Exact Query Reformulation over $\mathcal{SHOQ}$ DBoxes

Enrico Franconi, Volha Kerhet, Nhung Ngo

Free University of Bozen-Bolzano, Italy
lastname@inf.unibz.it

Abstract We formalise the problem of query reformulation over a description logic ontology and a DBox in a general framework. This framework supports deciding the existence of a safe-range first-order equivalent reformulation of a concept query in terms of the signature of a DBox. A constructive method to compute the reformulation is provided. We are particularly interested in safe-range reformulations since they can be transformed to relational queries and executed using SQL. We also discuss the completeness of the proposed framework with respect to finite and unrestricted models. As a case study we consider ontologies and queries expressed in $\mathcal{SHOQ}$.

1 Introduction

In this paper we study and develop a query rewriting framework which is applicable to description logics systems where data is stored in a classical finite relational database, in a way that in the literature has been called *DBox* [5,6]. A DBox is a set of ground atoms which semantically behaves like a database, i.e., the interpretation of the database predicates in the DBox is exactly equal to the database relations. The DBox predicates are *closed*, i.e., their extensions are the same in every interpretation, whereas the other predicates in the knowledge base are *open*, i.e., their extensions may vary among different interpretations. We do not consider here the *open* interpretation for the database predicates – i.e., the classical *ABox*. In an ABox, the interpretation of database predicates contains the database relations and possibly more. This notion clearly is less faithful in the representation of a database semantics since it would allow for spurious interpretations of database predicates with additional unwanted tuples not present in the original database.

In our general framework an ontology is a TBox in a first-order description logic, and queries are concept expressions. Within this setting, the framework provides support to decide the existence of a relational algebra (i.e., safe-range first-order) *equivalent* (a.k.a. *exact*) reformulation of the query in terms of the DBox signature. It also provides an effective approach to construct the reformulation. We are particularly interested in safe-range reformulations of queries because their range-restricted syntax is needed to reduce the original query answering problem to a relational algebra evaluation (e.g., via SQL) over the original database [7]. Our framework points out several conditions on the ontology and the query in order to guarantee the existence of a safe-range equivalent reformulation. We show that these conditions are not infeasible in practice and we also provide an efficient method to ensure their validation. Standard tableau techniques can be used to compute the reformulation.

In order to be complete, our framework is applicable to ontologies and queries expressed in any fragment of first-order logic enjoying the finitely controllable determinacy [3,8]. If the employed logic does not enjoy the finitely controllable determinacy our approach would become sound but incomplete, by still effectively implementable using standard theorem proving techniques. We have explored non-trivial applications where the framework is complete; in this paper, the application with $\mathcal{SHOQ}$ ontology and concept queries is discussed. We show how (i) to check whether the answers to a given query with an ontology are *solely* determined by the extension of the DBox predicates and, if so, (ii) to find an equivalent rewriting of the query in terms of the DBox predicates to allow the use of standard database technology for answering the query. This means we benefit from the low computational complexity in the size of the data for answering queries on relational databases. In addition, it is possible to reuse standard techniques of description logics reasoning to find rewritings, such as in [5].

The query reformulation problem has received strong interest in classical relational database research as well as modern knowledge representation studies. Differently from the mainstream research on query reformulation [9], which is mostly based on perfect or maximally contained rewritings with sound views (see, e.g., the DL-Lite approach [10]), we focus here on exact rewritings with exact views, since it characterises more precisely the query answering problem with ontologies and databases, and it allows for very expressive ontology languages. An exact reformulation has the same answer in any model of the ontology with the DBox, and it provides a fully determined answer, which may be useful, e.g., for materialisation.

This work extends the seminal works on exact rewritings with exact views [2,5,3] by focussing on safe-range reformulations and on the conditions ensuring their existence in description logics. This is necessary when the description logic at hand is not enjoying the Beth definability property [11], which would guarantee the rewriting to be safe-range. The detailed algorithms and all the proofs for a more general framework are available in the technical report [8].

The paper is organised as follows. Section 2 provides the necessary formal background and definitions. Section 3 introduces a characterisation of the query reformulation problem, and the conditions allowing for an effective reformulation are analysed. At the end, we discuss in details the application to $\mathcal{SHOQ}$ ontologies with a DBox.

2 Preliminaries

In this section we define the basic concepts that are used in the paper.

2.1 Description Logics and DBox

Let N_C, N_R and N_I be sets of concept, role and individual names respectively. And let $\mathcal{L}(N_C, N_R, N_I)$ be some description logic language over N_C, N_R and N_I. An *ontology* is a set of TBox assertions in $\mathcal{L}(N_C, N_R, N_I)$.

