Journal of Artificial Intelligence Research 48 (2013) Submitted 5/2013; published 12/2013

Exact Query Reformulation over Databases with First-order and Description Logics Ontologies

Enrico Franconi FRANCONI@INF.UNIBZ.IT
Volha Kerhet KERHET@INF.UNIBZ.IT
Nhung Ngo NGO@INF.UNIBZ.IT
Free University of Bozen-Bolzano, Italy

Abstract

We study a general framework for query rewriting in the presence of an arbitrary first-order logic ontology over a database signature. The framework supports deciding the existence of a safe-range first-order equivalent reformulation of a query in terms of the database signature, and if so, it provides an effective approach to construct the reformulation based on interpolation using standard theorem proving techniques (e.g., tableau). Since the reformulation is a safe-range formula, it is effectively executable as an SQL query. At the end, we present a non-trivial application of the framework with ontologies in the very expressive $\mathcal{ALCHOIQ}$ description logic, by providing effective means to compute safe-range first-order exact reformulations of queries.

1. Introduction

We address the problem of query reformulation with expressive ontologies over databases. An ontology provides a conceptual view of the database and it is composed by constraints on a vocabulary *extending* the basic vocabulary of the data. Querying a database using the terms in such a richer ontology allows for more flexibility than using only the basic vocabulary of the relational database directly.

In this paper we study and develop a query rewriting framework applicable to knowledge representation systems where data is stored in a classical finite relational database, in a way that in the literature has been called the *locally-closed world* assumption (Etzioni, Golden, & Weld, 1997), *exact views* (Marx, 2007; Nash, Segoufin, & Vianu, 2010; Fan, Geerts, & Zheng, 2012), or *DBox* (Seylan, Franconi, & de Bruijn, 2009; Franconi, Ibanez-Garcia, & Seylan, 2011). 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 ontology are *open*, i.e., their extensions may vary among different interpretations. We do not consider here the *open* interpretation for the database predicates (also called *ABox* or *sound views*). In an ABox, the interpretation of database predicates contains the database relations and possibly more. This notion 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 set of first-order formulas, and queries are (possibly open) first-order formulas. Within this setting, the framework provides precise semantic conditions to decide the existence of a safe-range first-order equivalent reformulation of a query in terms of the database signature. It also provides an effective approach to construct the reformulation with sufficient conditions. We are 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 (Abiteboul, Hull, & Vianu, 1995). Our framework points out several conditions on the ontologies and the queries to guarantee the existence of a safe-range reformulation. We show that these conditions are feasible in practice and we also provide an efficient method to ensure their validation. Standard theorem proving 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 finitely controllable determinacy (Nash et al., 2010), a stronger property than the finite model property of the logic. If the employed logic does not enjoy finitely controllable determinacy our approach would become sound but incomplete, but 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 $\mathcal{ALCHOIQ}$ ontologies 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 the paper by Seylan et al. (2009).

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 (Halevy, 2001), which is mostly based on perfect or maximally contained rewritings with sound views under relatively inexpressive constraints (see, e.g., the DL-Lite approach in Artale, Calvanese, Kontchakov, & Zakharyaschev, 2009), we focus here on exact rewritings with exact views, since it characterises precisely the query answering problem with ontologies and databases, in the case when the exact semantics of the database must be preserved. As an example, consider a ground negative query over a given standard relational database; by adding an ontology on top of it, its answer is not supposed to change—since the query uses only the signature of the database and additional constraints are not supposed to change the meaning of the query—whereas if the database were treated as an ABox (sound views) the answer may change in presence of an ontology. This may be important from the application perspective: a DBox preserves the behaviour of the legacy application queries over a relational database. Moreover, by focussing on exact reformulations of *definable* queries (as opposed to considering the certain answer semantics to arbitrary queries, such as in DL-Lite), we guarantee that answers to queries can be subsequently composed in an arbitrary way: this may be important to legacy database applications.

This work extends the works on exact rewritings with exact views by Marx (2007) and Nash et al. (2010) by focussing on *safe-range reformulations* and on the conditions ensuring their existence, and by considering general first-order ontologies extending the database signature, rather than just *local as view* constraints over the database predicates (Halevy, 2001). This paper extends the papers by Franconi, Kerhet, and Ngo (2012a, 2012b) by providing a precise semantic characterisation for the existence of an exact reformulation (Theorem 4) as opposed to just sufficient conditions, by considering the much more expressive description logic $\mathcal{ALCHOIQ}$, and by providing all the proofs.

The paper is organised as follows: section 2 provides the necessary formal background and definitions; section 3 introduces the notion of a query determined by a database; section 4 introduces a characterisation of the query reformulation problem; in sections 5 and 6 the conditions allowing for an effective reformulation are analysed, and a sound and complete algorithm to compute the reformulation is introduced. Finally, we present the case of $\mathcal{ALCHOIQ}$ ontologies. All the proofs are presented in details in the Appendix.

2. Preliminaries

Let $\mathcal{FOL}(\mathbb{C},\mathbb{P})$ be a classical function-free first-order language with equality over a signature $\Sigma = (\mathbb{C},\mathbb{P})$, where $\mathbb{C}$ is a finite set of *constants* and $\mathbb{P}$ is a set of *predicates* with associated arities. In the rest of this paper we will refer to an arbitrary fragment of $\mathcal{FOL}(\mathbb{C},\mathbb{P})$, which will be called $\mathcal{L}$.

We denote with $\mathbb{P}_{\{\varphi_1,\ldots,\varphi_n\}}$ the set of all predicates occurring in the formulas $\varphi_1,\ldots,\varphi_n$, with $\mathbb{C}_{\{\varphi_1,\ldots,\varphi_n\}}$ the set of all constants occurring in the formulas $\varphi_1,\ldots,\varphi_n$; for the sake of brevity, instead of $\mathbb{P}_{\{\varphi\}}$ (resp. $\mathbb{C}_{\{\varphi\}}$) we write $\mathbb{P}_\varphi$ (resp. $\mathbb{C}_\varphi$). We denote with $\sigma(\varphi_1,\ldots,\varphi_n)$ the signature of the formulas $\varphi_1,\ldots,\varphi_n$, namely the union of $\mathbb{P}_{\{\varphi_1,\ldots,\varphi_n\}}$ and $\mathbb{C}_{\{\varphi_1,\ldots,\varphi_n\}}$. We denote the arity of a predicate P as $\textsc{ar}(P)$. Given a formula φ, we denote the set of all variables appearing in φ as $\textsc{var}(\varphi)$, and the set of the free variables appearing in φ as $\textsc{free}(\varphi)$; we may use for φ the notation $\varphi_{[\mathbb{X}]}$, where $\mathbb{X} = \textsc{free}(\varphi)$ is the (possibly empty) set of free variables of the formula.

A *database (instance)* $\mathcal{DB}$ is a *finite* set of ground atoms of the form $P(c_1,\ldots,c_n)$, where $P \in \mathbb{P}$, n-ary predicate, and $c_i \in \mathbb{C}$ $(1 \leq i \leq n)$. The set of all predicates appearing in a database $\mathcal{DB}$ is denoted as $\mathbb{P}_{\mathcal{DB}}$, and the set of all constants appearing in $\mathcal{DB}$ is called the *active domain of* $\mathcal{DB}$, and is denoted as $\mathbb{C}_{\mathcal{DB}}$. A (possibly empty) finite set $\mathcal{KB}$ of closed formulas will be called an *ontology*.

As usual, an *interpretation* $\mathcal{I} = \langle\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\rangle$ includes a non-empty set—the domain $\Delta^{\mathcal{I}}$—and an interpretation function $\cdot^{\mathcal{I}}$ defined over constants and predicates of the signature. We say that interpretations $\mathcal{I} = \langle\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\rangle$ and $\mathcal{J} = \langle\Delta^{\mathcal{J}}, \cdot^{\mathcal{J}}\rangle$ are *equal*, written $\mathcal{I} = \mathcal{J}$, if $\Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$ and $\cdot^{\mathcal{I}} = \cdot^{\mathcal{J}}$. An interpretation $\mathcal{I}$ *embeds a database* $\mathcal{DB}$, if it holds that $a^{\mathcal{I}} = a$ for every database constant $a \in \mathbb{C}_{\mathcal{DB}}$ (the *standard name assumption (SNA)*, customary in databases, see Abiteboul et al., 1995) and that $(c_1,\ldots,c_n) \in P^{\mathcal{I}}$ if and only if $P(c_1,\ldots,c_n) \in \mathcal{DB}$. We denote the set of all interpretations embedding a database $\mathcal{DB}$ as $E(\mathcal{DB})$.

In other words, in every interpretation embedding a $\mathcal{DB}$ the interpretation of any database predicate is always the same and it is given exactly by its content in the database; this is, in general, not the case for the interpretation of the non-database predicates. We say that all the database predicates are *closed*, while all the other predicates are *open* and may be interpreted differently in different interpretations. We do not consider here the *open world* assumption (the *ABox*) for embedding a database in an interpretation. In an open world, an interpretation $\mathcal{I}$ soundly embeds a database if it holds that $(c_1,\ldots,c_n) \in P^{\mathcal{I}}$ if (but *not* only if) $P(c_1,\ldots,c_n) \in \mathcal{DB}$.

In order to allow for an arbitrary database to be embedded, we generalise the standard name assumption to all the constants in $\mathbb{C}$; this implies that the domain of any interpretation necessarily includes the set of all the constants $\mathbb{C}$. The finiteness of $\mathbb{C}$ corresponds to the finite ability of a database system to represent distinct constant symbols; $\mathbb{C}$ is meant to be unknown in advance, since different database systems may have different limits. We will see that the framework introduced here will not depend on the choice of $\mathbb{C}$.

Given an interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, we denote as $\mathcal{I}|_{\mathbb{S}}$ the interpretation restricted to the smaller signature $\mathbb{S} \subseteq \mathbb{P} \cup \mathbb{C}$, i.e., the interpretation with the same domain $\Delta^{\mathcal{I}}$ and the same interpretation function $\cdot^{\mathcal{I}}$ defined only for the constants and predicates from the set $\mathbb{S}$. The *semantic active domain* of a signature $\sigma' \subseteq \mathbb{P} \cup \mathbb{C}$ in an interpretation $\mathcal{I}$, denoted $adom(\sigma', \mathcal{I})$, is the set of all elements of the domain $\Delta^{\mathcal{I}}$ occurring in interpretations of predicates and constants from σ' in $\mathcal{I}$:

$$adom(\sigma', \mathcal{I}) := \bigcup_{P \in \sigma'} \bigcup_{(a_1,\ldots,a_n) \in P^{\mathcal{I}}} \{a_1, \ldots, a_n\} \cup \bigcup_{c \in \sigma'} \{c^{\mathcal{I}}\}.$$

If $\sigma' \subseteq \mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}$, then for any interpretations $\mathcal{I}$ and $\mathcal{J}$ embedding $\mathcal{DB}$ we have: $adom(\sigma', \mathcal{I}) = adom(\sigma', \mathcal{J})$; so, for such a case we introduce the notation $adom(\sigma', \mathcal{DB}) := adom(\sigma', \mathcal{I})$, where $\mathcal{I}$ is any interpretation embedding the database $\mathcal{DB}$. Intuitively $adom(\sigma', \mathcal{DB})$ includes the constants from σ' and from $\mathcal{D}B$ appearing in the relations corresponding to the predicates from σ'.

Let $\mathbb{X}$ be a set of variable symbols and $\mathbb{S}$ a set; a *substitution* is a total function $\Theta : \mathbb{X} \mapsto \mathbb{S}$ assigning an element in $\mathbb{S}$ to each variable in $\mathbb{X}$, including the empty substitution ϵ when $\mathbb{X} = \emptyset$. Domain and image (range) of a substitution Θ are written as $dom(\Theta)$ and $rng(\Theta)$ respectively. Given a subset of the set of constants $\mathbb{C}' \subseteq \mathbb{C}$, we write that a formula $\varphi_{[\mathbb{X}]}$ is true in an interpretation $\mathcal{I}$ with its free variables substituted according to a substitution $\Theta : \mathbb{X} \mapsto \mathbb{C}'$ as $(\mathcal{I} \models \varphi_{[\mathbb{X}/\Theta]})$. Given an interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ and a subset of its domain $\Delta \subseteq \Delta^{\mathcal{I}}$, we write that a formula $\varphi_{[\mathbb{X}]}$ is true in $\mathcal{I}$ with its free variables interpreted according to a substitution $\Theta : \mathbb{X} \mapsto \Delta$ as $(\mathcal{I}, \Theta \models \varphi)$. The *extension domain* of a formula $\varphi_{[\mathbb{X}]}$ with respect to the interpretation $\mathcal{I}$ is defined as the set of domain elements $\bigcup\{rng(\Theta) \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) \subseteq \Delta,\ \mathcal{I}, \Theta \models \varphi_{[\mathbb{X}]}\}$.

As usual, an interpretation in which a closed formula is true is called a *model* for the formula; the set of all models of a formula φ (resp. $\mathcal{KB}$) is denoted as $M(\varphi)$ (resp. $M(\mathcal{KB})$). A *database $\mathcal{DB}$ is legal for an ontology $\mathcal{KB}$* if there exists a model of $\mathcal{K}B$ embedding $\mathcal{D}B$. In the following, we will consider only consistent non-tautological ontologies and legal databases.

2.1 Queries

A *query* is a (possibly closed) formula. Given a query $\mathcal{Q}_{[\mathbb{X}]}$, we define its *certain answer* over $\mathcal{KB}$ and $\mathcal{DB}$ as follows:

Definition 1 (**Certain Answer**)**.** The *(certain) answer of a query $\mathcal{Q}_{[\mathbb{X}]}$ to a database $\mathcal{DB}$ under the ontology $\mathcal{KB}$* is the set of substitutions with constants:

$$\{\Theta \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) \subseteq \mathbb{C},\ \forall \mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB}) : \mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\Theta]}\}.$$

Query answering is defined as an *entailment* problem, and as such it is going to have the same (high) complexity as entailment.

Note, that if a query $\mathcal{Q}$ is closed (i.e., a Boolean query), then the certain answer is $\{\epsilon\}$ if $\mathcal{Q}$ is true in all the models of the ontology embedding the database, and $\emptyset$ otherwise. In the following, we assume that the closed formula $\mathcal{Q}_{[\mathbb{X}/\Theta]}$ is neither valid nor inconsistent under the ontology $\mathcal{KB}$, given a substitution $\Theta : \mathbb{X} \mapsto \mathbb{C}$ assigning to variables *distinct* constants not appearing in $\mathcal{Q}$, nor in $\mathcal{KB}$, nor in $\mathbb{C}_{\mathcal{DB}}$: this would lead to trivial reformulations.

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 $\mathcal{I}$ satisfies the standard name assumption if $c^{\mathcal{I}} = c$ for any $c \in \mathbb{C}$. Alternatively, an interpretation $\mathcal{I}$ satisfies the unique name assumption (UNA) if $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ for any different $a, b \in \mathbb{C}$. We denote the set of all interpretations satisfying standard name assumption as $I(\mathit{SNA})$. We denote the set of all interpretations satisfying unique name assumption as $I(\mathit{UNA})$. The following proposition allows us to freely interchange the standard name and the unique name assumptions with interpretations embedding databases. This is of practical advantage, since we can encode the unique name assumption in classical first-order logic reasoners, and many description logics reasoners do support natively the unique name assumption as an extension to OWL.

