On the SPARQL Direct Semantics Entailment Regime for OWL 2 QL

Antonella Poggi

Dipartimento di Scienze Documentarie,
Linguistico-Filologiche e Geografiche -
Sapienza Università di Roma
Viale Regina Elena 295, I-00185 Roma, Italy
poggi@dis.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 query answering is required to be especially efficient in the size of such data, namely AC^0 in data complexity. On the other hand, the syntax and the semantics of the SPARQL query language for OWL 2 is defined by means of the Direct Semantics Entailment Regime (DSER), which considers queries including any assertion expressible in the language of the queried ontology, i.e., both ABox atoms, TBox atoms and inequalities expressed by means of DifferentIndividuals atoms. Thus, in this paper, we investigate query answering over OWL 2 QL under DSER. In particular, we show that, by virtue of the restricted meaning assigned to existential variables and union, query answering can be reduced to the evaluation of a Datalog program. Finally, we investigate query answering under a new SPARQL entailment regime, called Direct Semantics Answering Regime (DSAR), obtained by modifying DSER in such a way that existentially quantified variables are assigned the classical logical meaning, and provide an algorithm for answering queries over OWL 2 QL ontologies under DSAR, that is AC^0 in data complexity, for a class of queries comprising both TBox atoms, ABox atoms and inequalities.

1 Introduction

OWL 2 QL is the OWL 2 profile especially designed to provide efficient query answering over Ontology-Based Data Access (OBDA) scenarios. In particular, in such a context, the aim is to access through an ontology a typically huge amount of data residing in external data sources. Hence, OWL 2 QL is based on $DL\text{-}Lite_{\mathtt{R}}$, that is a logic for which answering unions of conjunctive queries can be reduced to answering first-order queries, and thus is AC^0 with respect to the size of the data. We point out that OWL 2 QL does not impose the *Unique Name Assumption (UNA)*. Though, it allows specifying inequalities between IRIs, by means of DifferentIndividuals axioms, i.e., axioms of the form DifferentIndividuals(e_1 e_2) imposing that e_1 and e_2 denote distinct objects of the domain.

SPARQL 1.1 is the W3C query language that is considered as the de-facto standard query language for OWL 2. In particular, the syntax and the semantics of SPARQL queries over OWL 2 QL ontologies are defined by means of the so-called *Direct Semantics Entailment Regime (DSER)* [7]. In a nutshell, DSER restricts the syntax of *legal conjunctive queries* to conjunctions of assertions, that are legal OWL 2 QL assertions, possibly including variables in object or predicate position. As for the semantics, intuitively, DSER defines the answers to a conjunctive 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. Finally, the set of answers to a union of conjunctive queries is defined as the union of the answers to each single query.

In this paper, we investigate the problem of answering queries OWL 2 QL ontologies under DSER. Therefore, queries may contain both OWL 2 QL TBox atoms, e.g., of the form SubClassOf(x_1 x_2), and ABox atoms, e.g., of the form ClassAssertion(x_1 x_2). Furthermore, queries may comprise inequalities, expressed by means of DifferentIndividuals atoms. It is worth noting that the presence of inequalities within queries deserves special attention. Indeed, inequalities between ontology entities can be logically implied by an OWL 2 QL ontology. As a simple example consider the ontology consisting of the following axioms:

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

and suppose that we want to retrieve all classes that contain at least two distinct instances. The formulation of this query in SPARQL is the following:

```
select $x where{ SubClassOf($x owl:Thing).
                 ClassAssertion($x $y).ClassAssertion($x $z).
                 DifferentIndividuals($y $z)}
```

Note that this query is legal under DSER. Also, it is easy to see that, since :Male and :Female are disjoint, at least two individuals among :p, :petra and :peter denote distinct domain objects in every model. Hence, the answer to the query is {:Person,owl:Thing}. This example clearly shows that inequalities can be inferred and hence the presence of inequalities within queries requires to reason about them.

The goal of our work is to investigate query answering over OWL 2 QL ontologies under DSER, considering the whole class of queries considered legal under DSER. Note that, as far as we know, this is the first work that considers queries comprising, in particular, DifferentIndividuals atoms. Furthermore, since as already observed in [13], the current definition of DSER makes query answering over OWL 2 QL ontologies rather limited, we investigate a revised version of DSER, called *Direct Semantics Answering Regime (DSAR)*, which, in a nutshell, assigns to existentially quantified variables the classical logical meaning.

The main contributions of this paper can be summarized as follows:

- We show that it is possible to reduce query answering over OWL 2 QL ontologies under DSER to the evaluation of a Datalog program, which provides a practical algorithm that can benefit of commercial highly optimized Datalog engines (e.g., [2], [1]).

– Through the definition of DSAR we introduce a new semantics for `SPARQL` queries that is closer to classical first-order semantics and provide a query answering algorithm over `OWL 2 QL` ontologies under DSAR, for a class of queries comprising both TBox atoms, ABox atoms and `DifferentIndividuals` atoms, that is AC^0 in data complexity.