Let C be a (possibly complex) concept or an assertion in $\mathcal{L}(N_C, N_R, N_I)$. We denote as $\sigma(C)$ the signature of C, that is the union of all concept, role and individual names occurring in C.

A DBox $\mathcal{D}$ is a *finite* set of atomic concept and role assertions of the form $A(a)$ and $R(a, b)$ respectively, where $A \in N_C$, $R \in N_R$ and $a, b \in N_I$. The sets of all concept,

role and individual names appearing in $\mathcal{D}$ are denoted as $\sigma_{\mathcal{D}}(C)$, $\sigma_{\mathcal{D}}(R)$ and $\sigma_{\mathcal{D}}(I)$ respectively. We call *DBox predicates* the set $\sigma_{\mathcal{D}}(P) = \sigma_{\mathcal{D}}(C) \cup \sigma_{\mathcal{D}}(R)$.

As usual, an *interpretation* $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ includes a domain $\Delta^{\mathcal{I}}$ and an interpretation function $\cdot^{\mathcal{I}}$ that maps concepts to subsets of $\Delta^{\mathcal{I}}$, roles to binary relations on $\Delta^{\mathcal{I}}$ and individuals to elements of $\Delta^{\mathcal{I}}$.

We say that an interpretation $\mathcal{I}$ *embeds a DBox* $\mathcal{D}$, written $\mathcal{I}_{(\mathcal{D})}$, if it holds that: (i) $a^{\mathcal{I}} = a$ for every DBox individual $a \in \sigma_{\mathcal{D}}(I)$, i.e. a follows the *standard name assumption (SNA)* [7]; (ii) for every concept name A in $\sigma_{\mathcal{D}}(C)$ and every $u \in \Delta^{\mathcal{I}}$, $u \in A^{\mathcal{I}}$ if and only if $A(u) \in \mathcal{D}$; and (iii) for every role name R in $\sigma_{\mathcal{D}}(R)$ and every pair $(u, v) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, $(u, v) \in R^{\mathcal{I}}$ if and only if $R(u, v) \in \mathcal{D}$. In other words, in every interpretation embedding $\mathcal{D}$, the interpretation of any DBox predicate is always the same and it is given exactly by its content in the DBox; this is, in general, not the case for the interpretation of the non-DBox predicates. Under above embedding conditions, we say that all the DBox predicates are *closed*, while all the other predicates are *open* and may be interpreted differently in different interpretations.

In order to allow for an arbitrary DBox to be embedded, we generalise the standard name assumption to all the individual names in N_I; this implies that the domain of any interpretation necessarily includes the set of all the individual names N_I.

We denote an interpretation $\mathcal{I}$ with a specific domain Δ as $\mathcal{I}^{(\Delta)}$. Given an interpretation $\mathcal{I}$, we denote as $\mathcal{I}|_{\mathbb{S}}$ the interpretation restricted to the smaller signature $\mathbb{S} \subseteq N_C \cup N_R \cup N_I$, i.e., the interpretation with the same domain $\Delta^{\mathcal{I}}$ and the same interpretation function $\cdot^{\mathcal{I}}$ defined only for the concept, role and individual names from the set $\mathbb{S}$.

We call $FOL(\mathbb{C}, \mathbb{P})$ a function free first-order language with equality over a signature $\Sigma = (\mathbb{C}, \mathbb{P})$, where $\mathbb{C} = N_I$ is a set of constants and $\mathbb{P} = N_C \cup N_R$ is a set of predicates with arities 1 (for concept names) and 2 (for role names).

An interpretation in which an assertion φ (TBox or ABox) is true is called a *model* of φ; the set of all models of φ is denoted as $M(\varphi)$. The set of all models of all assertions in an ontology $\mathcal{T}$ is denoted as $M(\mathcal{T})$. We say that a *DBox $\mathcal{D}$ is legal for an ontology $\mathcal{T}$* if there exists a model of $\mathcal{T}$ embedding $\mathcal{D}$. In the paper, we consider only consistent non-tautological ontologies and legal DBoxes.