Proposition 1 (**SNA vs UNA**). *For any query $\mathcal{Q}_{[\mathbb{X}]}$, ontology $\mathcal{KB}$ and database $\mathcal{DB}$,*

$$\{\Theta \mid \mathit{dom}(\Theta) = \mathbb{X},\ \mathit{rng}(\Theta) \subseteq \mathbb{C},\ \forall \mathcal{I} \in I(\mathit{SNA}) \cap M(\mathcal{KB}) \cap E(\mathcal{DB}) : \mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\Theta]}\} =$$
$$\{\Theta \mid \mathit{dom}(\Theta) = \mathbb{X},\ \mathit{rng}(\Theta) \subseteq \mathbb{C},\ \forall \mathcal{I} \in I(\mathit{UNA}) \cap M(\mathcal{KB}) \cap E(\mathcal{DB}) : \mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\Theta]}\}.$$

Since a query can be an arbitrary first-order formula, its answer may depend on the domain, which we do not know in advance. For example, the query $Q(x) = \neg \mathit{Student}(x)$ over the database $\mathit{Student}(a), \mathit{Student}(b)$, with domain $\{a, b, c\}$ has the answer $\{x = c\}$, while with domain $\{a, b, c, d\}$ has the answer $\{x = c, x = d\}$. Therefore, the notion of *domain independent* queries has been introduced in relational databases. Here we adapt the classical definitions (Avron, 2008; Abiteboul et al., 1995) to our framework: we need a more general version of domain independence, namely domain independence w.r.t an ontology, i.e., restricted to the models of an ontology.

Definition 2 (**Domain Independence**). A formula $\mathcal{Q}_{[\mathbb{X}]}$ is *domain independent with respect to an ontology* $\mathcal{KB}$ iff for every two models $\mathcal{I}$ and $\mathcal{J}$ of $\mathcal{KB}$ (i.e., $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ and $\mathcal{J} = \langle \Delta^{\mathcal{J}}, \cdot^{\mathcal{J}} \rangle$) which agree on the interpretation of the predicates and constants (i.e. $\cdot^{\mathcal{I}} = \cdot^{\mathcal{J}}$), and for every substitution $\Theta : \mathbb{X} \mapsto \Delta^{\mathcal{I}} \cup \Delta^{\mathcal{J}}$ we have:

$$\begin{array}{l} \mathit{rng}(\Theta) \subseteq \Delta^{\mathcal{I}} \text{ and } \mathcal{I}, \Theta \models \mathcal{Q}_{[\mathbb{X}]} \quad \text{iff} \\ \mathit{rng}(\Theta) \subseteq \Delta^{\mathcal{J}} \text{ and } \mathcal{J}, \Theta \models \mathcal{Q}_{[\mathbb{X}]}. \end{array}$$

The above definition reduces to the classical definition of domain independence whenever the ontology is empty.

A weaker version of domain independence—which is relevant for open formulas—is the following.

Definition 3 (**Ground Domain Independence**). A formula $\mathcal{Q}_{[\mathbb{X}]}$ is *ground domain independent* iff $\mathcal{Q}_{[\mathbb{X}/\Theta]}$ is domain independent for every substitution $\Theta : \mathbb{X} \mapsto \mathbb{C}$.

For example, the formula $\neg P(x)$ is ground domain independent, but it is not domain independent.

The problem of checking whether a FOL formula is domain independent is undecidable (Abiteboul et al., 1995). 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 full details see Appendix A.3). For example, the formula $\neg A(x) \land B(x)$ 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 this coincides with the free variables then the formula is said to be safe-range (Abiteboul et al., 1995). Similar to domain independence, a formula is *ground safe-range* if any grounding of this formula is safe-range. An ontology $\mathcal{KB}$ is safe-range (domain independent), if every formula in $\mathcal{KB}$ is safe-range (domain independent).

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 SQL (Abiteboul et al., 1995).

3. Determinacy

The certain answer to a query includes all the substitutions which make the query true in *all* the models of the ontology embedding the database: so, if a substitution 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 ontology embedding the database. 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 query with respect to the data, the complexity to compute the certain answer in general increases up 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 embedding the database, namely, 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 (see section 4). The *determinacy* of a query w.r.t. a source database (Nash et al., 2010; Marx, 2007; Fan et al., 2012) has been called *implicit definability* of a formula (the query) from a set of predicates (the database predicates) by Beth (1953).

Definition 4 (**Finite Determinacy** or **Implicit Definability**). A query $\mathcal{Q}_{[\mathbb{X}]}$ is (finitely) *determined* by (or *implicitly definable* from) the database predicates $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$ *iff* for any two models $\mathcal{I}$ and $\mathcal{J}$ of the ontology $\mathcal{KB}$—both with a *finite* interpretation to the database predicates $\mathbb{P}_{\mathcal{DB}}$—whenever $\mathcal{I}|_{\mathbb{P}_{\mathcal{DB}}\cup\mathbb{C}} = \mathcal{J}|_{\mathbb{P}_{\mathcal{DB}}\cup\mathbb{C}}$ then for every substitution $\Theta : \mathbb{X} \mapsto \Delta^{\mathcal{I}}$ we have: $\mathcal{I}, \Theta \models \mathcal{Q}_{[\mathbb{X}]}$ iff $\mathcal{J}, \Theta \models \mathcal{Q}_{[\mathbb{X}]}$.

Intuitively, the answer of an implicitly 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 a substitution would make the query true in some model of the ontology 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). In practice, by focussing on finite determinacy of queries we guarantee that the user can always interpret the answers as being not only certain, but also *exact*—namely that whatever is not in the answer can never be part of the answer in any possible world.

In the following we focus on ontologies and queries in those fragments of $\mathcal{FOL}(\mathbb{C},\mathbb{P})$ for which determinacy under models with a finite interpretation of database predicates (finite determinacy) and determinacy under models with an unrestricted interpretation of database predicates (unrestricted determinacy) coincide. We say that these fragments have *finitely controllable determinacy*: we require that whenever a query is finitely determined then it is also determined in unrestricted models (the reverse is trivially true). Indeed, the results in this paper would fail if finite determinacy and unrestricted determinacy do not coincide: it can be shown (Gurevich, 1984) that Theorem 1 below fails if we consider only models with a finite interpretation of database predicates.

Example 1 (Example from database theory)**.** Let $\mathbb{P} = \{P, R, A\}$, $\mathbb{P}_{\mathcal{DB}} = \{P, R\}$,

$$\mathcal{KB} = \{\forall x, y, z.\, R(x, y) \land R(x, z) \to y = z, \qquad \forall x, y.\, R(x, y) \to \exists z.\, R(z, x),$$
$$(\forall x, y.\, R(x, y) \to \exists z.\, R(y, z)) \to (\forall x.\, A(x) \leftrightarrow P(x))\}.$$

The formula $\forall x, y.\, R(x, y) \to \exists z.\, R(y, z)$ is entailed from the first two formulas *only* over finite interpretations of R. The query $\mathcal{Q} = A(x)$ is finitely determined by P (it is equivalent to $P(x)$ under the models with a finite interpretation of R), but it is not determined by any database predicate under models with an unrestricted interpretation of R. This knowledge base does not enjoy finitely controllable determinacy.

The *exact reformulation* of a query (Nash et al., 2010) (also called *explicit definition* by Beth, 1953) is a formula logically equivalent to the query which makes use *only* of database predicates and constants.

Definition 5 (**Exact Reformulation** or **Explicit Definability**)**.** A query $\mathcal{Q}_{[\mathbb{X}]}$ is *explicitly definable from the database predicates* $\mathbb{P}_{\mathcal{DB}}$ *under the ontology* $\mathcal{KB}$ iff there is some formula $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ in $\mathcal{FOL}(\mathbb{C},\mathbb{P})$, such that $\mathcal{KB} \models \forall \mathbb{X}. \mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \widehat{\mathcal{Q}}_{[\mathbb{X}]}$ and $\sigma(\widehat{\mathcal{Q}}) \subseteq \mathbb{P}_{\mathcal{DB}}$. We call this formula $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ an *exact reformulation of* $\mathcal{Q}_{[\mathbb{X}]}$ *under* $\mathcal{KB}$ *over* $\mathbb{P}_{\mathcal{DB}}$.

Determinacy of a query is completely characterised by the existence of an exact reformulation of the query: it is well known that a first-order query is determined by database predicates *if and only if* there exists a first-order exact reformulation.

Theorem 1 (**Projective Beth definability**, Beth, 1953)**.** *A query $\mathcal{Q}$ is implicitly definable from the database predicates $\mathbb{P}_{\mathcal{DB}}$ under an ontology $\mathcal{KB}$, iff it is explicitly definable as a formula $\widehat{\mathcal{Q}}$ in $\mathcal{FOL}(\mathbb{C},\mathbb{P})$ over $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$.*

Let $\mathcal{Q}$ be any formula in $\mathcal{L}$ and $\widetilde{\mathcal{Q}}$ the formula obtained from it by uniformly replacing every occurrence of each non-database predicate P with a new predicate $\widetilde{P}$. We extend this renaming operator $\widetilde{\cdot}$ to any set of formulas in a natural way. One can check whether a query is implicitly definable by using the following theorem.

Theorem 2 (**Testing Determinacy**, Beth, 1953)**.** *A query $\mathcal{Q}_{[\mathbb{X}]}$ is implicitly definable from the database predicates $\mathbb{P}_{\mathcal{DB}}$ under the ontology $\mathcal{KB}$ iff $\mathcal{KB} \cup \widetilde{\mathcal{KB}} \models \forall \mathbb{X}. \mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \widetilde{\mathcal{Q}}_{[\mathbb{X}]}$.*

4. Exact Safe-Range Query Reformulation

In this section we analyse the conditions under which the original query answering problem corresponding to an entailment problem can be reduced systematically to a model checking problem of a safe-range formula over the database (e.g., using a database system with SQL). Given a database signature $\mathbb{P}_{\mathcal{DB}}$, an ontology $\mathcal{KB}$, and a query $\mathcal{Q}_{[\mathbb{X}]}$ expressed in $\mathcal{L}$ and determined by the database predicates, our goal is to find a safe-range reformulation $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ of $\mathcal{Q}_{[\mathbb{X}]}$ in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$, that when evaluated as a relational algebra expression over a legal database instance, gives the same answer as the certain answer of $\mathcal{Q}_{[\mathbb{X}]}$ to the database under $\mathcal{KB}$. This can be reformulated as the following problem:

Problem 1 (**Exact safe-range Query Reformulation**). Find an exact reformulation $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ of $\mathcal{Q}_{[\mathbb{X}]}$ under $\mathcal{KB}$ as a safe-range query in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ over $\mathbb{P}_{\mathcal{DB}}$.

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. *Given a database $\mathcal{DB}$, let $\mathcal{Q}_{[\mathbb{X}]}$ be implicitly definable from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$ and let $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ be an exact reformulation of $\mathcal{Q}_{[\mathbb{X}]}$ under $\mathcal{KB}$ over $\mathbb{P}_{\mathcal{DB}}$, then:*

$$\{\Theta \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) \subseteq \mathbb{C},\ \forall \mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB})\ :\ \mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\Theta]}\} =$$
$$\{\Theta \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) \subseteq \mathbb{C},\ \forall \mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB})\ :\ \mathcal{I} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\Theta]}\}.$$

From the above equation it is clear that in order to answer an exactly reformulated query, one may still need to consider all the models of the ontology embedding the database, 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 domain independent. Indeed there is only one interpretation (with a particular domain) embedding the database with the signature restricted to the database predicates.

Theorem 3 (**Adequacy of Exact safe-range Query Reformulation**). *Let $\mathcal{DB}$ be a database which is legal for $\mathcal{KB}$, and let $\mathcal{Q}_{[\mathbb{X}]}$ be a query. If $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is an exact domain independent (or safe-range) reformulation of $\mathcal{Q}_{[\mathbb{X}]}$ under $\mathcal{KB}$ over $\mathbb{P}_{\mathcal{DB}}$, then:*

$$\{\Theta \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) \subseteq \mathbb{C},\ \forall \mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB})\ :\ \mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\Theta]}\} =$$
$$\{\Theta \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) \subseteq adom(\sigma(\widehat{\mathcal{Q}}), \mathcal{DB}),\ \forall \mathcal{I} = \langle \mathbb{C}, \cdot^{\mathcal{I}} \rangle \in E(\mathcal{DB}) : \mathcal{I}|_{\mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\Theta]}\}.$$

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. Let us now see an example in which we cannot reduce the problem of answering an exact reformulation to model checking over a database, if the exact reformulation is not safe-range.

Example 2. Let $\mathbb{P} = \{P, A\}$, $\mathbb{P}_{\mathcal{DB}} = \{P\}$, $\mathbb{C} = \{a\}$,
$\mathcal{DB} = \{P(a, a)\}$, $\mathcal{KB} = \{\forall y. P(a, y) \vee A(y)\}$,
$\mathcal{Q}_{[\mathbb{X}]} = \widehat{\mathcal{Q}}_{[\mathbb{X}]} = \forall y. P(x, y)$ (i.e., $\mathbb{X} = \{x\}$).

- $\mathbb{C}$ includes the active domain $\mathbb{C}_{\mathcal{DB}}$ (it is actually equal).
- $\mathcal{DB}$ is legal for $\mathcal{KB}$ because there is $\mathcal{I} = \langle \{a\}, \cdot^{\mathcal{I}} \rangle$ such that $P^{\mathcal{I}} = \{(a,a)\}$, $A^{\mathcal{I}} = \emptyset$ and obviously, $\mathcal{I} \in M(\mathcal{KB})$.
- $\{\Theta \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) \subseteq \mathbb{C},\ \forall \mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB}) : \mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\Theta]}\} = \emptyset$ because one can take $\mathcal{I} = \langle \{a,b\}, \cdot^{\mathcal{I}} \rangle$ such that $P^{\mathcal{I}} = \{(a,a)\}$, $A^{\mathcal{I}} = \{b\}$; then $\mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB})$, but for the only possible substitution $\{x \to a\}$ we have: $\mathcal{I} \not\models \forall y\, P(a,y)$.
- However,
  $\{\Theta \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) \subseteq adom(\sigma(\widehat{\mathcal{Q}}), \mathcal{DB}),\ \forall \mathcal{I} = \langle \mathbb{C}, \cdot^{\mathcal{I}} \rangle \in E(\mathcal{DB}) : \mathcal{I}|_{\mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\Theta]}\} =$
  $\{x \to a\}$ □

As we have seen, answers to a query for which a reformulation exists will contain only constants from the active domain of the database and the query; therefore, ground statements in the ontology involving non-database predicates and non-active domain constants (for example, as ABox statements) will not play any role in the final evaluation of the reformulated query over the database.

5. Conditions for an Exact Safe-Range Reformulation

We have just seen the importance of getting an exact safe-range query reformulation. In this section we are going to study the conditions under which an exact safe-range query reformulation exists.