Related work. The problem of answering unions of conjunctive queries over Description Logics ontologies under the classical first-order semantics, was extensively studied in the literature (e.g. [14, 8]). Relevant to our work are results on query answering in *DL-Lite*$_{\mathtt{R}}$ [6, 4, 12], i.e., the logic underpinning `OWL 2 QL`, and, in particular, results of [11] showing that answering conjunctive queries with inequalities in *DL-Lite*$_{\mathtt{R}}$ is in general undecidable. We point out, however, that the problem of answering unions of conjunctive queries over `OWL 2 QL` ontologies under both DSER and DSAR differs from the problem of answering unions of conjunctive queries over *DL-Lite*$_{\mathtt{R}}$ ontologies under the standard first-order semantics. The main differences between such problems are the following:

1. queries that are legal under DSER and DSAR may contain TBox atoms, while answering of unions of conjunctive queries under the standard first-order semantics assumes that queries contain ABox atoms only,
2. under DSER and DSAR, the union is an operator that is applied to merge the certain answers to each query in the union, while under the standard first-order semantics the answers to a query Q that is a union of queries are the certain answers to the union, i.e., the intersection of the answers to Q evaluated over every model of the ontology,
3. under DSER, existential variables are handled as if they were target variables, in the sense that they can be bound to `IRI`s only, while under DSAR and under the standard first-order semantics, existential variables are assigned the classical logical meaning, i.e., they can be bound to a distinct domain element in every model.

Few recent works [13, 3, 9] have investigated the problem of answering `SPARQL` queries over `OWL 2 QL` ontologies under DSER. However, none of such works considers queries possibly containing `DifferentIndividuals`. In fact, in [9] the authors even claim that inequalities cannot occur within legal `OWL 2 QL` conjunctive queries, which is not true, since inequalities can in fact be expressed by means of `DifferentIndividuals` atoms. Finally, it is worth noting that the problem of reducing reasoning over Description Logics ontologies to the evaluation of a logic program has been studied in [10, 15]. In particular, such works are at the basis of the `OWL 2` RL profile[1], especially designed to provide reasoning via rule-based technologies. However, to the best of our knowledge, no algorithm

[1] `https://www.w3.org/TR/owl2-profiles/#OWL_2_RL`

was proposed for answering queries over OWL 2 RL ontologies under DSER, that considers queries possibly containing TBox and DifferentIndividuals atoms.

The paper is organized as follows. After providing, in Section 2, some preliminaries on the languages considered in the paper, in Section 3 we investigate query answering under DSER. Then, in Section 4, we introduce DSAR and study query answering under DSAR. Finally, in Section 5, we conclude the paper and discuss future work.

2 Preliminaries

In this section we briefly recall the OWL 2 QL ontology language and the query language. We express ontologies and queries in the extended functional style syntax [7].

Ontology *entities*, such as individuals, classes, object properties, etc., are denoted by *expressions*. *Atomic expressions* are denoted simply by a name, where the set of names coincides with the set IRI of *Internationalized Resource Identifiers*. *Complex expressions* are built on atomic expressions using the OWL 2 QL constructors, such as ObjectSomeValuesFrom, ObjectInverseOf, etc. For example, if e_1, e_2 are expressions, then ObjectSomeValuesFrom(e_1 e_2) is a complex expression. Values are denoted by *literals* where a literal (l, d) consists of a lexical value l and a datatype d. Let $\mathtt{D} = \{\mathtt{rdfs:Literal}\} \cup \mathtt{D}'$ denote the set of datatypes admitted in OWL 2 QL and $\mathtt{L}^{ql} = \{(l, d) \mid d \in \mathtt{D}'\}$ the set of literals admitted in OWL 2 QL.

An OWL 2 QL ontology (simply *ontology* in the following) is a finite set of OWL 2 QL *logical axioms*. Logical axioms are classified into (*i*) *TBox axioms*, i.e., SubClassOf, SubObjectPropertyOf, SubDataPropertyOf, ReflexiveObjectProperty, DataPropertyRange, DisjointClasses, DisjointObjectProperties, DisjointDataProperties, and IrreflexiveObjectProperty axioms, and (*ii*) *ABox axioms*, i.e., ClassAssertion, ObjectPropertyAssertion, DataPropertyAssertion, and DifferentIndividuals axioms. Axioms not in the above list can be expressed by appropriate combinations of the ones listed.

Given an ontology $\mathcal{O}$ we denote with $V_N^{\mathcal{O}}$ the set of atomic expressions occurring in $\mathcal{O}$ extended with the OWL 2 QL reserved vocabulary. An expression is said to appear in *class position* in $\mathcal{O}$ if it appears in a SubClassOf or DisjointClasses axiom, or in the first position of a ClassAssertion axiom. Similarly, we can define the notions of *object property*, *data property*, *data type*, *individual*, and *value position*. Without loss of generality, we assume that an expression cannot appear in positions of different types within the same ontology[2]. Finally,

[2] Note that even though OWL 2 punning allows the same entity to occur in positions of different types, here we adopt the Direct Semantics for OWL 2, which interprets as distinct names each such occurrence. That is why, without loss of generality, we can assume that an expression cannot appear in positions of different types.