2.2 Queries and certain answers

A *query* is a concept in $\mathcal{L}(N_C, N_R, N_I)$. Given a query Q, we define its *certain answer* to a DBox $\mathcal{D}$ under $\mathcal{T}$ as follows:

Definition 1 (Certain Answer) *The* (certain) answer *of a query Q to a DBox $\mathcal{D}$ under an ontology $\mathcal{T}$ is the set of individuals:*

$$\{a \in N_I \mid \forall \mathcal{I}_{(\mathcal{D})} \in M(\mathcal{T}) \,:\, \mathcal{I}_{(\mathcal{D})} \models Q(a)\}.$$

We now show that we can weaken the standard name assumption for the constants by just assuming *unique names*, without changing the certain answers. As we said before, an interpretation $\mathcal{I}$ embedding a DBox $\mathcal{D}$ satisfies the standard name assumption – written $\mathcal{I}_{(\mathcal{D})^{SNA}}$ – if $c^{\mathcal{I}} = c$ for any $c \in N_I$. Alternatively, an interpretation $\mathcal{I}$ embedding a DBox $\mathcal{D}$ satisfies the unique name assumption – written $\mathcal{I}_{(\mathcal{D})^{UNA}}$ – if $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ for any different $a, b \in N_I$. The following proposition allows us to freely interchange the

standard name and the unique name assumptions in dealing with interpretations embedding DBoxes. This is a practical advantage, since most description logics reasoners do have a native unique name assumption.

Proposition 1 (SNA vs UNA) *For any query $Q(x)$, ontology $\mathcal{T}$ and DBox $\mathcal{D}$,*

$\{a \in N_I \,|\, \forall \mathcal{I}_{(\mathcal{D})^{SNA}} \in M(\mathcal{T}) : \mathcal{I}_{(\mathcal{D})^{SNA}} \models Q(a)\} =$
$\{a \in N_I \,|\, \forall \mathcal{I}_{(\mathcal{D})^{UNA}} \in M(\mathcal{T}) : \mathcal{I}_{(\mathcal{D})^{UNA}} \models Q(a)\}.$

A query is *DBox-relativised* if and only if its answer is bounded by the DBox.

Definition 2 (DBox-relativised query) *A concept query Q is* DBox-relativised under ontology $\mathcal{T}$*, if in each model of $\mathcal{T}$ the interpretation of Q includes only domain elements which are among the interpretation of DBox predicates or of individuals from $\mathcal{T}$ or Q.*

2.3 Safe-range formulas

Since a query can be an arbitrary first-order formula, its answer can be infinite (since the domain is not restricted to be finite) or it may depend on the domain. To eliminate such cases, we will consider *domain independent* queries. For example, the query $Q = \neg Student$ over the DBox $Student(A), Student(B)$, with domain $\{A, B, C\}$ has the answer $\{x = C\}$, with domain $\{A, B, C, D\}$ has the answer $\{x = C, x = D\}$, and if we change the domain to an infinite one, the answer will be infinite even in presence of such a finite database. Therefore, the notion of *domain independent* queries has been introduced in relational databases.

In general, the problem of checking whether a FOL formula is domain independent is undecidable [7]. The well known *safe-range* syntactic fragment of FOL introduced by Codd is an *equally expressive* language; indeed any safe-range formula is domain independent, and any domain independent formula can be easily transformed into a logically equivalent safe-range formula. Intuitively, a formula is safe-range if and only if its variables are bounded by positive predicates or equalities – for the exact syntactical definition see, e.g., [7]. For example, the formula $\neg A(x) \wedge B(x)$ is safe-range, while queries $\neg A(x)$ and $\forall x.\, A(x)$ are not. To check whether a formula is safe-range, the formula is transformed into a logically equivalent *safe-range normal form* and its *range restriction* is computed according to a set of syntax based rules; the range restriction of a formula is a subset of its free variables, and if it coincides with the free variables then the formula is said to be safe-range.

Any formula in $FOL(\mathbb{C}, \mathbb{P})$ can be transformed to a logically equivalent *safe range normal form* (SRNF) by recursively applying the following steps :

- Variable substitution: no distinct pair of quantifiers may employ same variable.
- Elimination of universal quantifiers
- Elimination of implications
- Pushing negation
- Flattening of and/or

A formula is said to be SRNF if none of the aforementioned steps can be applied any more. Let φ be a formula in $FOL(\mathbb{C}, \mathbb{P})$, we denote the set of all variables appearing in φ as $\text{VAR}(\varphi)$, and the set of the free variables appearing in φ as $\text{FREE}(\varphi)$. The safe

range normal form of φ is denoted as $\text{SRNF}(\varphi)$. Let φ be a formula in SRNF. The *range restriction of* φ, denoted as $rr(\varphi)$, is either a subset of $\text{FREE}(\varphi)$ or $\bot$, and it is computed according to the following rules:

- $rr(R(t_1, \ldots, t_n)) = \text{VAR}(R(t_1, \ldots, t_n))$;
- $rr(x = y) = \emptyset$;
- $rr(x = c) = \{x\}$, where $c \in \mathbb{C}$;
- $rr(\varphi_1 \wedge \varphi_2) = rr(\varphi_1) \cup rr(\varphi_2)$;
- $rr(\varphi_1 \vee \varphi_2) = rr(\varphi_1) \cap rr(\varphi_2)$;
- $rr(\varphi \wedge x = y) = rr(\varphi)$, if $\{x, y\} \cap rr(\varphi) = \emptyset$; and $rr(\varphi \wedge x = y) = rr(\varphi) \cup \{x, y\}$ otherwise;
- $rr(\neg\varphi) = \emptyset \cap rr(\varphi)$;
- $rr(\exists x\varphi) = rr(\varphi) \setminus x$ if $x \in rr(\varphi)$ and $rr(\exists x\varphi) = \bot$ otherwise.