First of all, we will focus on the semantic notion of safe-range namely domain independence. While implicit definability is—as we already know—a sufficient condition for the existence of an exact reformulation, it does not guarantee alone the existence of a domain independent reformulation.

Example 3. Let $\mathbb{P} = \{A, B\}$, $\mathbb{P}_{\mathcal{DB}} = \{A\}$, $\mathcal{KB} = \{\forall x. B(x) \leftrightarrow A(x)\}$, $\mathcal{Q} = \neg B(x)$.
Then $\mathcal{Q}$ is implicitly definable from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$, and every exact reformulation of $\mathcal{Q}$ over $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$ is logically equivalent to $\neg A(x)$ and not domain independent. □

By looking at the example, it seems that the reason for the non domain independent reformulation lies in the fact that the ontology, which is domain independent, cannot guarantee existence of an exact domain independent reformulation of the non domain independent query. However, let us consider the following example:

Example 4. Let $\mathbb{P}_{\mathcal{DB}} = \{A, C\}$, $\mathcal{KB} = \{\neg A(a), \forall x.\, A(x) \leftrightarrow B(x)\}$ and let a query $\mathcal{Q} = \exists y\, \neg B(y) \wedge C(x)$. It is easy to see that $\mathcal{KB}$ is domain independent and $\mathcal{Q}$ is not. $\mathcal{Q}$ is implicitly definable from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$, and $\widehat{\mathcal{Q}} = \neg A(a) \wedge C(x)$ is an exact domain independent reformulation of $\mathcal{Q}$. □

It is obvious that in spite of the fact that the query $\mathcal{Q}$ is not domain independent, it is domain independent with respect to the ontology $\mathcal{KB}$. In other words, in this case the ontology guarantees the existence of an exact domain independent reformulation.

With queries that are domain independent with respect to an ontology, the following theorem holds, giving the *semantic* requirements for the existence of an exact domain independent reformulation.

Theorem 4 (**Semantic Characterisation**). *Given a set of database predicates $\mathbb{P}_{\mathcal{DB}}$, a domain independent ontology $\mathcal{KB}$, and a query $\mathcal{Q}_{[\mathbb{X}]}$, a domain independent exact reformulation $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ of $\mathcal{Q}_{[\mathbb{X}]}$ over $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$ exists if and only if $\mathcal{Q}_{[\mathbb{X}]}$ is implicitly definable from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$ and it is domain independent with respect to $\mathcal{KB}$.*

The above theorem shows us the semantic conditions to have an exact domain independent reformulation of a query, but it does not give us a method to compute such reformulation and its equivalent safe-range form. The following theorem gives us sufficient conditions for the existence of an exact safe-range reformulation in any decidable fragment of $\mathcal{FOL}(\mathbb{C},\mathbb{P})$ where finite and unrestricted determinacy coincide, and gives us a constructive way to compute it, if it exists.

Theorem 5 (**Constructive**). *If:*

1. $\mathcal{KB} \cup \widetilde{\mathcal{KB}} \models \forall \mathbb{X}.\, \mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \widetilde{\mathcal{Q}}_{[\mathbb{X}]}$ *(that is, $\mathcal{Q}_{[\mathbb{X}]}$ is implicitly definable),*
2. $\mathcal{Q}_{[\mathbb{X}]}$ *is safe-range (that is, $\mathcal{Q}_{[\mathbb{X}]}$ is domain independent),*
3. $\mathcal{KB}$ *is safe-range (that is, $\mathcal{KB}$ is domain independent),*

then there exists an exact reformulation $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ of $\mathcal{Q}_{[\mathbb{X}]}$ as a safe-range query in $\mathcal{FOL}(\mathbb{C},\mathbb{P})$ over $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$, that can be obtained constructively.

In order to constructively compute the exact safe-range query reformulation we use the tableau based method to find the Craig's interpolant (Fitting, 1996) to compute $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ from a validity proof of the implication $(\mathcal{KB} \wedge \mathcal{Q}_{[\mathbb{X}]}) \rightarrow (\widetilde{\mathcal{KB}} \rightarrow \widetilde{\mathcal{Q}}_{[\mathbb{X}]})$. See Section 6 for full details.

Let us now consider a fully worked out example, adapted from the paper by Nash et al. (2010).

Example 5. Given: $\mathbb{P} = \{R, V_1, V_2, V_3, A\}$, $\mathbb{P}_{\mathcal{DB}} = \{V_1, V_2, V_3, Adom\}$ where $Adom$ is the active domain of $\mathcal{DB}$,

$$\begin{aligned}\mathcal{KB} = \{\, & \forall x,y.\, V_1(x,y) \leftrightarrow \exists z,v.\, R(z,x) \wedge R(z,v) \wedge R(v,y),\\ & \forall x,y.\, V_2(x,y) \leftrightarrow \exists z.\, R(x,z) \wedge R(z,y),\\ & \forall x,y.\, V_3(x,y) \leftrightarrow \exists z,v.\, R(x,z) \wedge R(z,v) \wedge R(v,y),\\ & \mathcal{Q}(x,y) = \exists z,v,u.\, R(z,x) \wedge R(z,v) \wedge R(v,u) \wedge R(u,y)\}.\end{aligned}$$

The conditions of the theorem are satisfied: $\mathcal{Q}(x,y)$ is implicitly definable from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$; $\mathcal{Q}(x,y)$ is safe-range; $\mathcal{KB}$ is safe-range.
Therefore, with the tableau method one finds the Craig's interpolant to compute $\widehat{\mathcal{Q}}(x,y)$ from a validity proof of the implication $(\mathcal{KB} \wedge \mathcal{Q}_{[\mathbb{X}]}) \rightarrow (\widetilde{\mathcal{KB}} \rightarrow \widetilde{\mathcal{Q}}_{[\mathbb{X}]})$ and obtain $\widehat{\mathcal{Q}}(x,y) = \exists z.\, V_1(x,z) \wedge \forall v.\, (V_2(v,z) \rightarrow V_3(v,y))$—an exact ground safe-range reformulation. Since the answer of Q is in the active domain, we also have $\mathcal{KB} \models \widehat{\mathcal{Q}}(x,y) \rightarrow Adom(x) \wedge Adom(y)$. Then $\mathcal{KB} \models \mathcal{Q}(x,y) \leftrightarrow \widehat{\mathcal{Q}}(x,y) \wedge Adom(x) \wedge Adom(y)$. Therefore, $\exists z.\, V_1(x,z) \wedge \forall v.\, (V_2(v,z) \rightarrow V_3(v,y)) \wedge Adom(x) \wedge Adom(y)$ is an exact safe-range reformulation of $\mathcal{Q}(x,y)$ from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$. $\square$

6. Constructing the Safe-Range Reformulation

In this section we introduce a method to compute a safe-range reformulation of an implicitly definable query when conditions in theorem 5 are satisfied. The method is based on the notion of interpolant introduced by Craig (1957).

Definition 6 (Interpolant). The sentence χ is an *interpolant* for the sentence $\phi \to \psi$ in $\mathcal{FOL}(\mathbb{C},\mathbb{P})$, if all predicate and constant symbols of χ are in the set of predicate and constant symbols of both ϕ and ψ, and both $\phi \to \chi$ and $\chi \to \psi$ are valid sentences in $\mathcal{FOL}(\mathbb{C},\mathbb{P})$.

Theorem 6 (Craig's interpolation). *If $\phi \to \psi$ is a valid sentence in $\mathcal{FOL}(\mathbb{C},\mathbb{P})$, and neither ϕ nor ψ are valid, then there exists an interpolant.*

Note, that the Beth definability (Theorem 1) and Craig's interpolation theorem do not hold for all fragments of $\mathcal{FOL}(\mathbb{C},\mathbb{P})$: an interpolant may not always be expressed in the fragment itself, but obviously it is in $\mathcal{FOL}(\mathbb{C},\mathbb{P})$ (because of Theorem 6).

An interpolant is used to find an exact reformulation of a given implicitly definable query as follows.

Theorem 7 (Interpolant as definition). *Let $\mathcal{Q}_{[\mathbb{X}]}$ be a query with $n \geq 0$ free variables implicitly definable from the database predicates $\mathbb{P}_{\mathcal{DB}}$ under the ontology $\mathcal{KB}$. Then, the closed formula with $c_1,\dots,c_n$ distinct constant symbols in $\mathbb{C}$ not appearing in $\mathcal{KB}$ or $\mathcal{Q}_{[\mathbb{X}]}$:*

$$((\bigwedge \mathcal{KB}) \wedge \mathcal{Q}_{[\mathbb{X}/c_1,\dots,c_n]}) \to ((\bigwedge \widetilde{\mathcal{KB}}) \to \widetilde{\mathcal{Q}}_{[\mathbb{X}/c_1,\dots,c_n]}) \tag{1}$$

is valid, and its interpolant $\widehat{\mathcal{Q}}_{[c_1,\dots,c_n/\mathbb{X}]}$ is an exact reformulation of $\mathcal{Q}_{[\mathbb{X}]}$ under $\mathcal{KB}$ over $\mathbb{P}_{\mathcal{DB}}$.

Therefore, to find an exact reformulation of an implicitly definable query in terms of database predicates it is enough to find an interpolant of the implication (1) and then to substitute all the constants $c_1,\dots,c_n$ back with the free variables $\mathbb{X}$ of the original query. An interpolant can be constructed from a validity proof of (1) by using automated theorem proving techniques such as tableau or resolution. In order to guarantee the safe-range property of the reformulation, we use a tableau method as in the book by Fitting (1996).

6.1 Tableau-based Method to Compute an Interpolant

In this section we recall in our context the tableau based method to compute an interpolant (Fitting, 1996).

Assume $\phi \to \psi$ is valid, therefore $\phi \wedge \neg\psi$ is unsatisfiable. Then there is a closed tableau corresponding to $\phi \wedge \neg\psi$. In order to compute an interpolant from this tableau one needs to modify it to a *biased tableau.*

Definition 7 (Biased tableau). A biased tableau for formulas $\phi \wedge \neg\psi$ is a tree $T = (V,E)$ where:

- V is a set of nodes, each node is labelled by a set of biased formulas. A *biased formula* is an expression in the form of $L(\varphi)$ or $R(\varphi)$ where φ is a formula. For each node n, $S(n)$ denotes the set of biased formulas labelling n.

- The root of the tree is labelled by $\{L(\phi), R(\neg\psi)\}$
- E is a set of edges. Given 2 nodes n_1 and n_2, $(n_1, n_2) \in E$ iff there is a biased completion rule from n_1 to n_2. We say there is a biased completion rule from n_1 to n_2 if
  - $Y(\mu)$ is the result of applying a rule to $X(\varphi)$, where X and Y refer to L or R (for some rules, there are two possibilities of choosing $Y(\mu)$), and
  - $S(n_2) = (S(n_1) \setminus \{X(\varphi)\}) \cup \{Y(\mu)\}$.

Let C be the set of all constants in the input formulas of the tableau. C^{par} extends C with an infinite set of new constants. A constant is new if it does not occur anywhere in the tableau. With these notations, we have the following rules :

- Propositional rules

Negation rules			$\alpha-$rule	$\beta-$rule
$\dfrac{X(\neg\neg\varphi)}{X(\varphi)}$	$\dfrac{X(\neg\top)}{X(\bot)}$	$\dfrac{X(\neg\bot)}{X(\top)}$	$\dfrac{X(\varphi_1 \wedge \varphi_2)}{X(\varphi_1) \quad X(\varphi_2)}$	$\dfrac{X(\neg(\neg\varphi_1 \wedge \neg\varphi_2))}{X(\varphi_1) \mid X(\varphi_2)}$

- First order rules

$\gamma-$rule	$\sigma-$rule
$\dfrac{X(\forall x.\varphi)}{X(\varphi(t))}$ for any $t \in C^{par}$	$\dfrac{X(\exists x.\varphi)}{X(\varphi(c))}$ for a new constant c

- Equality rules

reflexivity rule	replacement rule
$\dfrac{X(\varphi)}{X(t = t)}$ $t \in C^{par}$ occurs in φ	$\dfrac{X(t = u) \quad Y(\varphi(t))}{Y(\varphi(u))}$

A node in the tableau is *closed* if it contains $X(\varphi)$ and $Y(\neg\varphi)$. If a node is closed, no rule is applied. In the other words, it becomes a leaf of the tree. A branch is closed if it contains a closed node and a tableau is closed if all of its branches are closed. Obviously, if the standard tableau for FOL is closed then so is the biased tableau and vice versa.

Given a closed biased tableau, the interpolant is computed by applying *interpolant rules.* An interpolant rule is written as $S \xrightarrow{int} I$, where I is a formula and $S = \{L(\phi_1), L(\phi_2), ..., L(\phi_n), R(\psi_1), R(\psi_2), ..., R(\psi_m)\}$.

- Rules for closed branches

r1. $S \cup \{L(\varphi), L(\neg\varphi)\} \xrightarrow{int} \bot$ r2. $S \cup \{R(\varphi), R(\neg\varphi)\} \xrightarrow{int} \top$

r3. $S \cup \{L(\bot)\} \xrightarrow{int} \bot$ r4. $S \cup \{R(\bot)\} \xrightarrow{int} \top$

r5. $S \cup \{L(\varphi), R(\neg\varphi)\} \xrightarrow{int} \varphi$ r6. $S \cup \{R(\varphi), L(\neg\varphi)\} \xrightarrow{int} \neg\varphi$

- Rules for propositional cases

p1. $\dfrac{S \cup \{X(\varphi)\} \xrightarrow{int} I}{S \cup \{X(\neg\neg\varphi)\} \xrightarrow{int} I}$ p2. $\dfrac{S \cup \{X(\top)\} \xrightarrow{int} I}{S \cup \{X(\neg\bot)\} \xrightarrow{int} I}$ p3. $\dfrac{S \cup \{X(\bot)\} \xrightarrow{int} I}{S \cup \{X(\neg\top)\} \xrightarrow{int} I}$

p4. $\dfrac{S \cup \{X(\varphi_1), X(\varphi_2)\} \xrightarrow{int} I}{S \cup \{X(\varphi_1 \wedge \varphi_2)\} \xrightarrow{int} I}$ p5. $\dfrac{S \cup \{L(\varphi_1)\} \xrightarrow{int} I_1 \; S \cup \{L(\varphi_2)\} \xrightarrow{int} I_2}{S \cup \{L(\neg(\neg\varphi_1 \wedge \neg\varphi_2))\} \xrightarrow{int} I_1 \vee I_2}$

p6. $\dfrac{S \cup \{R(\varphi_1)\} \xrightarrow{int} I_1 \; S \cup \{R(\varphi_2)\} \xrightarrow{int} I_2}{S \cup \{R(\neg(\neg\varphi_1 \wedge \neg\varphi_2))\} \xrightarrow{int} I_1 \wedge I_2}$