- If e occurs in individual position in $\mathcal{O}$, then $e^I \in {\Delta_O}^I$;
- If e occurs in class position in $\mathcal{O}$, then $e^I \subseteq {\Delta_O}^I$; in particular:
  - $\texttt{owl:Thing}^I = {\Delta_O}^I$;
  - $\texttt{owl:Nothing}^I = \emptyset$;
  - $\texttt{ObjectSomeValuesFrom}(e_1\ e_2)^I = \{d_1 \mid \langle d_1, d_2\rangle \in e_1^I, d_2 \in e_2^I\}$;
  - $\texttt{DataSomeValuesFrom}(e_1\ e_2)^I = \{d_1 \mid \langle d_1, d_2\rangle \in e_1^I, d_2 \in e_2^I\}$;
- if e occurs in object property position in $\mathcal{O}$, then $e^I \subseteq ({\Delta_O}^I \times {\Delta_O}^I)$; in particular:
  - $\texttt{owl:topObjectProperty}^I = ({\Delta_O}^I \times {\Delta_O}^I)$;
  - $\texttt{owl:bottomObjectProperty}^I = \emptyset$;
  - $\texttt{ObjectInverseOf}(e)^I = (e^I)^{-1}$;
- if e occurs in data property position in $\mathcal{O}$, then $e^I \subseteq ({\Delta_O}^I \times {\Delta_V}^I)$; in particular:
  - $\texttt{owl:topDataProperty}^I = ({\Delta_O}^I \times {\Delta_V}^I)$;
  - $\texttt{owl:bottomDataProperty}^I = \emptyset$;
- if e occurs in datatype position in $\mathcal{O}$, then $e^I \subseteq ({\Delta_V}^I)$; in particular:
  - $\texttt{rdfs:Literal}^I = \Delta_v$;
  - $D_1^I = D_1^{DT}$, for every $D_1 \in \mathtt{D}'$, where the function $\cdot^{DT}$ is the function defined in the OWL 2 QL datatype map, that assigns to each $d \in \mathtt{D}'$ the set of data values represented by d in all OWL 2 ontologies;
- If (l, d) in $\mathtt{L_{QL}}$, then $(l, d)^I = (l, d)^{LS}$, where $\cdot^{LS}$ is the function defined in the OWL 2 QL datatype map, that specifies, for each literal (l, d) admitted in OWL 2 QL which is the data value in d^{DT} denoted by such literal.

Table 1. Conditions for $\mathcal{I}$

we denote with $Exp^{\mathcal{O}}$ the *finite* set of all *well-formed expressions* that can be built on the basis of $V_N^{\mathcal{O}}$. For example, if e_1, e_2 are well-formed expressions, then $\texttt{ObjectSomeValuesFrom}(e_1 e_2)$ is a well-formed expression too.

In this paper we are interested in interpreting OWL 2 QL ontologies according to the Direct Semantics (DS)[3]. Hence, the semantics of OWL 2 QL is given by means of the usual notion of interpretation. Specifically, an *interpretation for* $\mathcal{O}$ is a pair $\mathcal{I} = (\Delta^I, \cdot^I)$, where:

- Δ^I is the disjoint union of the non-empty set ${\Delta_O}^I$, i.e., the *object domain*, and of the non-empty set Δ_v, i.e., the *value domain* consisting of all data values that, according to the OWL 2 QL datatype map[4], can be represented by literals in OWL 2 QL, and
- $\cdot^I$ is the *interpretation function*, i.e., a function that maps every expression in $Exp^{\mathcal{O}}$ and every literal in $\mathtt{L_{QL}}$ according to the set of conditions specified in Table 1.

The semantics of logical axioms is based on the usual notion of *satisfaction of an axiom* with respect to an interpretation I. Thus, for example,

[3] https://www.w3.org/TR/owl2-direct-semantics/
[4] https://www.w3.org/TR/2012/REC-owl2-syntax-20121211/#Datatype_Maps

$I \models$ `SubClassOf`$(e_1\ e_2)$ if $e_1^I \subseteq e_2^I$, and $I \models$ `DifferentIndividuals`$(e_1\ e_2)$ if $e_1^I \neq e_2^I$. Also, we say that an interpretation is a *model* of $\mathcal{O}$ if it satisfies all axioms in $\mathcal{O}$ and we say that $\mathcal{O}$ is *satisfiable* if there exists at least an interpretation that is a model of $\mathcal{O}$. Finally, if α is an axiom, we say that α is *logically implied* by an ontology $\mathcal{O}$, written $\mathcal{O} \models \alpha$, if $M \models \alpha$ for every model M of $\mathcal{O}$.