We consider $\bot$ as a special set, such that for any set Z : $\bot \cup Z = \bot \cap Z = \bot \setminus Z = Z \setminus \bot = \bot$. If φ is not in SRNF, then $rr(\varphi) := rr(\text{SRNF}(\varphi))$. We say that a *variable* $x \in \text{FREE}(\varphi)$ *has restricted range in* φ if $x \in rr(\varphi)$.

Definition 3 (Safe range) *A formula φ is safe range iff* $rr(\text{SRNF}(\varphi)) = \text{FREE}(\varphi)$.

We also consider a weaker version of safe-range property called ground safe-range. Given a formula, its *grounding* is the formula itself where all free variables are replaced by new constants.

Definition 4 (Ground safe-range) *A formula is ground safe-range if its grounding is safe-range.*

The safe-range fragment of first-order logic with the standard name assumption is equally expressive to the relational algebra, which is the core of the SQL query language [7].

For any concept C in $\mathcal{L}(N_C, N_R, N_I)$ we denote the corresponding logically equivalent formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ with one free variable x as $C(x)$. We will call any axiom (concept) in $\mathcal{L}(N_C, N_R, N_I)$ (ground) safe-range, if the corresponding logically equivalent formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ is (ground) safe-range. An ontology $\mathcal{T}$ in $\mathcal{L}(N_C, N_R, N_I)$ is safe-range, if every formula in $\mathcal{T}$ is safe-range.

3 Exact Safe-range Query Reformulation

In this section we state the problem of finding a first-order safe-range reformulation of a concept query. We then find the conditions to reduce the original query answering problem – which corresponds to an entailment problem – to a model checking problem of the reformulation over the DBox.

Let us consider the class of queries of interest. The certain answer to a query includes all the individuals which make the query true in *all* the models of the ontology: so, if an individual would make the query true only in some model, then it would be discarded from the certain answer. In other words, it may be the case that the answer to the query is not necessarily the same among all the models of the knowledge base. In this case, the query is not fully determined by the given source data; indeed, there

is some answer which is possible, but not certain. Due to the indeterminacy of the data wrt the query, the complexity to compute the certain answer in general increases, and it corresponds to the complexity of entailment in the logic. In this paper we focus on the case when a query has the same answer over all the models of the ontology, namely, on the case when the information requested by the query is fully available from the source data without ambiguity. In this way, the indeterminacy disappears, and the complexity of the process may decrease.

A query is *definable* [12] if its truth value in any model of the ontology depends *only* on the domain and on the interpretation of the database predicates and constants. The answer of a definable query does not depend on the interpretation of non-database predicates. Once the database and a domain are fixed, it is never the case that an individual would make the query true in some model of the knowledge base and false in others, since the truth value of an implicitly defined query depends only on the interpretation of the database predicates and constants and on the domain (which are fixed).

[12] proved that, in first-order logic, looking for definable queries from the DBox predicates amounts at having an *exact reformulation* of the query in terms of the DBox predicates.

Definition 5 (Exact Reformulation) *The $\mathcal{FOL}(\mathbb{C},\mathbb{P})$ formula $\widehat{Q}$ is an exact reformulation of Q under $\mathcal{T}$ over $\sigma_{\mathcal{D}}(P)$ if $\sigma(\widehat{Q}) \subseteq \sigma_{\mathcal{D}}(P)$ and $\mathcal{T} \models \forall x.Q(x) \leftrightarrow \widehat{Q}(x)$.*

Since we are dealing with finite databases, in the following we will focus on those fragments of $\mathcal{FOL}(\mathbb{C},\mathbb{P})$ for which the exact reformulation over unrestricted models and over finite models coincide; we say that these fragments have *finitely controllable determinacy*.

Given DBox predicates $\sigma_{\mathcal{D}}(P)$, an ontology $\mathcal{T}$, and a query Q in the $\mathcal{L}(N_C, N_R, N_I)$ language, our goal is to find a safe-range exact reformulation $\widehat{Q}$ of Q in $\mathcal{FOL}(\mathbb{C},\mathbb{P})$ expressed in terms of DBox predicates, that being evaluated as a relational algebra expression over a legal DBox (e.g., using a relational database system with SQL) gives the same answer as the certain answer of Q to the DBox under $\mathcal{T}$.