- Rules for first order cases :

f1. $\dfrac{S \cup \{X(\varphi(p))\} \xrightarrow{int} I}{S \cup \{X(\exists x.\varphi(x))\} \xrightarrow{int} I}$ where p is a parameter that does not occur in S or φ

f2. $\dfrac{S \cup \{L(\varphi(c))\} \xrightarrow{int} I}{S \cup \{L(\forall x.\varphi(x))\} \xrightarrow{int} I}$ if c occurs in $\{\varphi_1, ..., \varphi_n\}$

f3. $\dfrac{S \cup \{R(\varphi(c))\} \xrightarrow{int} I}{S \cup \{R(\forall x.\varphi(x))\} \xrightarrow{int} I}$ if c occurs in $\{\psi_1, ..., \psi_m\}$

f4. $\dfrac{S \cup \{L(\varphi(c))\} \xrightarrow{int} I}{S \cup \{L(\forall x.\varphi(x))\} \xrightarrow{int} \forall x.I[c/x]}$ if c does not occur in $\{\varphi_1, ..., \varphi_n\}$

f5. $\dfrac{S \cup \{R(\varphi(c))\} \xrightarrow{int} I}{S \cup \{R(\forall x.\varphi(x))\} \xrightarrow{int} \exists x.I[c/x]}$ if c does not occur in $\{\psi_1, ..., \psi_m\}$

- Rules for equality cases

e1. $\dfrac{S \cup \{X(\varphi(p)), X(t = t)\} \xrightarrow{int} I}{S \cup \{X(\varphi(p))\} \xrightarrow{int} I}$ e2. $\dfrac{S \cup \{X(\varphi(u)), X(t = u)\} \xrightarrow{int} I}{S \cup \{X(\varphi(t)), X(t = u)\} \xrightarrow{int} I}$

e3. $\dfrac{S \cup \{L(\varphi(u)), R(t = u)\} \xrightarrow{int} I}{S \cup \{L(\varphi(t)), R(t = u)\} \xrightarrow{int} t = u \to I}$ if u occurs in $\varphi(t), \psi_1, ..., \psi_m$

e4. $\dfrac{S \cup \{R(\varphi(u)), L(t = u)\} \xrightarrow{int} I}{S \cup \{R(\varphi(t)), L(t = u)\} \xrightarrow{int} t = u \wedge I}$ if u occurs in $\varphi(t), \psi_1, ..., \psi_m$

e5. $\dfrac{S \cup \{L(\varphi(u)), R(t = u)\} \xrightarrow{int} I}{S \cup \{L(\varphi(t)), R(t = u)\} \xrightarrow{int} I[u/t]}$ if u does not occur in $\varphi(t), \psi_1, ..., \psi_m$

e6. $\dfrac{S \cup \{R(\varphi(u)), L(t = u)\} \xrightarrow{int} I}{S \cup \{R(\varphi(t)), L(t = u)\} \xrightarrow{int} I[u/t]}$ if u does not occur in $\varphi(t), \psi_1, ..., \psi_m$

In summary, in order to compute an interpolant of ϕ and ψ, one first need to generate a biased tableaux proof of unsatisfiability of $\phi \wedge \neg\psi$ using biased completion rules and then apply interpolant rules from bottom leaves up to the root.

Let us consider an example to demonstrate how the method works.

Example 6. Let $\mathbb{P} = \{S, G, U\}$, $\mathbb{P}_{\mathcal{DB}} = \{S, U\}$,

$$\begin{aligned}\mathcal{KB} = \{\, & \forall x(S(x) \to (G(x) \vee U(x)))\\ & \forall x(G(x) \to S(x))\\ & \forall x(U(x) \to S(x))\\ & \forall x(G(x) \to \neg U(x))\}\end{aligned}$$

$\mathcal{Q}(x) = G(x)$

Obviously, $\mathcal{Q}$ is implicitly definable from S and U, since the ontology states that G and U partition S. Now we will follow the tableau method to find its exact reformulation. For compactness, we use the notation S^I instead of $S \xrightarrow{int} I$.

$$\begin{aligned}S_0 = \{ & L(\forall x(S(x) \to (G(x) \vee U(x)))),\\ & L(\forall x(G(x) \to S(x))),\\ & L(\forall x(U(x) \to S(x))),\\ & L(\forall x(G(x) \to \neg U(x))),\\ & L(G(c)),\\ & R(\forall x(S(x) \to (G_1(x) \vee U(x)))),\\ & R(\forall x(G_1(x) \to S(x))),\\ & R(\forall x(U(x) \to S(x))),\\ & R(\forall x(G_1(x) \to \neg U(x))),\\ & R(\neg G_1(c))\}\end{aligned}$$

By applying the rule for $\forall$ and removing the implication, we have:

$$\begin{aligned}S_1 = \{ & L(\neg S(c) \vee G(c) \vee U(c)),\\ & L(\neg G(c) \vee S(c))),\\ & L(\neg U(c) \vee S(c)),\\ & L(\neg G(c) \vee \neg U(c)),\\ & L(G(c)),\\ & R(\neg S(c) \vee G_1(c) \vee U(c)),\\ & R(\neg G_1(c) \vee S(c)),\\ & R(\neg U(c) \vee S(c)),\\ & R(\neg G_1(c) \vee \neg U(c)),\\ & R(\neg G_1(c))\}\end{aligned}$$

and the interpolant of S_1 can be computed as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{S_4 \cup \{R(\neg S(c)\}^{S(c)} \quad S_4 \cup \{R(U(c))\}^{\neg U(c)}}{S_4 = S_3 \cup \{R(\neg S(c) \vee U(c))\}^{(S(c) \wedge \neg U(c))}}\vee \quad S_3 \cup \{R(G_1(c))\}^{\top}}{S_3 = S_2 \cup \{L(\neg U(c))\}^{(S(c) \wedge \neg U(c))}}B.7 \quad S_2 \cup \{L(\neg G(c))\}^{\perp}}{S_2 = S_1 \cup \{L(S(c))\}^{(S(c) \wedge \neg U(c))}}B.5 \quad S_1 \cup \{L(\neg G(c))\}^{\perp}}{S_1^{(S(c) \wedge \neg U(c))}}B.3$$

Therefore, $S(c) \wedge \neg U(c)$ is the interpolant and $\widehat{\mathcal{Q}}(x) = S(x) \wedge \neg U(x)$ is an exact reformulation of $\mathcal{Q}(x)$.

Algorithm 1 Safe-range Reformulation

Input: a safe-range $\mathcal{KB}$, a safe-range and implicitly definable query $\mathcal{Q}_{[\mathbb{X}]}$.

Output: an exact safe-range reformulation.

1: Compute the interpolant $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ as in Theorem 7

2: For each free variable x which is not bounded by any positive predicate in $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ do $\widehat{\mathcal{Q}}_{[\mathbb{X}]} := \widehat{\mathcal{Q}}_{[\mathbb{X}]} \wedge \mathit{Adom}_{\widehat{\mathcal{Q}}}(x)$

3: Return $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$

6.2 A Safe-Range Reformulation

Now we want to show that the reformulation computed by the above tableau based method under the condition of Theorem 5 generates a ground safe-range query.

Theorem 8 (**Ground safe-range Reformulation**). *Let $\mathcal{KB}$ be an ontology, and let $\mathcal{Q}$ be a query which is implicitly definable from $\mathbb{P}_{\mathcal{DB}}$. If $\mathcal{KB}$ and $\mathcal{Q}$ are safe-range then a rewritten query $\widehat{\mathcal{Q}}$ obtained using the tableau method described in Section 6.1 is ground safe-range.*

In other words, the conditions of Theorem 8 guarantee that all quantified variables in the reformulation are range-restricted. We need to consider now the still unsafe free variables. The theorem below will help us deal with non-range-restricted free variables. Let us first define the *active domain predicate* of a query $\mathcal{Q}$ as the safe-range formula:

$$\mathit{Adom}_{\mathcal{Q}}(x) := \bigvee_{P\in\mathbb{P}_{\mathcal{Q}}} \big(\exists z_1,\ldots,z_{\mathrm{AR}(P)-1}.\, P(x,z_1,\ldots,z_{\mathrm{AR}(P)-1}) \vee \ldots \vee P(z_1,\ldots,z_{\mathrm{AR}(P)-1},x)\big) \vee \bigvee_{c\in\mathbb{C}_{\mathcal{Q}}} (x=c).$$

Theorem 9 (**Range of the query**). *Let $\mathcal{KB}$ be a domain independent ontology, and let $\mathcal{Q}_{[x_1,\ldots,x_n]}$ be a query which is domain independent with respect to $\mathcal{KB}$. Then*

$$\mathcal{KB} \models \forall x_1,\ldots,x_n.\, \mathcal{Q}_{[x_1,\ldots,x_n]} \rightarrow \mathit{Adom}_{\mathcal{Q}}(x_1) \wedge \ldots \wedge \mathit{Adom}_{\mathcal{Q}}(x_n).$$

Given a safe-range ontology, a safe-range and implicitly definable query is obviously domain independent with respect to the ontology. In this case, Theorem 9 says that the answer of the reformulation can only include active domain elements. Therefore, the active domain predicate can be used as a “guard” for free variables which are not bounded by any positive predicate.

Based on Theorem 8 and Theorem 9, we propose a complete procedure to construct a safe-range reformulation in Algorithm 1.

7. The Guarded Negation Fragment of $\mathcal{ALCHOIQ}$

$\mathcal{ALCHOIQ}$ is an extension of the description logic $\mathcal{ALC}$ with role hierarchies, individuals, inverse roles, and qualified cardinality restrictions: it corresponds to the $\mathcal{SHOIQ}$ description logic without transitive roles; it is the logic at the basis of OWL. The syntax and semantics of $\mathcal{ALCHOIQ}$ concept expressions is summarised in the Figure 1, where A is an atomic concept, C and D are concepts, o is an individual name, P is an atomic role, and R is either P or P^{-}. The *forall* and the qualified and unqualified *atmost* operators can be derived by using negation and the *atleast* operator in the usual way. A TBox in $\mathcal{ALCHOIQ}$

Syntax	Semantics
A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
$\{o\}$	$\{o^{I}\} \subseteq \Delta^{I}$
P	$P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
P^{-}	$\{(y,x)\|(x,y) \in P^{\mathcal{I}}\}$
$\neg C$	$\Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$
$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$\geq nR$	$\{x\|\#(\{y\|(x,y) \in R^{\mathcal{I}}\}) \geq n\}$
$\geq nR.C$	$\{x\|\#(\{y\|(x,y) \in R^{\mathcal{I}}\} \cap C^{\mathcal{I}}) \geq n\}$

Figure 1: Syntax and semantics of $\mathcal{ALCHOIQ}$ concepts and roles

is a set of concept inclusion axioms $C \sqsubseteq D$ and role inclusion axioms $R \sqsubseteq S$ (where C, D are concepts and R, S are roles) with the usual description logics semantics.

In this section, we present an application of Theorem 5, by introducing the $\mathcal{ALCHOIQ}_{GN}$ description logic, the *guarded negation* syntactic fragment of $\mathcal{ALCHOIQ}$ (Figure 2) which happens to express *exactly* the domain independent concepts and TBoxes of $\mathcal{ALCHOIQ}$. The language restricts $\mathcal{ALCHOIQ}$ by just prescribing that negated concepts should be *guarded* by some generalised atom (an atomic concept, a nominal, an unqualified *atleast* number restriction), i.e., absolute negation is forbidden. Similarly, the derived *forall* and *atmost* operators would be guarded by using their standard definition as the dual of the *atleast* operator, but with the guarded negation. $\mathcal{ALCHOIQ}_{GN}$ is actually at the intersection of the GNFO fragment (Bárány, ten Cate, & Otto, 2012) and $\mathcal{ALCHOIQ}$ (see Appendix A.5 for details on GNFO).

$\mathcal{ALCHOIQ}_{GN}$ has the very important property of *coinciding* with the domain independent fragment of $\mathcal{ALCHOIQ}$, therefore providing an excellent candidate language for ontologies and queries satisfying the conditions of Theorem 5.

Theorem 10 (**Expressive power equivalence**). *The domain independent fragment of $\mathcal{ALCHOIQ}$ and $\mathcal{ALCHOIQ}_{GN}$ are equally expressive.*

In other words the theorem says that any domain independent TBox axiom and any domain independent concept query in $\mathcal{ALCHOIQ}$ is logically equivalent, respectively, to a TBox axiom and a concept query in $\mathcal{ALCHOIQ}_{GN}$, and vice-versa. This theorem provides the description logics version of Codd's theorem. Codd's theorem states that the safe-range syntactic fragment of FOL and the domain-independent fragment of FOL are precisely equivalent in expressive power; that is, a database query can be formulated in one language if and only if it can be expressed in the other.

$$
\begin{array}{lcl}
R & ::= & P \mid P^{-} \\
B & ::= & A \mid \{o\} \mid \ \geq nR \\
C & ::= & B \mid \ \geq nR.C \mid \ \geq nR.\neg C \mid B \sqcap \neg C \mid C \sqcap D \mid C \sqcup D
\end{array}
$$

Figure 2: Syntax of $\mathcal{ALCHOIQ}_{GN}$ concepts and roles

7.1 Applying the Constructive Theorem

We want to reformulate concept queries over an ontology with a DBox so that the reformulated query can be evaluated as an SQL query over the database represented by the DBox. In this context, the database is a DBox, the ontology is an $\mathcal{ALCHOIQ}_{GN}$ TBox, and the query is an *$\mathcal{ALCHOIQ}_{GN}$ concept query.* A concept query is either an $\mathcal{ALCHOIQ}_{GN}$ concept expression denoting an open formula with one free variable, or an $\mathcal{ALCHOIQ}_{GN}$ ABox concept assertion denoting a boolean query. As expected, a DBox includes ground atomic statements of the form $A(a)$ and $P(a, b)$ (where A is an atomic concept and P is an atomic role). From Theorem 10 we can draw the following corollary.

Corollary 1. *$\mathcal{ALCHOIQ}_{GN}$ TBoxes and concept queries are domain independent.*

We can also prove the following theorem.

Theorem 11. *$\mathcal{ALCHOIQ}_{GN}$ TBoxes with concept queries have finitely controllable determinacy.*

Therefore, we satisfy the conditions of Theorem 5, with a language which is like the very expressive $\mathcal{ALCHOIQ}$ description logic, but with *guarded* negation.

We argue that non-guarded negation should not appear in a cleanly designed ontology, and, if present, should be fixed. Indeed, the use of *absolute* negative information—such as, e.g., in "a non-`male` is a `female`" ($\neg$ `male` $\sqsubseteq$ `female`)—should be discouraged 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 *guarded* negative information in the subsumee should be allowed—such as in the axiom "a non-`male` `person` is a `female`" (`person` $\sqcap \neg$ `male` $\sqsubseteq$ `female`).

This observation suggests a fix for non-guarded negations: for every non-guarded negation users will be asked to replace it by a guarded one, where the guard may be an arbitrary atomic concept, or nominal, or non-qualified existential. Therefore, the user is asked to make explicit the *type* of that concept, in a way to make it domain independent; note that the type could be also a fresh new atomic concept. We believe that the fix we are proposing for $\mathcal{ALCHOIQ}$ is a reasonable one, and would make all $\mathcal{ALCHOIQ}$ ontologies eligible to be used with our framework.