As for the query language, we rely on the notion of `SPARQL` *entailment regime*, as defined in the W3C standard specification[5]. 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 queries, i.e., what are the answers to a query. In this paper, we are interested in the `OWL 2 QL` Direct Semantics Entailment Regime (DSER). Thus, as for (*i*), as already discussed above, we assume to deal with `OWL 2 QL` ontologies interpreted according to DS. As for (*ii*), legal queries are conjunctive queries, called *Basic Graph Patterns* in the `SPARQL` jargon, where a conjunctive query q is an expression of the form

$$\textbf{select}\ \$x_1 \ldots \$x_n\ \ \textbf{where}\ \ B$$

where $x_1, \ldots, x_n$ are variables, called *target variables*, belonging to an alphabet $\mathcal{V}$ disjoint from $V_N^{\mathcal{O}}$, n is the *arity* of q, and B, called *query body*, denoted $body(q)$, is a conjunction of atoms including all variables x_i, for $i \in \{1, n\}$, and possibly including entities in $V_N^{\mathcal{O}}$ and variables in $\mathcal{V}$, distinct from x_i, called *existential variables* of q. More precisely, atoms can be classified into *TBox* and *ABox* atoms, having the form of legal `OWL 2 QL` TBox and ABox axioms respectively, and possibly including variables in object or predicate position[6]. Finally, as for (*iii*), DSER defines the answers to a query as follows. Given a tuple of variables $\boldsymbol{z} = (z_1, \ldots, z_n)$, a tuple of `IRI`s $\boldsymbol{w} = (w_1, \ldots, w_n)$, and a conjunction of atoms B, we denote by $\sigma[\boldsymbol{z} \to \boldsymbol{w}](B)$ the conjunction of atoms obtained from B by substituting each z_i in $\boldsymbol{z}$ with w_i in $\boldsymbol{w}$, for $i \in \{1, \ldots, n\}$. Now, let $\mathcal{O}$ be an ontology and q a conjunctive query such that $\boldsymbol{x}$ is the n-tuple of target variables of q and $\boldsymbol{y}$ is the m-tuple of existential variables of q. We say that an n-tuple of `IRI`s $\boldsymbol{t}$ is an *answer to q over $\mathcal{O}$ under DSER* if there exists an m-tuple of `IRI`s $\boldsymbol{v}$ such that

$$\mathcal{O} \models \sigma[(\boldsymbol{x}, \boldsymbol{y}) \to (\boldsymbol{t}, \boldsymbol{v})](body(q)).$$

Since in this paper we investigate query answering under DSER where queries are *unions of conjunctive queries*, we still have to specify what are the answers to such queries. Specifically, given a query $Q = q_1 \cup q_2 \cup \ldots q_m$, *the answers to Q under DSER* are defined as the union of the set of answers to each q_i in Q.

In the next sections, we will denote by $Ans^{DSER}(Q, \mathcal{O})$ the set of answers to the query Q over the ontology $\mathcal{O}$ under DSER. Finally, we point out that, from

[5] `https://www.w3.org/TR/sparql11-entailment/`

[6] Since we assumed that the same entity name cannot occur in positions of different types within the ontology, without loss of generality, we ignore here the so-called *typing constraint*, preventing the same variable to occur in positions of different types within queries.

now on, we implicitly assume to deal with a satisfiable `OWL 2 QL` ontology $\mathcal{O}$ that does not contain any data property [7].

3 Query answering under DSER

In this section, we investigate query answering under DSER. In particular we present the algorithm *AnsByDat* that reduces query answering under DSER to the evaluation of a Datalog program.

First of all, let us introduce the database schema $\mathcal{S}^{ql}$. $\mathcal{S}^{ql}$ is defined as the union of the following sets of predicates (where p/n indicates that the predicate p has arity n):

- $\mathcal{S}^{ql,\mathcal{T}}$ = {`isacCC`/2, `isacCR`/3, `isacCI`/3, `isacRC`/2, `isacRR`/3, `isacRI`/3, `isacIC`/2, `isacIR`/3, `isacII`/3, `isarRR`/2, `isarRI`/2, `isarIR`/2, `isarII`/2, `disjcCC`/2, `disjcCR`/2, `disjcCI`/2, `disjcRC`/2, `disjcRR`/2, `disjcRI`/2, `disjcIC`/2, `disjcIR`/2, `disjcII`/2, `disjrRR`/2, `disjrRI`/2, `disjrIR`/2, `disjrII`/2, `irref`/1, `refl`/1}.
- $\mathcal{S}^{ql,\mathcal{A}}$ = {`instc`/2, `instr`/3, `existR`/2, `existI`/2, `diff`/2}

Also, we introduce a function allowing to encode `OWL 2 QL` assertions into assertions over $\mathcal{S}^{ql}$. Thus, let τ be the *encoding function* from the set of `OWL 2 QL` logical assertions, possibly including variables in $\mathcal{V}$, to the set of logical assertions over the alphabet $\mathcal{S}^{ql}$, possibly including constants in `IRI`s and variables in $\mathcal{V}$. τ is defined in Table 2, where, for brevity, we have used the usual Description Logic notation for `OWL 2 QL` assertions α. Intuitively, each predicate p in $\mathcal{S}^{ql}$ ($\mathcal{S}^{ql,\mathcal{T}}$, $\mathcal{S}^{ql,\mathcal{A}}$) encodes a specific form of `OWL 2 QL` (TBox, ABox, respectively) axiom, and for each such assertion that is either an axiom or an atom, the function τ returns respectively an instance of p or an atom with predicate p.