Since an exact reformulation is equivalent under the ontology to the original query, the certain answer of the original query and of the reformulated query are identical. More precisely, the following proposition holds.

Proposition 2 *Let $\widehat{Q}$ be an exact reformulation of Q under $\mathcal{T}$ over $\sigma_{\mathcal{D}}(P)$, then:*

$$\{a \in N_I \,|\, \forall \mathcal{I}_{(\mathcal{D})} \in M(\mathcal{T}) : \mathcal{I}_{(\mathcal{D})} \models Q(a)\} =$$
$$\{a \in N_I \,|\, \forall \mathcal{I}_{(\mathcal{D})} \in M(\mathcal{T}) : \mathcal{I}_{(\mathcal{D})} \models \widehat{Q}(a)\}.$$

From the above equation it is clear that in order to answer an exactly reformulated query, one still may need to consider all the models $\mathcal{I}_{(\mathcal{D})}$ of the ontology embedding the DBox – i.e., we still have an entailment problem to solve. The following theorem states the condition to reduce the original query answering problem – based on entailment – to the problem of checking the validity of the exact reformulation over a *single* model: the condition is that the reformulation should be safe-range.

Theorem 1 (Adequacy of Exact safe-range Query Reformulation) *Let $\mathcal{T}$ be an ontology in $\mathcal{L}(N_C, N_R, N_I)$, Q be a query in $\mathcal{L}(N_C, N_R, N_I)$ and $\mathcal{D}$ be a legal DBox for $\mathcal{T}$. If $\widehat{Q}$ is an exact reformulation of Q under $\mathcal{T}$ over $\sigma_{\mathcal{D}}(P)$ and $\widehat{Q}$ is safe-range, then:*

$\{a \in N_I \mid \forall \mathcal{I}_{(\mathcal{D})} \in M(\mathcal{T}) : \mathcal{I}_{(\mathcal{D})} \models Q(a)\} =$
$\{a \in adom(\sigma(\widehat{Q}), \mathcal{D}) \mid \mathcal{I}_{(\mathcal{D})}^{(N_I)}|_{\sigma_{\mathcal{D}}(P) \cup N_I} \models \widehat{Q}(a)\}$,
where $adom(\sigma(\widehat{Q}), \mathcal{D})$ *consists of all the constants from* $\widehat{Q}$ *and from the assertions in* $\mathcal{D}$ *corresponding to concept and role names appearing in* $\widehat{Q}$.

A safe-range reformulation is *necessary* to transform a first-order query to a relational algebra query which can then be evaluated by using SQL techniques. The theorem above shows in addition that being safe-range is also a *sufficient* property for an exact reformulation to be correctly evaluated as an SQL query.

However, given an arbitrary input (an ontology, a DBox and a concept query), one can not guarantee the existence of an exact safe-range reformulation. Therefore, in the rest of this section we introduce conditions on the input to get an exact safe-range reformulation. Moreover, since we are dealing with a finite DBox, we have also to consider the condition under which the existence of an exact safe-range reformulation under unrestricted reasoning and under finite reasoning coincide.

Let Q be any formula and $\widetilde{Q}$ the formula obtained from it by uniformly replacing every occurrence of each non-DBox predicate P with a new predicate $\widetilde{P}$. We extend this renaming operator $\widetilde{\cdot}$ to any set of formulas in a natural way. Then the following *constructive theorem* gives us sufficient conditions to check the existence of an exact safe-range reformulation.

Theorem 2 (Constructive Theorem) *If the following conditions hold:*

1. $\mathcal{T} \cup \widetilde{\mathcal{T}} \models Q \equiv \widetilde{Q}$;
2. Q *is DBox-relativised under* $\mathcal{T}$;
3. Q *is ground safe-range;*
4. $\mathcal{T}$ *is safe-range;*

then there exists an exact safe-range reformulation $\widehat{Q}$ *of* Q *in* $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ *over* $\sigma_{\mathcal{D}}(P)$ *under* $\mathcal{T}$.

The above conditions can be divided into two groups: the first condition forces the existence of an exact reformulation, while the three last conditions guarantee its safe-range property. The first condition says that one does not need to consider non-DBox predicates to answer the query. In other words, its answer in any model of the ontology depends *only* on the domain and on the interpretation of the DBox predicates and constants. This property of a query is called *implicit definability* from a set of predicates (the DBox predicates) in first-order logic [12]. The second condition points out that the answer of the query is necessarily in the set of individuals appearing in the DBox original query or ontology.