7.2 A Complete Procedure

$\mathcal{ALCHOIQ}_{GN}$ is a decidable logic and it is a feasible application of our general framework. Given an $\mathcal{ALCHOIQ}_{GN}$ ontology $\mathcal{KB}$ and a concept query $\mathcal{Q}$, we can apply the procedure below to generate a safe-range reformulation over the database concepts and roles (based on the constructive theorem, all the conditions of which are satisfied), if it exists.

Input: An $\mathcal{ALCHOIQ}_{GN}$ TBox $\mathcal{KB}$, a concept query $\mathcal{Q}$ in $\mathcal{ALCHOIQ}_{GN}$, and a database signature (database atomic concepts and roles).

1. Check the implicit definability of the query $\mathcal{Q}$ by testing if $\mathcal{KB} \cup \widetilde{\mathcal{KB}} \models \mathcal{Q} \equiv \widetilde{\mathcal{Q}}$ using a standard OWL2 reasoner ($\mathcal{ALCHOIQ}_{GN}$ is a sublanguage of OWL2). Continue if this holds.

2. Compute a safe-range reformulation $\widehat{\mathcal{Q}}$ from the tableau proof generated in step 1 (see Section 6). This can be implemented as a simple extension of a standard DL reasoner even in the presence of the most important optimisation techniques such as semantic branching, absorption, and backjumping as explained by Seylan et al. (2009) and ten Cate, Franconi, and Seylan (2011).

Output: A safe-range reformulation $\widehat{\mathcal{Q}}$ expressed over the database signature.

Note that the procedure for checking determinacy and computing the reformulation could be run in offline mode at compile time. Indeed, it could be run for each atomic concept in the ontology, and store persistently the outcome for each of them if the reformulation has been successful. This pre-computation may be an expensive operation, since—as we have seen—it is based on entailment, but the complexity involves only the size of the ontology and not of the data.

In order to get an idea about the size of the reformulations, for the $\mathcal{ALCFI}$ description logic there is a tableau-based algorithm computing explicit definitions of at most double exponential size (ten Cate et al., 2011; ten Cate, Franconi, & Seylan, 2013); this algorithm is optimal because it is also shown that the smallest explicit definition of an implicitly defined concept *may* be double exponentially long in the size of the input TBox.

Clearly, similarly to DL-Lite reformulations, more research is needed in order to optimise the reformulation step in order to make it practical. However, note that the framework presented here has a clear advantage from the point of view of *conceptual modelling* since implicit definitions (that is, queries) under general TBoxes can be double exponentially more succinct than acyclic concept definitions (that is, explicit queries over the database).

There is also another interesting open problem about checking that a given database is legal with respect to a given ontology. Remember that a database $\mathcal{DB}$ is legal for an ontology $\mathcal{KB}$ if there exists a model of $\mathcal{KB}$ embedding $\mathcal{DB}$. This check involves heavy computations for which an optimised algorithm is still unknown: as a matter of fact, the only known method today is to reduce the problem to a satisfiability problem where the database is embedded in a TBox using nominals (Franconi et al., 2011). More research is needed in order to optimise the reasoning with nominals in this special case.

Appendix A.5 contains all the definitions and theorems needed to prove theorems 10 and 11.

8. Conclusion

We have introduced a framework to compute the exact reformulation of first-order queries to a database under ontologies. We have found the exact 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 $\mathcal{ALCHOIQ}$ language.

We have also implemented a tool based on the *Prover9* theorem prover (McCune, 2011). Given an arbitrary first-order ontology, a database signature, and an arbitrary first-order query in TPTP syntax, the tool performs all the tests on them to check whether a reformulation can be computed, and it computes an optimal safe-range reformulation.

This framework is useful in data exchange-like scenarios, where the target database (made by determined relations) should be materialised as a proper database, over which arbitrary queries should be performed. This is not achieved in a context with non-exact rewritings preserving the certain answers. In our scenario with description logics ontologies, rewritings of concept queries are pre-computed offline once. We have shown that our framework works in theory also in the case of arbitrary safe-range first-order queries, and our tool shows that this is possible in practice. In the case of description logics, we are working on extending the theoretical framework with conjunctive queries: we need finitely controllable determinacy with conjunctive queries, which seems to follow for some description logic from the works by Bárány, Gottlob, and Otto (2010) and Rosati (2011).

In 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 query 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. With the tool, we plan to evaluate our proposed technique in a real context.

Concurrently, we are exploring the problem of *fixing* real ontologies in order to enforce definability when it is known it should be the case (Franconi, Ngo, & Sherkhonov, 2012c). This happens when it is intuitively obvious that the answer of a query can be found from the available data (that is, the query is definable from the database), but the mediating ontology does not entail the definability. We introduce the novel problem of definability abduction and we solve it completely in the data exchange scenario.

We thank the anonymous reviewers for the very useful comments we got on earlier versions of this paper. We wish to thank Alex Borgida, Tommaso Di Noia, Umberto Straccia, David Toman, and Grant Weddell for the fruitful discussions we had on the topics of this paper.

Appendix A. Proofs

A.1 Proofs of Section 2

Proposition 1

Proof. Let

$$A_{sna} = \{\Theta \mid dom(\Theta) = \mathbb{X},\, rng(\Theta) \subseteq \mathbb{C},\, \forall \mathcal{I} \in I(SNA) \cap M(\mathcal{KB}) \cap E(\mathcal{DB}) \,:\, \mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\Theta]}\}$$

and

$$A_{una} = \{\Theta \mid dom(\Theta) = \mathbb{X},\, rng(\Theta) \subseteq \mathbb{C},\, \forall \mathcal{I} \in I(UNA) \cap M(\mathcal{KB}) \cap E(\mathcal{DB}) \,:\, \mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\Theta]}\}$$

Since SNA is stricter than UNA, i.e. $I(SNA) \subseteq I(UNA)$, we have: $A_{una} \subseteq A_{sna}$ trivially.

Let $\bar{\Theta} \in A_{sna}$. If $\bar{\Theta} \notin A_{una}$ then there is an interpretation $\bar{\mathcal{I}} = \langle \Delta^{\bar{\mathcal{I}}}, \cdot^{\bar{\mathcal{I}}} \rangle$ embedding $\mathcal{DB}$ and satisfying UNA such that $\bar{\mathcal{I}} \in M(\mathcal{KB})$ and $\bar{\mathcal{I}} \not\models \mathcal{Q}_{[\mathbb{X}/\bar{\Theta}]}$. Let us construct new interpretation $\bar{\mathcal{J}} = \langle \Delta^{\bar{\mathcal{J}}}, \cdot^{\bar{\mathcal{J}}} \rangle$ embedding $\mathcal{DB}$ as follows:

- $\Delta^{\bar{\mathcal{J}}} := (\Delta^{\bar{\mathcal{I}}} \setminus \{a^{\bar{\mathcal{I}}} \mid a \in \mathbb{C}\}) \cup \mathbb{C};$

- for each constant $a \in \mathbb{C}$, $a^{\bar{\mathcal{J}}} := a$;
- for every predicate $P \in \mathbb{P}$, $\mathbb{P}^{\bar{\mathcal{J}}}$ is constructed from $\mathbb{P}^{\bar{\mathcal{I}}}$ by replacing of each element $a^{\bar{\mathcal{I}}} \in \mathbb{P}^{\bar{\mathcal{I}}}$, where a is some constant, with a.

Obviously, $\bar{\mathcal{J}}$ satisfies SNA and $\bar{\mathcal{J}}$ and $\bar{\mathcal{I}}$ are isomorphic. Since first-order logic sentences cannot distinguish two isomorphic structures, $\bar{\mathcal{J}} \not\models \mathcal{Q}_{[\mathbb{X}/\bar{\Theta}]}$ which contradicts with the assumption $\bar{\Theta} \in A_{sna}$. Therefore $\bar{\Theta} \in A_{una}$. $\square$

A.2 Proofs of Section 4

Proposition 2.

Proof. Since $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is an exact reformulation of $\mathcal{Q}_{[\mathbb{X}]}$, $\mathcal{KB} \models \forall \mathbb{X}.\mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \widehat{\mathcal{Q}}_{[\mathbb{X}]}$. Then, for any model $\mathcal{I} \in M(\mathcal{KB})$ and for any substitution $\Theta : \mathbb{X} \mapsto \Delta^{\mathcal{I}}$ we have: $\mathcal{I}, \Theta \models \mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \widehat{\mathcal{Q}}_{[\mathbb{X}]}$, which is equivalent to $(\mathcal{I}, \Theta \models \mathcal{Q}_{[\mathbb{X}]} \iff \mathcal{I}, \Theta \models \widehat{\mathcal{Q}}_{[\mathbb{X}]})$.

Now, let $\bar{\Theta}$ be any substitution from $\{\Theta \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) = \mathbb{C},\ \forall \mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB}) : \mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\Theta]}\}$, and $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ be any model of the $\mathcal{KB}$ embedding the $\mathcal{DB}$ (if there are any). Let $\tilde{\Theta} := \cdot^{\mathcal{I}} \circ \bar{\Theta}$—a composition of the substitution $\bar{\Theta}$ and the interpretation function $\cdot^{\mathcal{I}}$ (i.e. $\tilde{\Theta}(x) = a \in \Delta$ iff $\bar{\Theta}(x) = c \in \mathbb{C}$ and $c^{\mathcal{I}} = a$). Then $\mathcal{I}, \tilde{\Theta} \models \mathcal{Q}_{[\mathbb{X}]} \iff \mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\bar{\Theta}]}$ and $\mathcal{I}, \tilde{\Theta} \models \widehat{\mathcal{Q}}_{[\mathbb{X}]} \iff \mathcal{I} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\bar{\Theta}]}$. Summing up: $\mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\bar{\Theta}]} \iff \mathcal{I} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\bar{\Theta}]}$. Hence, $\bar{\Theta} \in \{\Theta \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) = \mathbb{C},\ \forall \mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB}) : \mathcal{I} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\Theta]}\}$. The inverse inclusion can be proved similarly. $\square$

Theorem 3.

Proof. First of all recall that we assume SNA. In order to prove the theorem, one needs the following two propositions.

Proposition 3 (**Domain Independence**). *A query $\mathcal{Q}_{[\mathbb{X}]}$ is domain independent iff for every two interpretations $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ and $\mathcal{J} = \langle \Delta^{\mathcal{J}}, \cdot^{\mathcal{J}} \rangle$ which agree on the interpretation of the predicates from $\mathbb{P}_{\mathcal{Q}}$ (and all constants $\mathbb{C}$), and for every substitution $\Theta : \mathbb{X} \mapsto \Delta^{\mathcal{I}} \cup \Delta^{\mathcal{J}}$ we have:*

$$
\begin{array}{ll}
 & rng(\Theta) \subseteq \Delta^{\mathcal{I}} \ \ \textit{and} \ \ \mathcal{I}, \Theta \models \mathcal{Q}_{[\mathbb{X}]} \\
\textit{iff} & \\
 & rng(\Theta) \subseteq \Delta^{\mathcal{J}} \ \ \textit{and} \ \ \mathcal{J}, \Theta \models \mathcal{Q}_{[\mathbb{X}]}.
\end{array}
$$

Proof. ($\Leftarrow$) Obviously, if the second part of the proposition holds, then the query is domain independent.

($\Rightarrow$) Suppose, the query is domain independent. Let $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ and $\mathcal{J} = \langle \Delta^{\mathcal{J}}, \cdot^{\mathcal{J}} \rangle$ be any two interpretations, which agree on the interpretation of all the predicates from $\mathbb{P}_{\mathcal{Q}}$ (and all constants $\mathbb{C}$), that is $\cdot^{\mathcal{I}|_{\mathbb{P}_{\mathcal{Q}} \cup \mathbb{C}}} = \cdot^{\mathcal{J}|_{\mathbb{P}_{\mathcal{Q}} \cup \mathbb{C}}}$. Let us fix any substitution $\Theta : \mathbb{X} \mapsto \Delta^{\mathcal{I}} \cup \Delta^{\mathcal{J}}$ (if the query is closed, we just omit everything, that concerns a substitution below in the proof) such that:

$$rng(\Theta) \subseteq \Delta^{\mathcal{I}} \ \text{ and } \ \mathcal{I}, \Theta \models \mathcal{Q}_{[\mathbb{X}]}. \tag{2}$$

Let us consider interpretations $\mathcal{I}' = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}'} \rangle$ and $\mathcal{J}' = \langle \Delta^{\mathcal{J}}, \cdot^{\mathcal{J}'} \rangle$, such that $\cdot^{\mathcal{I}|_{\mathbb{P}_{\mathcal{Q}} \cup \mathbb{C}}} = \cdot^{\mathcal{I}'|_{\mathbb{P}_{\mathcal{Q}} \cup \mathbb{C}}} = \cdot^{\mathcal{J}'|_{\mathbb{P}_{\mathcal{Q}} \cup \mathbb{C}}} = \cdot^{\mathcal{J}|_{\mathbb{P}_{\mathcal{Q}} \cup \mathbb{C}}}$, and $\forall P \in \mathbb{P} \setminus \mathbb{P}_{\mathcal{Q}} : P^{\mathcal{I}'} = \emptyset = P^{\mathcal{J}'}$. Let us consider now $\mathcal{I}$ and $\mathcal{I}'$. They have the same domain and interpret all the predicates and constants, occurring in $\mathcal{Q}_{[\mathbb{X}]}$ equally. Therefore, since $\mathcal{I}, \Theta \models \mathcal{Q}_{[\mathbb{X}]}$ (by (2)), $\mathcal{I}', \Theta \models \mathcal{Q}_{[\mathbb{X}]}$.

Let us consider interpretations $\mathcal{I}'$ and $\mathcal{J}'$. By construction, they agree on interpretation of all predicates and constants. Therefore, we can apply the definition of domain independence to them. Then, since

$$rng(\Theta) \subseteq \Delta^{\mathcal{I}} \text{ and } \mathcal{I}', \Theta \models \mathcal{Q}_{[\mathbb{X}]}, \tag{3}$$

we have, that

$$rng(\Theta) \subseteq \Delta^{\mathcal{J}} \text{ and } \mathcal{J}', \Theta \models \mathcal{Q}_{[\mathbb{X}]}. \tag{4}$$

Then again interpretations $\mathcal{J}$ and $\mathcal{J}'$ have the same domain and interpret all the predicates and constants, occurring in $\mathcal{Q}_{[\mathbb{X}]}$ equally. Thus, because of (4),

$$rng(\Theta) \subseteq \Delta^{\mathcal{J}} \text{ and } \mathcal{J}, \Theta \models \mathcal{Q}_{[\mathbb{X}]}. \tag{5}$$

Therefore, (2) $\Longrightarrow$ (5). Similarly (5) $\Longrightarrow$ (2), and the proposition is proved. □

Proposition 4. *If $\mathcal{Q}_{[\mathbb{X}]}$ is domain independent, then for any interpretation $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ and any substitution $\Theta : \mathbb{X} \mapsto \Delta$, such that $\mathcal{I}, \Theta \models \mathcal{Q}_{[\mathbb{X}]}$, the following holds:*

$$rng(\Theta) \subseteq adom(\sigma(\mathcal{Q}_{[\mathbb{X}]}), \mathcal{I}).$$

Proof. Assume, that $\mathbb{X} = \{x\}$, that is $\mathcal{Q}$ has one free variable x (the proof can be easily extended then to the general case).