We are now ready to present the algorithm *AnsByDat*. Specifically, given a query Q over an ontology $\mathcal{O}$ the algorithm proceeds as follows:

1. it constructs a database $D^{\mathcal{O}}$ that is an instance of $\mathcal{S}^{ql}$ and represents the axioms of $\mathcal{O}$,
2. it builds a set $\mathcal{P}^{ql}$ of Datalog rules encoding the semantics of `OWL 2 QL` axioms, thus allowing to reason about them,
3. it builds a set $\mathcal{P}^{Q}$ of Datalog rules encoding Q whose head predicate is `Ans`, and
4. it evaluates the Datalog program $\mathcal{P} = (\mathcal{P}^{ql} \cup \mathcal{P}^{Q})$ over $D^{\mathcal{O}}$ and returns the extension of `Ans`.

More in details, the database $D^{\mathcal{O}}$ is simply obtained by applying τ to each axiom of $\mathcal{O}$. As for the program $\mathcal{P}^{ql}$, it is the union of the two sets of rules,

[7] The whole approach can be extended to capture data properties too.

α	$\tau(\alpha)$	α	$\tau(\alpha)$
$\tau(c1 \sqsubseteq c2)$	$\mathtt{isacCC}(c1, c2)$	$\tau(c1 \sqsubseteq \exists r2.c2)$	$\mathtt{isacCR}(c1, r2, c2)$
$\tau(c1 \sqsubseteq \exists r2^{-}.c2)$	$\mathtt{isacCI}(c1, r2, c2)$	$\tau(\exists r1 \sqsubseteq c2)$	$\mathtt{isacRC}(r1\ \ c2)$
$\tau(\exists r1 \sqsubseteq \exists r2.c2)$	$\mathtt{isacRR}(r1, r2, c2)$	$\tau(\exists r1 \sqsubseteq \exists r2^{-}.c2)$	$\mathtt{isacRI}(r1, r2, c2)$
$\tau(\exists r1^{-} \sqsubseteq c2)$	$\mathtt{isacIC}(r1\ \ c2)$	$\tau(\exists r1^{-} \sqsubseteq \exists r2.c2)$	$\mathtt{isacIR}(r1, r2, c2)$
$\tau(\exists r1^{-} \sqsubseteq \exists r2^{-}.c2)$	$\mathtt{isacII}(r1, r2, c2)$	$\tau(r1 \sqsubseteq r2)$	$\mathtt{isarRR}(r1, r2)$
$\tau(r1 \sqsubseteq r2^{-})$	$\mathtt{isarRI}(r1, r2)$	$\tau(r1^{-} \sqsubseteq r2)$	$\mathtt{isarIR}(r1, r2)$
$\tau(r1^{-} \sqsubseteq r2^{-})$	$\mathtt{isarII}(r1, r2)$	$\tau(c1 \sqsubseteq \neg c2)$	$\mathtt{disjcCC}(c1, c2)$
$\tau(c1 \sqsubseteq \neg\exists r2)$	$\mathtt{disjcCR}(c1, r2)$	$\tau(c1 \sqsubseteq \neg\exists r2^{-})$	$\mathtt{disjcCI}(c1, r2)$
$\tau(\exists r1 \sqsubseteq \neg c2)$	$\mathtt{disjcRC}(c1, c2)$	$\tau(\exists r1 \sqsubseteq \neg\exists r2)$	$\mathtt{disjcRR}(c1, r2)$
$\tau(\exists r1 \sqsubseteq \neg\exists r2^{-})$	$\mathtt{disjcRI}(c1, r2)$	$\tau(\exists r1^{-} \sqsubseteq \neg c2)$	$\mathtt{disjcIC}(c1, c2)$
$\tau(\exists r1^{-} \sqsubseteq \neg\exists r2)$	$\mathtt{disjcIR}(c1, r2)$	$\tau(\exists r1^{-} \sqsubseteq \neg\exists r2^{-})$	$\mathtt{disjcII}(c1, r2)$
$\tau(r1 \sqsubseteq \neg r2)$	$\mathtt{disjrRR}(r1, r2)$	$\tau(r1 \sqsubseteq \neg r2^{-})$	$\mathtt{disjrRI}(r1, r2)$
$\tau(r1^{-} \sqsubseteq \neg r2)$	$\mathtt{disjrIR}(r1, r2)$	$\tau(r1^{-} \sqsubseteq \neg r2^{-})$	$\mathtt{disjrII}(r1, r2)$
$\tau(irreflexive(r1))$	$\mathtt{irref}(r1)$	$\tau(reflexive(r1))$	$\mathtt{refl}(r1)$
$\tau(c(x))$	$\mathtt{instcC}(c, x)$	$\tau(r(x, y))$	$\mathtt{instr}(r, x, y)$
$\tau(x \neq y)$	$\mathtt{diff}(x, y)$		

Table 2. Function τ

$\mathcal{P}^{ql,\mathcal{T}}$ and $\mathcal{P}^{ql,\mathcal{A}}$, which, intuitively, are obtained by translating by means of τ the set of inference rules that are needed to compute the *closure* of the TBox and the *saturation* of the ABox, respectively. For the lack of space, we cannot report here the whole set of rules of $\mathcal{P}^{ql}$. Thus, in the following, we illustrate $\mathcal{P}^{ql}$, by considering two specific rules, namely:

```
isacCI(c1, r2, c2) :- isacCC(c1, c3), isacCI(c3, r2, c2). (*)
instr(r1, x, y) :- instr(r2, y, x), isarRI(r2, r1). (**)
```

It is not hard to see that the above rules encode the following inference rules: (*)$\forall c1, r2, c2$ if $\mathcal{O} \models c1 \sqsubseteq c3$ and $\mathcal{O} \models c3 \sqsubseteq \exists r2.c2$, then $\mathcal{O} \models c1 \sqsubseteq \exists r2.c2$, and (**)$\forall r1, x, y$ if $\mathcal{O} \models r2(y, x)$ and $\mathcal{O} \models r2 \sqsubseteq r1^{-}$, then $\mathcal{O} \models r1(x, y)$.