The first two conditions are necessary to have an exact safe-range reformulation, i.e. if there is an exact safe-range reformulation, then the original query should be implicitly definable and DBox-relativised. The last two conditions are just sufficient ones, as the following example shows.

Example 1 *Let* $\mathbb{P} = \{A, B, C\}$, $\sigma_{\mathcal{D}}(I) = \{C\}$, $\mathcal{T} = \{A \equiv C, \top \sqsubseteq B\}$, $Q = A \sqcap B$.

– Q *is implicitly definable from the DBox predicates under* $\mathcal{T}$ *because the first assertion of* $\mathcal{T}$ *gives an explicit definition of* Q;

– *Q is safe-range;*
– *Q is DBox-relativised under $\mathcal{T}$ because of the first assertion of $\mathcal{T}$.*
– *$\widehat{Q}(x) = C$ is an exact safe-range reformulation of Q under $\mathcal{T}$ over $\sigma_{\mathcal{D}}(I)$.*

But $\mathcal{T}$ is not safe-range because of the second assertion. □

4 A case study: $\mathcal{SHOQ}$

Syntax	Semantics
A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$\neg C$	$\Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$
$\exists R.C$	$\{x \mid$exists y such that $(x,y) \in R^{\mathcal{I}}$ and $y \in C^{\mathcal{I}}\}$
$\forall R.C$	$\{x \mid$forall y $(x,y) \in R^{\mathcal{I}}$ implies $y \in C^{\mathcal{I}}\}$
$\{o\}$	$\{o\}^I \subseteq \Delta^I$
$\geq nR.C$	$\{x \mid \#(\{y \mid (x,y) \in R^{\mathcal{I}}\} \cap C^{\mathcal{I}}) \geq n\}$
$\leq nR.C$	$\{x \mid \#(\{y \mid (x,y) \in R^{\mathcal{I}}\} \cap C^{\mathcal{I}}) \leq n\}$

Table 1. Syntax and semantics of $\mathcal{SHOQ}$ concepts

$\mathcal{SHOQ}$ is an extension of the description logic $\mathcal{ALC}$ with transitive roles, role hierarchies, qualified number restrictions, and individuals; it is a fragment of first-order logic and of OWL2. The syntax and semantics of $\mathcal{SHOQ}$ is summarised in table 1, where A is an atomic concept, C and D are concepts, o is an individual name and R is an atomic role. $\mathcal{SHOQ}$ is a pretty much standard description logic; for more details see, e.g., [13]. A TBox in $\mathcal{SHOQ}$ is a set of concept inclusion axioms $C \sqsubseteq D$, role inclusion axioms $R \sqsubseteq S$, and transitivity axioms $Trans(R)$ (where C, D are concepts and R, S are atomic roles) with the usual expected semantics.

In this section, we present an application of our framework where the ontology is a TBox in $\mathcal{SHOQ}$, and the query is a $\mathcal{SHOQ}$ concept.

Finitely controllable determinacy. Does $\mathcal{SHOQ}$ have finite controllability of determinacy? It is enough to check that the entailment $\mathcal{T} \cup \widetilde{\mathcal{T}} \models Q \equiv \widetilde{Q}$ coincide in the unrestricted and finite cases. The finite controllability of this equivalence axiom entailment in $\mathcal{SHOQ}$ is guaranteed because of the two following reasons:

– The entailment $\mathcal{T} \cup \widetilde{\mathcal{T}} \models Q \equiv \widetilde{Q}$ can be reduced in $\mathcal{SHOQ}$ to a concept satisfiability problem for an empty TBox.
– $\mathcal{SHOQ}$ has finite model property [14].

So, we can use a standard $\mathcal{SHOQ}$ reasoner (e.g., an OWL2 reasoner) to check the first condition.

Safe-range ontology. Let's now check whether a $\mathcal{SHOQ}$ ontology is safe-range. Role inclusion and transitivity axioms are always safe-range. Unfortunately, concept inclusion axioms in $\mathcal{SHOQ}$ ontologies may not be safe-range: for example, the axiom $\neg\ \texttt{male} \sqsubseteq \texttt{female}$ is not safe-range. It is easy to see that an axiom $C \sqsubseteq D$ is not safe-range if and only if $C(x)$ is not safe-range and $D(x)$ is safe-range: just observe that the axiom is logically equivalent to the formula $\neg\exists x.\, C(x) \wedge \neg D(x)$ in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$. The following proposition provides rules deciding whether a $\mathcal{SHOQ}$ concept is safe-range.