Let us prove by contradiction. Suppose, there exists a substitution $\{x \to b\}$ such that $\mathcal{I}, \{x \to b\} \models \mathcal{Q}(x)$ and $b \in \Delta \setminus adom(\sigma(\mathcal{Q}(x)), \mathcal{I})$. Let us consider interpretation $\mathcal{I}' = \langle \Delta \cup \{a\}, \cdot^{\mathcal{I}} \rangle$, where a is any brand-new element, that does not appear in Δ. Then $\mathcal{I}', \{x \to b\} \models \mathcal{Q}(x)$ because of domain independence of $\mathcal{Q}(x)$. Consider then another interpretation $\mathcal{I}'' = \langle \Delta \cup \{a\}, \cdot^{\mathcal{I}''} \rangle$ such that occurrence of b in interpretation of any predicate is replaced with the element a. In other words, for any n-ary predicate $P \in \mathbb{P} \setminus \sigma(\mathcal{Q}(x))$, $(\ldots, a, \ldots) \in P^{\mathcal{I}''}$ iff $(\ldots, b, \ldots) \in P^{\mathcal{I}'}$ (since by supposition b does not appear in interpretations of predicates in the query). Interpretations of all the other predicates and all the constants are the same. Then $\mathcal{I}''$ satisfies SNA (even if $b \in \mathbb{C}$). Then, since $\mathcal{I}', \{x \to b\} \models \mathcal{Q}_{[\mathbb{X}]}$, by construction of $\mathcal{I}''$ we have: $\mathcal{I}'', \{x \to a\} \models \mathcal{Q}(x)$, because we changed just interpretations of predicates, that do not appear in the query. Then since $\mathcal{I}'$ and $\mathcal{I}''$ have the same domain and agree on interpretations of all the predicates in $\mathcal{Q}(x)$ and all constants, the following holds: $\mathcal{I}', \{x \to a\} \models \mathcal{Q}(x)$.

Let us now consider interpretations $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ and $\mathcal{I}' = \langle \Delta \cup \{a\}, \cdot^{\mathcal{I}} \rangle$. They have the same interpretation function. Therefore, since $\mathcal{Q}(x)$ is domain independent and $\mathcal{I}', \{x \to a\} \models \mathcal{Q}(x)$, we have: $rng(\{x \to a\}) \subseteq \Delta$. That is $a \in \Delta$. It is a contradiction, because by supposition $a \notin \Delta$. □

Now we prove the theorem itself.

$L := \{\Theta \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) \subseteq \mathbb{C},\ \forall \mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB}) \ : \ \mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\Theta]}\}$;

$R := \{\Theta \mid dom(\Theta) = \mathbb{X},\ rng(\Theta) \subseteq adom(\sigma(\widehat{\mathcal{Q}}), \mathcal{DB}),\ \forall \mathcal{I} = \langle \mathbb{C}, \cdot^{\mathcal{I}}\rangle \in E(\mathcal{DB}) : \mathcal{I}|_{\mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\Theta]}\}$.

Let $\bar{\Theta} \in L$. Then for any $\mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB})$ we have: $\mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\bar{\Theta}]}$ and $\mathcal{I} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\bar{\Theta}]}$, because of Proposition 2.

Consider any $\mathcal{J} = \langle \mathbb{C}, \cdot^{\mathcal{I}}\rangle$ embedding $\mathcal{DB}$. $\mathcal{I}$ and $\mathcal{J}$ agree on interpretations of $\mathbb{C}$ (since we have SNA) and predicates from the set $\sigma(\widehat{\mathcal{Q}}) \cap \mathbb{P}$ which is a subset of $\mathbb{P}_{\mathcal{DB}}$. Then, since $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is domain independent, by Proposition 3 we have: $\mathcal{J} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\bar{\Theta}]}$. Since $\sigma(\widehat{\mathcal{Q}}_{[\mathbb{X}]}) \subseteq \mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}$, $\mathcal{J}|_{P_{\mathcal{DB}} \cup \mathbb{C}} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\bar{\Theta}]}$. Since $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is domain independent, by Proposition 4 we have: $rng(\bar{\Theta}) \subseteq adom(\sigma(\widehat{\mathcal{Q}}_{[\mathbb{X}]}), \mathcal{J})$. $adom(\sigma(\widehat{\mathcal{Q}}_{[\mathbb{X}]}), \mathcal{J}) = adom(\sigma(\widehat{\mathcal{Q}}_{[\mathbb{X}]}), \mathcal{DB})$, because we assume SNA and $\sigma(\widehat{\mathcal{Q}}_{[\mathbb{X}]}) \subseteq \mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}$. Therefore, $rng(\bar{\Theta}) \subseteq adom(\sigma(\widehat{\mathcal{Q}}_{[\mathbb{X}]}), \mathcal{DB})$. Then $\bar{\Theta} \in R$ and, hence, $L \subseteq R$.

Let $\bar{\Theta} \in R$ ($rng(\bar{\Theta}) \subseteq adom(\sigma(\widehat{\mathcal{Q}}_{[\mathbb{X}]}), \mathcal{DB})$). Then for any $\mathcal{J} = \langle \mathbb{C}, \cdot^{\mathcal{I}}\rangle$ embedding $\mathcal{DB}$ we have: $\mathcal{J}|_{P_{\mathcal{DB}} \cup \mathbb{C}} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\bar{\Theta}]}$. Then $\mathcal{J} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\bar{\Theta}]}$. Consider any $\mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB})$. Then $\mathcal{J}$ and $\mathcal{I}$ agree on interpretations of $\mathbb{C}$ (since we have SNA) and $\mathbb{P}_{\mathcal{DB}}$. Since $\sigma(\widehat{\mathcal{Q}}_{[\mathbb{X}/\bar{\Theta}]}) \subseteq \mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}$ and $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is domain independent, by Proposition 3 we have: $\mathcal{I} \models \widehat{\mathcal{Q}}_{[\mathbb{X}/\bar{\Theta}]}$. Since $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is exact reformulation of $\mathcal{Q}_{[\mathbb{X}]}$ under $\mathcal{KB}$ over $\mathbb{P}_{\mathcal{DB}}$, by Proposition 2 we have: $\mathcal{I} \models \mathcal{Q}_{[\mathbb{X}/\bar{\Theta}]}$. Then $\bar{\Theta} \in L$ and, hence, $R \subseteq L$.

Theorem 3 is proved completely. □

A.3 Definitions and Proofs of Section 5

Proposition 5. *Let $\mathcal{KB}$ be a domain independent ontology. If interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\rangle$ is a model of $\mathcal{KB}$, then any $\mathcal{J} = \langle \Delta^{\mathcal{J}}, \cdot^{\mathcal{J}}\rangle$, such that $\cdot^{\mathcal{I}} = \cdot^{\mathcal{J}}$, is also a model of $\mathcal{KB}$.*

Proof. Let α be any sentence from $\mathcal{KB}$. Then, since $\mathcal{I}$ is a model of $\mathcal{KB}$, $\mathcal{I} \models \alpha$. α is domain independent, because $\mathcal{KB}$ is domain independent. Hence, since $\cdot^{\mathcal{I}} = \cdot^{\mathcal{J}}$, $\mathcal{J} \models \alpha$. Thus, $\mathcal{J}$ is a model of any sentence from $\mathcal{KB}$. It means, that $\mathcal{J}$ is a model of $\mathcal{KB}$. □

Proposition 6. *Let $\mathcal{KB}$ be an ontology, and let $\mathcal{Q}_{[\mathbb{X}]}$ be a query which is domain independent with respect to $\mathcal{KB}$. Any exact reformulation of $\mathcal{Q}_{[\mathbb{X}]}$ under $\mathcal{KB}$ (over any set of predicates) is also domain independent with respect to $\mathcal{KB}$.*

Proof. Let $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ be any exact reformulation of $\mathcal{Q}_{[\mathbb{X}]}$ under $\mathcal{KB}$ (over some set of predicates), $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}\rangle$ and $\mathcal{J} = \langle \Delta^{\mathcal{J}}, \cdot^{\mathcal{J}}\rangle$ be any two models of $\mathcal{KB}$ such that $\cdot^{\mathcal{I}} = \cdot^{\mathcal{J}}$, and $\Theta : \mathbb{X} \mapsto \Delta^{\mathcal{I}} \cup \Delta^{\mathcal{J}}$ be any substitution such that

$$rng(\Theta) \subseteq \Delta^{\mathcal{I}} \ \text{ and } \ \mathcal{I}, \Theta \models \widehat{\mathcal{Q}}_{[\mathbb{X}]}.$$

Then, since $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is exact reformulation of $\mathcal{Q}_{[\mathbb{X}]}$, we have: $\mathcal{I}, \Theta \models \mathcal{Q}_{[\mathbb{X}]}$. Then, since $\mathcal{Q}_{[\mathbb{X}]}$ is domain independent with respect to $\mathcal{KB}$, we have:

$$rng(\Theta) \subseteq \Delta^{\mathcal{J}} \ \text{ and } \ \mathcal{J}, \Theta \models \mathcal{Q}_{[\mathbb{X}]}.$$

And again, since $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is exact reformulation of $\mathcal{Q}_{[\mathbb{X}]}$, we have: $\mathcal{J}, \Theta \models \widehat{\mathcal{Q}}_{[\mathbb{X}]}$. Thus, $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is domain independent with respect to $\mathcal{KB}$ by definition. □

Lemma 1. *Let $\mathcal{KB}$ be a domain independent ontology, and let $\mathcal{Q}_{[\mathbb{X}]}$ be a query which is domain independent with respect to $\mathcal{KB}$. Then for any $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ which is a model of $\mathcal{KB}$ and any substitution $\Theta : \mathbb{X} \mapsto \Delta$ such that $\mathcal{I}, \Theta \models \mathcal{Q}_{[\mathbb{X}]}$ the following holds:*

$$rng(\Theta) \subseteq adom(\sigma(\mathcal{Q}_{[\mathbb{X}]}), \mathcal{I}).$$

Proof. Without loss of generality assume, that $\mathbb{X} = \{x\}$, that is $\mathcal{Q}$ has one free variable x (the proof can be easily extended then to the general case).

Let us prove by contradiction. Suppose that $\mathcal{I}, \{x \to b\} \models \mathcal{Q}(x)$, where $b \in \Delta \setminus adom(\sigma(\mathcal{Q}_{[\mathbb{X}]}), \mathcal{I})$. Since $\mathcal{KB}$ is domain independent, for any brand-new element a, that does not appear in Δ, interpretation $\bar{\mathcal{I}} = \langle \Delta \cup \{a\}, \cdot^{\mathcal{I}} \rangle$ is also a model of $\mathcal{KB}$ by Proposition 5. Then, since $\mathcal{Q}(x)$ is domain independent with respect to $\mathcal{KB}$ and $\mathcal{I}$ and $\bar{\mathcal{I}}$ have the same interpretation function, $\bar{\mathcal{I}}, \{x \to b\} \models \mathcal{Q}(x)$.

Consider a new interpretation $\mathcal{I}^1 = \langle \Delta \cup \{a\}, \cdot^{\mathcal{I}^1} \rangle$ constructed from $\bar{\mathcal{I}}$ such that occurrence of b in interpretation of any predicate is replaced by element a. In other words, for any n-ary predicate $P \in \mathbb{P} \setminus \mathbb{P}_{\mathcal{Q}}$, $(\ldots, a, \ldots) \in P^{\mathcal{I}^1}$ iff $(\ldots, b, \ldots) \in P^{\mathcal{I}}$ (since by supposition b does not appear in interpretations of predicates in the query).

Then, since $\bar{\mathcal{I}}, \{x \to b\} \models \mathcal{Q}(x)$ and by construction of $\mathcal{I}^1$ we have: $\mathcal{I}^1, \{x \to a\} \models \mathcal{Q}(x)$ (since we simply replace b, that does not appear neither as a constant in $\mathcal{Q}(x)$ nor in interpretations of predicates in $\mathcal{Q}(x)$, with a). Then, since $\bar{\mathcal{I}}$ and $\mathcal{I}^1$ have the same domain $\Delta \cup \{a\}$ and agree on interpretations of all the predicates from $\mathcal{Q}(x)$ and all the constants (since we assume SNA), we have: $\bar{\mathcal{I}}, \{x \to a\} \models \mathcal{Q}(x)$.

Let us now consider interpretations $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ and $\bar{\mathcal{I}} = \langle \Delta \cup \{a\}, \cdot^{\mathcal{I}} \rangle$. They are both models of $\mathcal{KB}$ and have the same interpretation function $\cdot^{\mathcal{I}}$. So, since $\mathcal{Q}(x)$ is domain independent with respect to $\mathcal{KB}$ and $\bar{\mathcal{I}}, \{x \to a\} \models \mathcal{Q}(x)$, we have: $a \in \Delta$ and $\mathcal{I}, \{x \to a\} \models \mathcal{Q}(x)$ by definition of domain independence with respect to an ontology. It is a contradiction, because by supposition $a \notin \Delta$. The lemma is proved. $\square$

Let α be any set of formulas. Then $Adom_\alpha$ is defined similarly to $Adom_{\mathcal{Q}}$, where Q is a query.

Lemma 2. *Let $\mathcal{KB}$ be a domain independent ontology, and let $\mathcal{Q}_{[\mathbb{X}]}$ ($\mathbb{X} = \{x_1, ..., x_n\}$) be a query which is domain independent with respect to $\mathcal{KB}$. Then the following holds:*

$$\mathcal{KB} \models \forall \mathbb{X}. \mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \mathcal{Q}_{[\mathbb{X}]} | Adom_{\mathcal{KB} \cup \mathcal{Q}}$$

where $\mathcal{Q}_{[\mathbb{X}]} | Adom_{\mathcal{KB} \cup \mathcal{Q}}$ is $\mathcal{Q}'_{[\mathbb{X}]} \wedge Adom_{\mathcal{KB} \cup \mathcal{Q}}(x_1) \wedge ... \wedge Adom_{\mathcal{KB} \cup \mathcal{Q}}(x_n)$, and $\mathcal{Q}'_{[\mathbb{X}]}$ is $\mathcal{Q}_{[\mathbb{X}]}$ such that:

- *Every sub-formula of $\mathcal{Q}_{[\mathbb{X}]}$ in the form of $\exists x. \phi(x)$ is replaced by $\exists x. \phi(x) \wedge Adom_{\mathcal{KB} \cup \mathcal{Q}}(x)$*
- *Every sub-formula of $\mathcal{Q}_{[\mathbb{X}]}$ in the form of $\forall x. \phi(x)$ is replaced by $\forall x. Adom_{\mathcal{KB} \cup \mathcal{Q}}(x) \to \phi(x)$*

Proof. Without loss of generality, we will prove the lemma when $n = 1$. In this case, we write $\mathcal{Q}(x)$ instead of $\mathcal{Q}_{[\mathbb{X}]}$. We prove by contradiction.