Let us now show how the algorithm constructs the set $\mathcal{P}^{Q}$. If Q has target variables $\boldsymbol{x}$ and is the union of the conjunctive queries $q_1, \ldots, q_p$, then $\mathcal{P}^{Q}$ is the set of rules $\{\mathtt{Ans}(\boldsymbol{x})\text{:-}\tau(body(q_i)) \mid i = 1, \ldots, p\}$ where, by an abuse of notation, we denote by $\tau(body(q_i))$ the conjunction of the assertions obtained by applying τ to each atom in $body(q_i)$.

Theorem 1. *If $\mathcal{O}$ is an ontology and Q is a union of conjunctive queries that is legal under the* OWL 2 QL *DSER, $Ans^{DSER}(Q, \mathcal{O}) = AnsByDat(Q, \mathcal{O})$.*

Proof (sketch). Let $Can(\mathcal{O})$ be the canonical model of $\mathcal{O}$. $Can(\mathcal{O})$ contains both elements belonging to $V_N^{\mathcal{O}}$ and elements belonging to a set of variables $\mathcal{V}^{sk}$. Furthermore, let $\mathcal{H}$ be the minimum Herbrand model $\mathcal{H}$ of $\mathcal{P}$ containing $D^{\mathcal{O}}$. Finally, let $Diff^{\mathcal{O}}$ be the subset of $V_I^{\mathcal{O}} \times V_I^{\mathcal{O}}$ (where $V_I^{\mathcal{O}}$ denotes the subset of elements of $V_N^{\mathcal{O}}$ that occur in individual position in $\mathcal{O}$) such that every pair

$(a,b) \in Diff^{\mathcal{O}}$ is such that a and b are interpreted as distinct domain elements in every model of $\mathcal{O}$. The proof relies on the following properties. First, $\mathcal{H}$ is such that a tuple belongs to a predicate in $\mathcal{S}^{ql,\mathcal{A}} \setminus \{\texttt{diff}\}$ if and only if a corresponding *extensional* property is satisfied in $Can(\mathcal{O})$, over elements of $V_N^{\mathcal{O}}$. For example, for every (c,x) that belongs to `instc` in $\mathcal{H}$, $x \in c^{Can(\mathcal{O})}$, where $x \in V_N^{\mathcal{O}}$. Also, for every (r,x) that belongs to `existR` in $\mathcal{H}$, there exists $y \in \mathcal{V}^{sk}$ such that $(x,y) \in r^{Can(\mathcal{O})}$, where $x \in V_N^{\mathcal{O}}$. Second, $\mathcal{H}$ is such that a tuple belongs to a predicate of $\mathcal{S}^{ql,\mathcal{T}}$ if and only if the corresponding *intensional* property is satisfied by $Can(\mathcal{O})$. Thus, for example, for every $(c1,c2)$ that belongs to `isacCC` in $\mathcal{H}$, we have that for every x, if $x \in c1^{Can(\mathcal{O})}$, then $x \in c2^{Can(\mathcal{O})}$. Also, for every (c,r) that belongs to `isacCR` in $\mathcal{H}$, we have that for every x, if $x \in c1^{Can(\mathcal{O})}$, then there exists y such that $(x,y) \in r^{Can(\mathcal{O})}$. Third, $\mathcal{H}$ is such that for every (x,y) that belongs to `diff` in $\mathcal{H}$, $(x,y) \in Diff^{\mathcal{O}}$, where $x,y \in V_N^{\mathcal{O}}$. It can be shown that, by virtue of the above properties, since a tuple $\boldsymbol{t}$ is an answer to a query Q over $\mathcal{O}$ under DSER if and only if $\boldsymbol{t}$ is an answer to Q over the restriction of $Can(\mathcal{O})$ to the elements of $V_N^{\mathcal{O}}$, and since $\boldsymbol{t}$ is an answer to $\mathcal{P}^Q$ if and only if `Ans(`$\boldsymbol{t}$`)` belongs to $\mathcal{H}$, then $\boldsymbol{t}$ is an answer to a query Q over $\mathcal{O}$ under DSER if and only if `Ans(`$\boldsymbol{t}$`)` belongs to $\mathcal{H}$. $\square$

It is worth noting that the *AnsByDat* algorithm works for all queries that are legal under DSER. Hence, in particular, it allows to answer queries comprising `DifferentIndividuals`.

As for complexity, we remark that the algorithm above is obviously PTIME in both data and ontology complexity, since the evaluation of a Datalog program is PTIME in data complexity.