Proposition 3 *Let A be an atomic concept, C and D be $\mathcal{SHOQ}$ concepts. Then the open formulas:*

1. *$A(x), (\exists R.C)(x), \{o\}(x), (\geq nR.C)(x)$ are safe-range;*
2. *$(\forall R.C)(x), (\leq nR.C)(x)$ are not safe-range;*
3. *$(C \sqcap D)(x)$ is safe-range if and only if $C(x)$ is safe-range or $D(x)$ is safe-range;*
4. *$(C \sqcup D)(x)$ is safe-range if and only if $C(x)$ is safe-range and $D(x)$ is safe-range;*
5. *$\neg C(x)$ is safe-range if and only if $C(x)$ is not safe-range.*

Proposition 4 *For any $\mathcal{SHOQ}$ concept C, $C(x)$ is ground safe-range.*

The presence of non-safe-range axioms in an ontology would prevent the application of our framework, but we argue that non-safe-range axioms should not appear in a cleanly designed $\mathcal{SHOQ}$ ontology, and, if present, they should be fixed. Indeed, the use of *absolute* negative information in the subsumee – such as in the axiom "a non-`male` is a `female`" ($\neg\ \texttt{male} \sqsubseteq \texttt{female}$) – should be deprecated by a clean design methodology, since the subsumer would include *all sorts* of objects in the universe (but the ones of the subsumee type) without any obvious control. Only *relativised* negative information in the subsumee should be allowed – such as in the axiom "a non-`male` `person` is a `female`" ($\texttt{person} \sqcap \neg\ \texttt{male} \sqsubseteq \texttt{female}$). This observations suggests a fix for non-safe-range axioms: for every non-safe-range axiom $C \sqsubseteq D$ users will be asked to replace it by the safe-range one $C \sqcap E \sqsubseteq D$, where E is an arbitrary safe-range concept. Therefore, the user is asked to make explicit the *type* of the subsumee, in a way to make it safe-range; note that the type could be also a fresh new atomic concept. We believe that the fix we are proposing for $\mathcal{SHOQ}$ is a reasonable one, and would make all $\mathcal{SHOQ}$ ontologies eligible to be used with our framework.

Ground safe-range and DBox-relativised query. Let $\mathcal{T}$ be a $\mathcal{SHOQ}$ ontology, and Q an implicitly definable query, which is a possibly complex concept in $\mathcal{SHOQ}$. In order to be able to use our framework, a query should be ground safe-range and *DBox*-relativised under the ontology. We already know by proposition 4 that a concept query is always ground safe-range. A query is DBox-relativised if it returns only DBox individuals; it may be strange for a user to issue a query which is not meant to return just DBox objects. One can check if Q is DBox-relativised under the ontology by using the following proposition.

Proposition 5 *The query Q is DBox-relativised under $\mathcal{T}$ if and only if:*

$$\mathcal{T} \models_{\mathcal{SHOIQ}} Q \sqsubseteq \bigsqcup_{i=1}^{k} \{o_i\} \sqcup \bigsqcup_{i=1}^{n} A_i \sqcup \bigsqcup_{i=1}^{m} (\exists R_i.\top \sqcup \exists R_i^{-}.\top), \tag{1}$$

where $\{A_1, \ldots, A_n\}$ *is the set of all DBox concepts appearing in* $\mathcal{T}$ *and* $\mathcal{Q}$*;*
$\{R_1, \ldots, R_m\}$ *is the set of all DBox roles appearing in* $\mathcal{T}$ *and* $\mathcal{Q}$*; and*
$\{o_1, \ldots, o_k\}$ *is the set of all individual names appearing in* $\mathcal{T}$ *and* $\mathcal{Q}$*.*

In other words, if the $\mathcal{SHOIQ}$ entailment in the proposition is valid, then the query is *DBox*-relativised under the ontology. We use $\mathcal{SHOIQ}$ instead of $\mathcal{SHOQ}$ because we need inverse roles. Due to the incompleteness wrt finite model reasoning of the $\mathcal{SHOIQ}$ test, one might conclude that a query is not *DBox*-relativised but in fact it is *DBox*-relativised under finite model reasoning. In the rare case a user is issuing a real non-DBox-relativised query, or a DBox-relativised query which failed the above test due to its incompleteness, we would ask the user to conjoin the query with a safe-range concept composed only by database atomic concepts, which would become the *type* of the query. We believe that also this fix for the queries is a reasonable one, and would make all queries eligible to be used with our framework.

A complete procedure. Given a $\mathcal{SHOQ}$ ontology $\mathcal{T}$, a legal DBox $\mathcal{D}$ and a concept query $\mathcal{Q}$, one can apply the procedure below to generate a safe-range exact reformulation over the DBox predicates.

Input: A $\mathcal{SHOQ}$ TBox $\mathcal{T}$, a concept query Q in $\mathcal{SHOQ}$ and a DBox predicates (DBox atomic concepts and roles).