Assume there is a model $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ of $\mathcal{KB}$ and an element $a \in \Delta^{\mathcal{I}}$ such that $\mathcal{I}, \{x \to a\} \models \mathcal{Q}(x)$ but $\mathcal{I}, \{x \to a\} \not\models \mathcal{Q}(x) | Adom_{\mathcal{KB} \cup \mathcal{Q}}$.

We construct a new interpretation $\mathcal{J} = \langle Adom^{\mathcal{I}}_{\mathcal{KB}\cup\mathcal{Q}} \cup \mathbb{C}, \cdot^{\mathcal{J}} \rangle$ such that for any predicate $P \in \mathbb{P}_{\mathcal{KB}\cup\mathcal{Q}}$, $P^{\mathcal{J}} := P^{\mathcal{I}}$, and for any predicate $P \in \mathbb{P} \setminus \mathbb{P}_{\mathcal{KB}\cup\mathcal{Q}}$, $P^{\mathcal{J}} := \emptyset$.

Since $\mathcal{KB}$ is domain independent, $\mathcal{J}$ is also a model of $\mathcal{KB}$ by Proposition 5. Then, $\mathcal{J}, \{x \to a\} \models \mathcal{Q}(x)$ because $\mathcal{Q}$ is domain independent with respect to $\mathcal{KB}$. As a consequence, however, $\mathcal{J}, \{x \to a\} \models \mathcal{Q}(x)|Adom_{\mathcal{KB}\cup\mathcal{Q}}$ by the definition of $\mathcal{Q}(x)|Adom_{\mathcal{KB}\cup\mathcal{Q}}$. $\mathcal{Q}(x)|Adom_{\mathcal{KB}\cup\mathcal{Q}}$ is safe-range by construction (see Definition 10). Hence, it is domain independent. Therefore $\mathcal{I}, \{x \to a\} \models \mathcal{Q}(x)|Adom_{\mathcal{KB}\cup\mathcal{Q}}$. Contradiction.

Assume there is a model $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ of $\mathcal{KB}$ and an element $a \in \Delta^{\mathcal{I}}$ such that $\mathcal{I}, \{x \to a\} \models \mathcal{Q}(x)|Adom_{\mathcal{KB}\cup\mathcal{Q}}$ but $\mathcal{I}, \{x \to a\} \not\models \mathcal{Q}(x)$. One can lead to a contradiction similarly as above. Therefore, the lemma is proved. □

Theorem 4.

Proof. The theorem can be proved after Theorem 5.

- The "if" direction. Based on Lemma 2, one can see that exact reformulations of $\mathcal{Q}_{[\mathbb{X}]}$ are also exact reformulations of $\mathcal{Q}_{[\mathbb{X}]}|Adom_{\mathcal{KB}\cup\mathcal{Q}}$. Since $\mathcal{Q}_{[\mathbb{X}]}|Adom_{\mathcal{KB}\cup\mathcal{Q}}$ is safe-range and $\mathcal{KB}$ can always be transformed to a logically equivalent safe-range ontology $\mathcal{KB}'$, obviously the exact safe-range reformulation $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ found in Theorem 5 which takes $\mathcal{KB}'$ and $\mathcal{Q}_{[\mathbb{X}]}|Adom_{\mathcal{KB}\cup\mathcal{Q}}$ as its input is the exact domain independent reformulation of $\mathcal{Q}_{[\mathbb{X}]}$.

- The "only if" direction.

  Suppose, that there exists an exact domain independent reformulation $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ of $\mathcal{Q}_{[\mathbb{X}]}$ over $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$. Then it is domain independent with respect to $\mathcal{KB}$. Hence, by Proposition 6, $\mathcal{Q}_{[\mathbb{X}]}$ is domain independent with respect to $\mathcal{KB}$. Since there exists an exact reformulation of $\mathcal{Q}_{[\mathbb{X}]}$, $\mathcal{Q}_{[\mathbb{X}]}$ is implicitly definable from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$ by the Theorem 1.

The theorem is proved completely. □

In order to help readers follow easier, we recall here formal definitions of safe-range and safe-range normal form (Abiteboul et al., 1995).

Definition 8 (safe-range normal form). denoted by SRNF
A first order formula can be transformed to SRNF by following steps :

- Variable substitution: no distinct pair of quantifiers may employ same variable.
- Remove universal quantifiers
- Remove implications
- Push negation
- Flatten and/or

Definition 9 (**Range restriction of a formula**)**.** denoted by rr
Input : a formula φ in SRNF
Output : a subset of $\textsc{free}(\varphi)$ or $\bot$
Case φ of

- $R(e_1, ..., e_n)$: $rr(\varphi)$ = set of variables in $e_1, ..., e_n$
- $x = a$ or $a = x$, where a is a constant : $rr(\varphi) = \{x\}$
- $x = y$: $rr(\varphi) = \emptyset$
- $\varphi_1 \wedge \varphi_2$: $rr(\varphi) = rr(\varphi_1) \cup rr(\varphi_2)$
- $\varphi_1 \vee \varphi_2$: $rr(\varphi) = rr(\varphi_1) \cap rr(\varphi_2)$
- $\varphi_1 \wedge x = y$: $rr(\varphi) = rr(\varphi_1)$ if $\{x, y\} \cap rr(\varphi_1) = \emptyset$; $rr(\varphi) = rr(\varphi_1) \cup \{x, y\}$ otherwise
- $\neg\varphi_1$: $rr(\varphi) = \emptyset \cap rr(\varphi_1)$
- $\exists x \varphi_1$: $rr(\varphi) = rr(\varphi_1) \backslash \{x\}$ if $x \in rr(\varphi_1)$; $rr(\varphi) = \bot$ otherwise

Note : $\bot \cup Z = \bot \cap Z = \bot \backslash Z = Z \backslash \bot = \bot$

Definition 10 (**safe-range**)**.** A formula φ is safe-range iff $rr(\text{SRNF}(\varphi)) = \textsc{free}(\varphi)$.

Definition 11 (**ground safe-range**)**.** A formula φ is ground safe-range iff after substitution of free variables of φ with constants it becomes safe-range.

Observation 1.

1. For any query $\mathcal{Q}_{[\mathbb{X}]}$ and any interpretation $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ the following holds:

$$Adom_{\mathcal{Q}}^{\mathcal{I}} = adom(\sigma(\mathcal{Q}_{[\mathbb{X}]}), \mathcal{I}).$$

2. $Adom_{\mathcal{Q}}(x)$ is safe-range.

Theorem 5.

Proof. The theorem can be proved after Theorem 8 and Theorem 9.

We will use the following lemma in the proof.

Lemma 3. *If $\mathcal{KB}$ is an ontology, $\mathcal{Q}_{[\mathbb{X}]}$ ($\mathbb{X} = \{x_1, \ldots, x_n\}$) is ground safe-range query and*

$$\mathcal{KB} \models \forall \mathbb{X}.\, \mathcal{Q}_{[\mathbb{X}]} \to \psi_1(x_1) \wedge \ldots \wedge \psi_n(x_n), \tag{6}$$

where $\psi_1, \ldots, \psi_n$ are n safe-range formulas, then the query $\widehat{\mathcal{Q}}_{[\mathbb{X}]} := \mathcal{Q}_{[\mathbb{X}]} \wedge \psi_1(x_1) \wedge \ldots \wedge \psi_n(x_n)$ is safe-range and $\mathcal{KB} \models \forall \mathbb{X}.\, \mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \widehat{\mathcal{Q}}_{[\mathbb{X}]}$.

Proof. Let $\mathcal{Q}'_{[\mathbb{X}]}$ be a safe-range normal form of the query $\mathcal{Q}_{[\mathbb{X}]}$, i.e. $\mathcal{Q}'_{[\mathbb{X}]} := \text{SRNF}(\mathcal{Q}_{[\mathbb{X}]}) = \exists \mathbb{Y}.\, \phi_{[\mathbb{X}\cup\mathbb{Y}]}$, where $\phi_{[\mathbb{X}\cup\mathbb{Y}]}$ is in conjunctive normal form (the safe-range normal form of the query is in prenex normal form). Then $\mathcal{Q}'_{[\mathbb{X}]}$ is ground safe-range, and $\mathcal{KB} \models \mathcal{Q}'_{[\mathbb{X}]} \leftrightarrow \mathcal{Q}_{[\mathbb{X}]}$. Hence, $\mathcal{KB} \models \forall \mathbb{X}.\, \mathcal{Q}'_{[\mathbb{X}]} \to \psi_1(x_1) \wedge \ldots \wedge \psi_n(x_n)$. Let $\mathcal{Q}''_{[\mathbb{X}]} := \mathcal{Q}'_{[\mathbb{X}]} \wedge \psi'_1(x_1) \wedge \ldots \wedge \psi'_n(x_n)$, where each $\psi'_i(x_i) = \text{SRNF}(\psi_i(x_i))$. Then by 6, $\mathcal{KB} \models \mathcal{Q}'_{[\mathbb{X}]} \leftrightarrow \mathcal{Q}''_{[\mathbb{X}]}$. On the other hand $\mathcal{KB} \models \forall \mathbb{X}.\, \widehat{\mathcal{Q}}_{[\mathbb{X}]} \leftrightarrow \mathcal{Q}''_{[\mathbb{X}]}$ by construction. Summing up everything, we have: $\mathcal{KB} \models \widehat{\mathcal{Q}}_{[\mathbb{X}]} \leftrightarrow \mathcal{Q}_{[\mathbb{X}]}$ and the only thing we need to prove is that $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is safe-range.

One can see, that $\mathcal{Q}''_{[\mathbb{X}]} \equiv \exists \mathbb{Y}.\, (\phi_{[\mathbb{X}\cup\mathbb{Y}]} \wedge \psi'_1(x_1) \wedge \ldots \wedge \psi'_n(x_n))$ which is a safe-range normal form of $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$. Since $\mathcal{Q}'_{[\mathbb{X}]} = \exists \mathbb{Y}.\, \phi_{[\mathbb{X}\cup\mathbb{Y}]}$ is ground safe-range, then $rr(\phi_{[\mathbb{X}\cup\mathbb{Y}]}) \setminus \mathbb{X} = \mathbb{Y}' \subseteq \mathbb{Y}$, where for any $y \in \mathbb{Y} \setminus \mathbb{Y}'$ there exists a conjunct $x = y$ in $\phi_{[\mathbb{X}\cup\mathbb{Y}]}$, for some $x \in \mathbb{X}$. Then, since each $\psi'_i(x_i)$ is safe-range, by definition of range restriction $rr(\phi_{[\mathbb{X}\cup\mathbb{Y}]} \wedge \psi'_1(x_1) \wedge \ldots \wedge \psi'_n(x_n)) = \mathbb{X} \cup \mathbb{Y}$, and then $rr(\exists \mathbb{Y}.\, (\phi_{[\mathbb{X}\cup\mathbb{Y}]} \wedge \psi'_1(x_1) \wedge \ldots \wedge \psi'_n(x_n))) = \mathbb{X} = \text{FREE}(\mathcal{Q}''_{[\mathbb{X}]})$. Therefore, $\exists \mathbb{Y}.\, (\phi_{[\mathbb{X}\cup\mathbb{Y}]} \wedge \psi'_1(x_1) \wedge \ldots \wedge \psi'_n(x_n))$ is safe-range by definition, and hence $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is safe-range. $\square$

Let us continue to prove the theorem.

If $\mathbb{X} = \emptyset$, ($\mathcal{Q}$ is closed) then we build an exact safe-range reformulation $\widehat{\mathcal{Q}}$ by using Theorem 8.

Suppose now, $\mathbb{X} = \{x_1, \ldots, x_n\}$. Since $\mathcal{Q}_{[\mathbb{X}]}$ is safe-range and implicitly definable from $\mathbb{P}_{\mathcal{DB}}$, we apply Theorem 8 for $\mathcal{Q}_{[\mathbb{X}]}$ and construct a ground safe-range rewriting $\mathcal{Q}'_{[\mathbb{X}]}$ expressed over $\mathbb{P}_{\mathcal{DB}}$ such that $\mathcal{KB} \models \forall \mathbb{X}.\, \mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \mathcal{Q}'_{[\mathbb{X}]}$. Since $\mathcal{Q}_{[\mathbb{X}]}$ is domain independent (since it is safe-range), it is also domain independent with respect to $\mathcal{KB}$. Hence, by Proposition 6, $\mathcal{Q}'_{[\mathbb{X}]}$ is also domain independent with respect to $\mathcal{KB}$. Moreover, $\mathcal{KB}$ is safe-range and, hence, domain independent. Then by Theorem 9:

$$\mathcal{KB} \models \forall \mathbb{X}.\, \mathcal{Q}'_{[\mathbb{X}]} \to \mathit{Adom}_{\mathcal{Q}}(x_1) \wedge \ldots \wedge \mathit{Adom}_{\mathcal{Q}}(x_n).$$

By the second item of Observation 1 $\mathit{Adom}_{\mathcal{Q}}(x)$ is a safe-range formula. Then by Lemma 3 the query $\widehat{\mathcal{Q}}_{[\mathbb{X}]} := \mathcal{Q}'_{[\mathbb{X}]} \wedge \mathit{Adom}_{\mathcal{Q}'}(x_1) \wedge \ldots \wedge \mathit{Adom}_{\mathcal{Q}'}(x_n)$ is safe-range and $\mathcal{KB} \models \forall \mathbb{X}.\, \mathcal{Q}'_{[\mathbb{X}]} \leftrightarrow \widehat{\mathcal{Q}}_{[\mathbb{X}]}$. Since $\mathcal{KB} \models \forall \mathbb{X}.\, \mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \mathcal{Q}'_{[\mathbb{X}]}$, we have: $\mathcal{KB} \models \forall \mathbb{X}.\, \mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \widehat{\mathcal{Q}}_{[\mathbb{X}]}$. Therefore, the constructed query $\widehat{\mathcal{Q}}_{[\mathbb{X}]}$ is the one we were looking for.

Theorem 5 is proved completely. $\square$

A.4 Proofs of Section 6

Theorem 7.