4 The Direct Semantics Answering Regime

In this section we investigate query answering over `OWL 2 QL` ontologies, under a `SPARQL` entailment regime, called *Direct Semantics Answering Regime (DSAR)*, obtained from DSER by defining answers to conjunctive queries as in classical first-order semantics. In particular, based on previous results, we show that query answering under DSAR is undecidable, we identify a specific class of queries with `DifferentIndividuals`, for which the problem is tractable and provide an algorithm for answering queries within such class that is AC^0 in data complexity.

We first formally introduce the (`OWL 2 QL`) DSAR. Specifically, DSAR is such that (*i*) it adopts the same syntax and semantics of DSER for the queried ontology, i.e., it considers `OWL 2 QL` ontologies interpreted according to DS, (*ii*) it considers as legal the same set of queries that are legal under DSER, i.e., any query that is a conjunction of `OWL 2 QL` assertions, and (*iii*) it defines the set of answers to a conjunctive query as follows. Let $\boldsymbol{t}$ be an n-tuple of `IRI`s and q a conjunctive query over $\mathcal{O}$. $\boldsymbol{t}$ is an *answer to q over $\mathcal{O}$ under DSAR*, and we write $\boldsymbol{t} \in Ans^{DSAR}(q,\mathcal{O})$, if

$$\mathcal{O} \models \exists y\, \sigma[\boldsymbol{x} \to \boldsymbol{t}](body(q)),$$

where $\boldsymbol{x}$ are the target variables of q and $\boldsymbol{y}$ are the existential variables of q. Note that, in contrast to DSER, DSAR assigns to existentially quantified variables the classical logical meaning. Finally, *answers to unions of conjunctive queries under DSAR* are defined as under DSER, i.e. as the result of computing the union to the answers to each conjunctive query in the union.

The following lemma immediately follows from the fact `OWL 2 QL` is based on $DL\text{-}Lite_{\mathtt{R}}$ and from results of [11] showing that the problem of answering conjunctive queries with ABox atoms and inequalities in $DL\text{-}Lite_{\mathtt{R}}$ is undecidable.

Lemma 1. *Query answering in* `OWL 2 QL` *under DSAR is undecidable.*

Inspired by results of the previous section, in order to identify a class of queries for which query answering under DSAR is decidable, we introduce the notion of query *with bound inequalities*. Specifically, given a query Q we say that Q is with bound inequalities if for every variable x, if x occurs within a `DifferentIndividuals` atom of Q, then x is a target variable of Q. Intuitively, a query with bound inequalities is such that if it requires two objects to be distinct, such objects are to be distinct in every model of $\mathcal{O}$. Obviously, all queries that do not involve inequalities are queries with bound inequalities.

We now present the algorithm *AnsByRew* for answering queries over `OWL 2 QL` ontologies under DSAR. Given an ontology $\mathcal{O}$ and a query Q with bound inequalities that is a union of conjunctive queries $q_1, \ldots, q_p$, the algorithm *AnsByRew* proceeds as follows:

1. it constructs an instance $B^{\mathcal{O}}$ of the schema $\mathcal{S}^{ql}$ by applying τ to every axiom of $\mathcal{O}$, thus storing into $B^{\mathcal{O}}$ a relational representation of $\mathcal{O}$;
2. it evaluates the program $\mathcal{P}^{ql,\mathcal{T}}$ over $B^{\mathcal{O}}$;
3. it rewrites Q into a query Q' by applying the rewriting technique proposed in [5] for $DL\text{-}Lite_{\mathtt{R}}$; note that such a rewriting will leave unmodified all TBox atoms as well as all `DifferentIndividuals` atoms;
4. it translates Q' into a query Q'' over the alphabet $\mathcal{S}^{ql}$ by applying τ to each conjunctive query within Q'; note that Q'' is a first-order query over $\mathcal{S}^{ql}$;
5. it rewrites Q'' into a union of queries Q''' not containing any atom with predicate `diff`; to do so, for every conjunctive query q''_i in Q'', it constructs a union Q'''_i of all queries q'''_j such that q'''_j is obtained from q''_i by substituting each atom $\mathtt{diff}(x, y)$ with either:
   - an atom $\mathtt{instr}(r, x, y)$ for $r_1 \in \mathtt{irrif}(r)$, or
   - a pair of atoms $\mathtt{instC}(c1, x)$, $\mathtt{instC}(c2, y)$ for $(c1, c2) \in \mathtt{disjCC}$, or
   - a pair of atoms $\mathtt{instC}(c1, x)$, $\mathtt{existsR}(r2, y)$ for $(c1, r2) \in \mathtt{disjCR}$, or
   - a pair of atoms $\mathtt{instC}(c1, x)$, $\mathtt{existsI}(r2, y)$ for $(c1, r2) \in \mathtt{disjCI}$, or
   - a pair of atoms $\mathtt{existI}(r1, x)$, $\mathtt{existsI}(r2, y)$ for $(r1, r2) \in \mathtt{disjII}$.
6. it evaluates the first-order query Q''' over the database $B^{\mathcal{O}}$.

