal* 8, 3 (2017), 471–487. https://doi.org/10.3233/SW-160217 Semantic Web Journal outstanding paper award for 2016.

[2] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Mariano Rodriguez-Muro, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. 2011. The Mastro System for Ontology-based Data Access. *Semantic Web J.* 2, 1 (2011), 43–53.

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

[4] Birte Glimm. 2011. Using SPARQL with RDFS and OWL Entailment. In *Reasoning Web. Semantic Technologies for the Web of Data – 7th Int. Summer School Tutorial Lectures (RW).* 137–201.

[5] Georg Gottlob and Andreas Pieris. 2015. Beyond SPARQL under OWL 2 QL Entailment Regime: Rules to the Rescue. In *Proc. of the 24th Int. Joint Conf. on Artificial Intelligence (IJCAI).* 2999–3007.

[6] Víctor Gutiérrez-Basulto, Yazmin Angélica Ibáñez-García, Roman Kontchakov, and Egor V. Kostylev. 2015. Queries with negation and inequalities over lightweight ontologies. *J. of Web Semantics* 35 (2015), 184–202.

[7] Stanislav Kikot, Roman Kontchakov, and Michael Zakharyaschev. 2012. Conjunctive Query Answering with OWL 2 QL. In *Proc. of the 13th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR).* 275–285.

[8] Ilianna Kollia and Birte Glimm. 2013. Optimizing SPARQL Query Answering over OWL Ontologies. *J. Artif. Intell. Res. (JAIR)* 48 (2013), 253–303.

[9] Roman Kontchakov, Martin Rezk, Mariano Rodriguez-Muro, Guohui Xiao, and Michael Zakharyaschev. 2014. Answering SPARQL Queries over Databases under OWL 2 QL Entailment Regime. In *Proc. of the 13th Int. Semantic Web Conf. (ISWC).* 552–567.

[10] Maurizio Lenzerini, Lorenzo Lepore, and Antonella Poggi. 2016. Answering Metaqueries over Hi (OWL 2 QL) Ontologies. In *Proc. of the 25th Int. Joint Conf. on Artificial Intelligence (IJCAI).* 1174–1180.

[11] Maurizio Lenzerini, Lorenzo Lepore, and Antonella Poggi. 2016. A Higher-Order Semantics for Metaquerying in OWL 2 QL. In *Proc. of the 15th Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR).* 577–580.

[12] Hector J. Levesque. 1984. Foundations of a Functional Approach to Knowledge Representation. *Artificial Intelligence* 23 (1984), 155–212.

[13] Boris Motik. 2016. On treating n-ary predicates in RDFox. (2016). Personal communication.

[14] Antonella Poggi. 2016. On the SPARQL Direct Semantics Entailment Regime for OWL 2 QL. In *Proceedings of the 29th International Workshop on Description Logics, Cape Town, South Africa, April 22-25, 2016.*