1. Check implicit definability of the query Q by testing $\mathcal{T} \cup \widetilde{\mathcal{T}} \models Q \equiv \widetilde{Q}$ using standard DL reasoner of $\mathcal{SHOQ}$. If it is the case, continue.
2. Check whether $\mathcal{T}$ is safe-range, and fix it if it is not safe-range.
3. Check the DBox-relativisation of Q, and fix it if it is not DBox-relativised.
4. Use the constructive theorem to
   (a) compute a ground safe-range reformulation $Q'(x)$ from the tableau proof generated in step 1 (this is an extension of what has been presented in [5,11]; see [8] for a complete characterisation);
   (b) transform it to a safe-range one as follows: $\widehat{Q}(x) := Q'(x) \wedge ADOM(x)$,where $ADOM$ is a predicate containing all the individuals in the DBox, in $\mathcal{T}$, and in Q. $ADOM$ actually represents the subsumer of the TBox axiom in (1).

Output: A safe-range first-order exact reformulation $\widehat{Q}(x)$ expressed over the DBox predicates.

Note that the above procedure could be executed once for all at compile time: indeed, it could be run for each atomic concept in the ontology, and the outcome for each of them could be stored persistently, if the reformulation has been successful.

5 Conclusion

We have introduced a framework to compute the exact reformulation of concept queries to a DBox in description logics. We have found the conditions which guarantee that a safe-range reformulation exists, and we show that it can be evaluated as a relational algebra query over the database to give the same answer as the original query under the ontology. A non-trivial case study has been presented in the field of description logics, with the $\mathcal{SHOQ}$ language.

As a future work, we would like to study optimisations of reformulations. From the practical perspective, since there might be many rewritten queries from one original query, the problem of selecting an optimised one in terms of query evaluation is very important. In fact, one has to take into account which criteria should be used to optimise, such as: the size of the rewritings, the numbers of used predicates, the priority of predicates, the number of relational operators, and clever usage of duplicates.

We wish to thank David Toman, İnanç Seylan, Jos de Bruijn, Alex Borgida, Grant Weddell, Tommaso Di Noia, Umberto Straccia, Balder ten Cate with whom we have learnt a lot about query rewriting based on Beth definability, and anonymous reviewers for insightful comments.

References

1. Etzioni, O., Golden, K., Weld, D.S.: Sound and efficient closed-world reasoning for planning. Artif. Intell. **89** (January 1997) 113–148
2. Marx, M.: Queries determined by views: pack your views. In: Proceedings of the 26th ACM symposium on Principles of Database Systems. PODS '07 (2007) 23–30
3. Nash, A., Segoufin, L., Vianu, V.: Views and queries: Determinacy and rewriting. ACM Trans. Database Syst. **35** (July 2010) 21:1–21:41
4. Fan, W., Geerts, F., Zheng, L.: View determinacy for preserving selected information in data transformations. Inf. Syst. **37** (March 2012) 1–12
5. İnanç Seylan, Franconi, E., de Bruijn, J.: Effective query rewriting with ontologies over DBoxes. In: Proc. of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009). (2009) 923–925
6. Franconi, E., Ibanez-Garcia, Y.A., İnanc Seylan: Query answering with DBoxes is hard. Electronic Notes in Theoretical Computer Science, Elsevier **278** (November 2011) 71–84
7. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases. Addison-Wesley (1995)
8. Franconi, E., Kerhet, V., Ngo, N.: Exact query reformulation with expressive ontologies and databases. Technical Report 12158, KRDB Tech research group, Free University of Bozen-Bolzano (March 2012) `http://www.inf.unibz.it/krdb/pub/TR/KRDB-Tech-12158.pdf`.
9. Halevy, A.Y.: Answering queries using views: A survey. The VLDB Journal **10** (December 2001) 270–294
10. Artale, A., Calvanese, D., Kontchakov, R., Zakharyaschev, M.: The DL-Lite family and relations. J. Artif. Intell. Res. (JAIR) **36** (2009) 1–69
11. ten Cate, B., Franconi, E., İnanç Seylan: Beth definability in expressive description logics. In: Proc. of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011). (2011) 1099–1106
12. Beth, E.: On Padoa's method in the theory of definition. Indagationes Mathematicae **15** (1953) 330–339
13. Horrocks, I., Sattler, U.: Ontology reasoning in the SHOQ(D) description logic. In: In Proc. of the 17th Int. Joint Conf. on Artificial Intelligence (IJCAI 2001). (2001) 199–204
14. Lutz, C., Areces, C., Horrocks, I., Sattler, U.: Keys, nominals, and concrete domains. J. Artif. Int. Res. **23** (June 2005) 667–726