Theorem 2. *If $\mathcal{O}$ is an ontology and Q is a query with bound inequalities legal under DSER, then $Ans^{DSAR}(Q, \mathcal{O}) = AnsByRew(Q, \mathcal{O})$.*

Proof (sketch). The proof relies on two crucial properties. First, the evaluation of the program $\mathcal{P}^{\texttt{QL},\mathcal{T}}$ computes the closure of the TBox of $\mathcal{O}$. Thus, the instances of the predicates in $\mathcal{S}^{ql,\mathcal{T}}$ in $B^{\mathcal{O}}$ represent all TBox axioms logically implied by $\mathcal{O}$, and logical implication of TBox atoms can be checked by evaluating their encoding through τ on $B^{\mathcal{O}}$. Second, since Q is such that its `DifferentIndividuals` atoms involve only target variables, any tuple $\boldsymbol{t}$ that is a certain answer to Q is such that there exists $t_i, t_j \in \boldsymbol{t}$ such that $\texttt{DifferentIndividuals}(t_i, t_j)$ belongs to the minimum model of $\mathcal{P}$ that contains $B^{\mathcal{O}}$. □

Clearly, the above theorem shows that query answering under DSAR is AC^0 in data complexity, for the class of queries with bound inequalities, while it is PTime in ontology complexity.

Also, it is easy to see that query answering under DSER can be reduced to query answering under DSAR. Hence, it follows that one can use *AnsByRew* to solve query answering under DSER, which in particular shows that the complexity of the algorithm *AnsByDat* is not optimal. Still, we argue that *AnsByDat* may be useful and its usage may be sometimes convenient since it allows to exploit commercial highly optimized Datalog engines.

5 Conclusion

In this work we have investigated query answering in `OWL 2 QL` under the `SPARQL` DSER, taking into account specific aspects arising from a careful adherence to all the standards in play. In particular, we have shown that despite the set of legal queries for DSER comprise queries with inequalities, query answering under DSER can be reduced to the evaluation of a Datalog program. Furthermore, we have introduced a `SPARQL` entailment regime that defines answers to conjunctive queries as in classical first-order semantics, and have provided a query answering algorithm for answering queries under such regime, for a restricted class of queries with inequalities, that is AC^0 in data complexity.

We plan to continue our work along several directions. In particular, we plan to consider query answering for other more general classes of queries with inequalities that are legal under DSER, aiming at finding what is the boundary between decidable (tractable) and undecidable (untractable) query answering. Also, we plan to carry out an experimental comparison of the perfomance of query answering under DSER using Datalog rewriting with that of query answering under DSER using the rewriting technique proposed in Section 4.

Acknowledgements We gratefully acknowledge support by the MIUR under the SIR project MODEUS (RBSI14TQHQ), and by Regione Lazio under the project MAGISTER.

References

1. M. Alviano, W. Faber, N. Leone, S. Perri, G. Pfeifer, and G. Terracina. The disjunctive datalog system DLV. In *Datalog 2010*, pages 282–301, 2010.
2. M. Aref, B. ten Cate, T. J. Green, B. Kimelfeld, D. Olteanu, E. Pasalic, T. L. Veldhuizen, and G. Washburn. Design and implementation of the logicblox system. In *Proc. of ACM SIGMOD*, pages 1371–1382, 2015.
3. M. Arenas, G. Gottlob, and A. Pieris. Expressive languages for querying the semantic web. In *Proc. of PODS 2014*, pages 14–26, 2014.
4. A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The dl-lite family and relations. *CoRR*, abs/1401.3487, 2014.
5. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodriguez-Muro, R. Rosati, M. Ruzzi, and D. F. Savo. The Mastro system for ontology-based data access. *Semantic Web J.*, 2(1):43–53, 2011.
6. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *J. of Automated Reasoning*, 39(3):385–429, 2007.
7. B. Glimm. Using SPARQL with RDFS and OWL entailment. In *RW-11*, pages 137–201. 2011.
8. B. Glimm, I. Horrocks, C. Lutz, and U. Sattler. Conjunctive query answering for the description logic SHIQ. *CoRR*, abs/1111.0049, 2011.
9. G. Gottlob and A. Pieris. Beyond SPARQL under OWL 2 QL entailment regime: Rules to the rescue. In *Proc. of IJCAI 2015*, pages 2999–3007, 2015.
10. B. N. Grosof, I. Horrocks, R. Volz, and S. Decker. Description logic programs: Combining logic programs with description logic. In *Proc. of WWW 2003*, pages 48–57, 2003.
11. V. Gutiérrez-Basulto, Y. A. Ibáñez-García, R. Kontchakov, and E. V. Kostylev. Queries with negation and inequalities over lightweight ontologies. *J. of Web Semantics*, 35:184–202, 2015.
12. S. Kikot, R. Kontchakov, and M. Zakharyaschev. Conjunctive query answering with OWL 2 QL. In *Proc. of KR 2012*, 2012.
13. R. Kontchakov, M. Rezk, M. Rodriguez-Muro, G. Xiao, and M. Zakharyaschev. Answering SPARQL queries over databases under OWL 2 QL entailment regime. In *Proc. of ISWC 2014*, pages 552–567, 2014.
14. C. Lutz, D. Toman, and F. Wolter. Conjunctive query answering in the description logic EL using a relational database system. In *Proc. of IJCAI 2009*, pages 2070–2075, 2009.
15. R. Volz. *Web ontology reasoning with logic databases.* PhD thesis, Universität Karlsruhe, 2004.