Proof. First we will prove that if $\mathcal{Q}$ is implicitly definable then the formula (1) is valid. Applying syntactic definition of implicit definability: $\mathcal{KB} \cup \widetilde{\mathcal{KB}} \models \forall \mathbb{X}. \mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \widetilde{\mathcal{Q}_{[\mathbb{X}]}}$. Therefore, when we replace $\mathbb{X}$ by a set of constants $c_1, ..., c_n$, the following formula is valid $(\bigwedge \mathcal{KB} \wedge \bigwedge \widetilde{\mathcal{KB}}) \to (\mathcal{Q}_{[\mathbb{X}/c_1,...,c_n]} \leftrightarrow \widetilde{\mathcal{Q}}_{[\mathbb{X}/c_1,...,c_n]})$. As a consequence, (1) is valid.
Next, we have to prove $\mathcal{KB} \models (\mathcal{Q}_{[\mathbb{X}]} \leftrightarrow \widehat{\mathcal{Q}}_{[c_0,...,c_{n-1}/\mathbb{X}]})$ where $\widehat{\mathcal{Q}}_{[\mathbb{X}/c_1,...,c_n]}$ is a Craig interpolant of (1). Since $\widehat{\mathcal{Q}}_{[\mathbb{X}/c_1,...,c_n]}$ is an interpolant:

1. $((\bigwedge \mathcal{KB}) \land \mathcal{Q}_{[\mathbb{X}/c_1,\ldots,c_n]}) \to \widehat{\mathcal{Q}}_{[\mathbb{X}/c_1,\ldots,c_n]}$
   Then : $\mathcal{KB} \models (\mathcal{Q}_{[\mathbb{X}/c_1,\ldots,c_n]} \to \widehat{\mathcal{Q}}_{[\mathbb{X}/c_1,\ldots,c_n]})$

2. $\widehat{\mathcal{Q}}_{[\mathbb{X}/c_1,\ldots,c_n]} \to ((\bigwedge \widetilde{\mathcal{KB}}) \to \widetilde{\mathcal{Q}}_{[\mathbb{X}/c_1,\ldots,c_n]})$
   Then : $\widetilde{\mathcal{KB}} \models (\widehat{\mathcal{Q}}_{[\mathbb{X}/c_1,\ldots,c_n]} \to \widetilde{\mathcal{Q}}_{[\mathbb{X}/c_1,\ldots,c_n]})$.
   Since $\sigma(\widehat{\mathcal{Q}}) \subseteq \mathcal{P}_{DB}$, the relation $\mathcal{KB} \models (\widehat{\mathcal{Q}}_{[\mathbb{X}/c_1,\ldots,c_n]} \to \mathcal{Q}_{[\mathbb{X}/c_1,\ldots,c_n]})$ holds as well

From(1)(2) we have the expected statement.
Last but not least, since $\sigma(\widehat{\mathcal{Q}}_{[\mathbb{X}/c_1,\ldots,c_n]}) \subseteq \mathbb{P}_{\mathcal{DB}}$ then $\sigma(\widehat{\mathcal{Q}}_{[c_1,\ldots,c_n/\mathbb{X}]}) \subseteq \mathbb{P}_{\mathcal{DB}}$.
With above statements, $\widehat{\mathcal{Q}}_{[c_1,\ldots,c_n/\mathbb{X}]}$ is really an explicit definition of $\mathcal{Q}$ □

Theorem 8.

Proof. We need the following propositions to prove the theorem.

Proposition 7. $\varphi_1 \land \varphi_2$ *is safe-range and closed iff* φ_1 *and* φ_2 *are safe-range and closed.*

Proof. We have:

- $rr(\varphi_1 \land \varphi_2) = rr(\varphi_1) \cup rr(\varphi_2)$
- $\textsc{free}(\varphi_1 \land \varphi_2) = \textsc{free}(\varphi_1) \cup \textsc{free}(\varphi_2)$
- $rr(\varphi_1) = \bot$ or $rr(\varphi_1) \subseteq \textsc{free}(\varphi_1)$
- $rr(\varphi_2) = \bot$ or $rr(\varphi_2) \subseteq \textsc{free}(\varphi_2)$
- $\varphi_1 \land \varphi_2$ is closed iff $free(\varphi_1) = \textsc{free}(\varphi_2) = \emptyset$
- φ_1 is closed iff $\textsc{free}(\varphi_1) = \emptyset$
- φ_2 is closed iff $\textsc{free}(\varphi_2) = \emptyset$
- $\varphi_1 \land \varphi_2$ is safe-range iff $rr(\varphi_1 \land \varphi_2) = \textsc{free}(\varphi_1 \land \varphi_2)$
- φ_1 is safe-range iff $rr(\varphi_1) = \textsc{free}(\varphi_1)$
- φ_1 is safe-range iff $rr(\varphi_2) = \textsc{free}(\varphi_2)$

Therefore:

- $\varphi_1 \land \varphi_2$ is closed iff φ_1 and φ_2 are closed
- $\varphi_1 \land \varphi_2$ is closed, safe-range iff φ_1 and φ_2 are closed, safe-range.

□

Proposition 8. $\varphi_1 \lor \varphi_2$ *is safe-range and closed iff* φ_1 *and* φ_2 *are safe-range and closed.*

Proof. We have:

- $rr(\varphi_1 \lor \varphi_2) = rr(\varphi_1) \cap rr(\varphi_2)$

- $\text{FREE}(\varphi_1 \vee \varphi_2) = \text{FREE}(\varphi_1) \cup \text{FREE}(\varphi_2)$
- $rr(\varphi_1) = \bot$ or $rr(\varphi_1) \subseteq \text{FREE}(\varphi_1)$
- $rr(\varphi_2) = \bot$ or $rr(\varphi_2) \subseteq \text{FREE}(\varphi_2)$
- $\varphi_1 \vee \varphi_2$ is closed iff $\text{FREE}(\varphi_1) = \text{FREE}(\varphi_2) = \emptyset$
- φ_1 is closed iff $\text{FREE}(\varphi_1) = \emptyset$
- φ_2 is closed iff $\text{FREE}(\varphi_2) = \emptyset$
- $\varphi_1 \vee \varphi_2$ is safe-range iff $rr(\varphi_1 \vee \varphi_2) = \text{FREE}(\varphi_1 \vee \varphi_2)$
- φ_1 is safe-range iff $rr(\varphi_1) = \text{FREE}(\varphi_1)$
- φ_1 is safe-range iff $rr(\varphi_2) = \text{FREE}(\varphi_2)$

Therefore:

- $\varphi_1 \vee \varphi_2$ is closed iff φ_1 and φ_2 are closed
- $\varphi_1 \vee \varphi_2$ is closed, safe-range iff φ_1 and φ_2 are closed, safe-range.

$\square$

Proposition 9. $\forall \vec{x} \varphi(\vec{x})$ *is closed and safe-range then* $\varphi(\vec{t})$ *is closed and safe-range where* $\vec{t}$ *are constants.*

Proof. Obviously, if $\forall \vec{x} \varphi(\vec{x})$ is closed then $\varphi(\vec{t})$ is closed.
Assume that $\varphi(\vec{t})$ is not safe-range. Since it's closed $rr(\text{SRNF}(\varphi(\vec{t}))) = \bot$
$\Rightarrow$ $\text{SRNF}(\varphi(\vec{t}))$ must contain a subformula which is in the form $\exists \vec{z} \varphi'(\vec{t}, \vec{z})$
where $\vec{z} \not\subseteq rr(\text{SRNF}(\varphi'(\vec{t}, \vec{z})))$
$\Rightarrow$ $\text{SRNF}(\varphi(\vec{x}))$ must contain a subformula which is in the form $\exists \vec{z} \varphi'(\vec{x}, \vec{z})$
where $\vec{z} \not\subseteq rr(\text{SRNF}(\varphi'(\vec{x}, \vec{z})))$
$\Rightarrow$ $\text{SRNF}(\neg\varphi(\vec{x}))$ must contain a subformula which is in the form $\exists \vec{z} \varphi'(\vec{x}, \vec{z})$
where $\vec{z} \not\subseteq rr(\text{SRNF}(\varphi'(\vec{x}, \vec{z})))$ because pushing negation does not effect the formula under $\exists$
$\Rightarrow rr(\text{SRNF}(\neg\varphi(\vec{x}))) = \bot$
$\Rightarrow rr(\text{SRNF}(\neg\exists\vec{x}\neg\varphi(\vec{x}))) = \bot$
$\Rightarrow rr(\text{SRNF}(\forall\vec{x}\varphi(\vec{x}))) = \bot$
$\forall \vec{x} \varphi(\vec{x})$ is not safe-range
$\Rightarrow$ *contradiction.* $\square$

Proposition 10. $\exists \vec{x} \varphi(\vec{x})$ *is closed and safe-range then* $\varphi(\vec{t})$ *is closed and safe-range where* $\vec{t}$ *are constants.*

Proof. Undoubtedly, if $\exists \vec{x} \varphi(\vec{x})$ is closed then $\varphi(\vec{t})$ is closed.
Assume that $\varphi(\vec{t})$ is not safe-range. Since it is closed, $rr(\text{SRNF}(\varphi(\vec{t}))) = \bot$
$\Rightarrow$ $\text{SRNF}(\varphi(\vec{t}))$ must contain a subformula which is in the form $\exists \vec{z} \varphi'(\vec{t}, \vec{z})$
where $\vec{z} \not\subseteq rr(\text{SRNF}(\varphi'(\vec{t}, \vec{z})))$

$\Rightarrow$ SRNF($\varphi(\vec{x})$) must contain a subformula which is in the form $\exists\vec{z}\varphi'(\vec{x},\vec{z})$
where $\vec{z} \not\subseteq rr(\text{SRNF}(\varphi'(\vec{x},\vec{z})))$
$\Rightarrow rr(\text{SRNF}(\varphi(\vec{x}))) = \bot$
$\Rightarrow rr(\text{SRNF}(\exists\vec{x}\varphi(\vec{x}))) = \bot$
$\Rightarrow \exists\vec{x}\varphi(\vec{x})$ is not safe-range
$\Rightarrow$ *contradiction.* ☐

Based on these propositions, we prove Theorem 8 as follows.

First, we will show that if ϕ and ψ are closed and safe-range and $\phi \to \psi$ is valid then so is their interpolant. Assume T is a biased tableau of of $\phi \wedge \neg\psi$. Therefore the root node of T is $S = \{L(\phi), R(\neg\psi)\}$. Based on all the tableau expansion rules and above propositions, at every expansion step where $S = \{L(\varphi_1), ..., L(\varphi_n), R(\psi_1), ..., R(\psi_m)\}$, $\varphi_1, ..., \varphi_n$ and $\psi_1, ..., \psi_m$ are safe-range and closed(*) .
Now we need to prove that the interpolant at each step is safe-range and closed (**) by induction on the shape of proof and the set of rules in Section 6.

- Rules for closed branches: It's trivial because φ and $\neg\varphi$ are safe-range and closed because of (*)

- Rules for propositional case :
  For the rule (p1)(p2)(p3)(p4) nothing changes, so one does not need to prove.
  For the rule (p5), apply the Proposition 8, (**) holds.
  For the rule (p6), apply the Proposition 7,(**) holds.

- Rules for first order case :
  For the rule (f1) (f2) (f3) nothing changes, so one does not need to prove.
  For the rule (f4), since c does not occur in $\{\varphi_1, ..., \varphi_n\}$ then the only case to have c in I is that S contains $R(\neg\varphi(c))$. Therefore $S \cup \{L(\varphi(c))\} \overset{int}{\to} I = \varphi(c)$. Since $\forall x.\varphi(x)$ is safe-range (due to (*)) then $\forall x.I[c/x]$ is safe-range too
  For the rule (f5), since c does not occur in $\{\psi_1, ..., \psi_m\}$ then the only case to have c in I is that S contains $L(\neg\varphi(c))$. Therefore $S \cup \{R(\varphi(c))\} \overset{int}{\to} I = \neg\varphi(c)$. Since $\forall x.\varphi(x)$ is safe-range (due to (*)) then $\exists x.\neg I[c/x]$ is safe-range too

- Rules for equality : Because all the input formulas are closed and do not contain function symbols, all equations are ground. Therefore, they do not influence the safe-range property of interpolant in each step.

As a consequence, because $\mathcal{Q}(\vec{c})$, $\mathcal{KB}$, $\mathcal{KB}'$,$\neg\mathcal{Q}'(\vec{c})$ are closed and safe-range then so is the interpolant $\widehat{\mathcal{Q}}(\vec{c})$ of $\mathcal{KB} \wedge \mathcal{Q}(\vec{c})$ and $\mathcal{KB}' \to \mathcal{Q}'(\vec{c})$. ☐

Theorem 9.

Proof. As a consequence of Lemma 1, Theorem 9 holds. ☐

A.5 Definitions and Proofs of Section 7

The safe-range fragment of $\mathcal{ALCHOIQ}$. We call any axiom (concept) in $\mathcal{ALCHOIQ}$ (ground) safe-range, if the corresponding logically equivalent (open) formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ is (ground) safe-range. For any concept C we denote the corresponding logically equivalent formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ with one free variable x as $C(x)$. Unfortunately concept inclusion axioms in $\mathcal{ALCHOIQ}$ ontologies may not be safe-range: for example, the axiom $\neg$ `male` $\sqsubseteq$ `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 $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ (which is actually in a safe-range normal form). The following proposition provides recursive rules deciding whether an $\mathcal{ALCHOIQ}$ concept is safe-range.

Proposition 11. *Let A be an atomic concept, let C and D be $\mathcal{ALCHOIQ}$ concepts, and let R be either an atomic role or an inverse atomic role. Then:*

1. *$A, \{o\}, \geq nR, \geq nR.C$ are safe-range;*
2. *$C \sqcap D$ is safe-range if and only if C is safe-range or D is safe-range;*
3. *$C \sqcup D$ is safe-range if and only if C is safe-range and D is safe-range;*
4. *$\neg C$ is safe-range if and only if C is not safe-range.*

Proof. It is enough to prove the proposition just for atomic roles because the order of variables in binary atoms of a first-order logic translation of an $\mathcal{ALCHOIQ}$ concept does not affect the safe-range property of the translation. Therefore hereafter we assume that R is an atomic role.

- Since A is an atomic concept, $A(x)$ is safe-range.
- $\{o\}(x) = (x = o)$ - safe-range.
- $(\geq nR)(x) = \exists x_1, \ldots, x_n.\ R(x, x_1) \wedge \ldots \wedge R(x, x_n) \wedge (x_1 \neq x_2) \wedge \ldots \wedge (x_{n-1} \neq x_n)$ - safe-range.
- $(\geq nR.C)(x) = \exists x_1, \ldots, x_n.\ R(x, x_1) \wedge \ldots \wedge R(x, x_n) \wedge C(x_1) \wedge \ldots \wedge C(x_n) \wedge (x_1 \neq x_2) \wedge \ldots \wedge (x_{n-1} \neq x_n)$ - safe-range.
- Let us prove, that $(C \sqcap D)(x) = C(x) \wedge D(x)$ is safe-range if and only if $C(x)$ is safe-range or $D(x)$ is safe-range.

  $\Leftarrow$) Let $C(x)$ or $D(x)$ be safe-range and let both of them be in safe-range normal forms. Then
  $C(x) \wedge D(x)$ is safe-range by definition.

  $\Rightarrow$) Let $C(x) \wedge D(x)$ be safe-range and in safe-range normal form (i.e. both $C(x)$ and $D(x)$ are in safe-range normal form). Let us prove by contradiction. Suppose, both $C(x)$ and $D(x)$ are not safe-range. Then $C(x) \wedge D(x)$ is not safe-range by definition. It is a contradiction. Therefore, $C(x)$ is safe-range or $D(x)$ is safe-range.

- Let us prove, that $(C \sqcup D)(x) = C(x) \vee D(x)$ is safe-range if and only if $C(x)$ is safe-range and $D(x)$ is safe-range.

  $\Leftarrow$) Let $C(x)$ and $D(x)$ be both safe-range and in safe-range normal forms. Then $C(x) \vee D(x)$ is safe-range by definition.

  $\Rightarrow$) Let $C(x) \vee D(x)$ be safe-range and in safe-range normal form (i.e. both $C(x)$ and $D(x)$ are in safe-range normal form). Let us prove by contradiction. Suppose, $C(x)$ or $D(x)$ is not safe-range. Then $C(x) \vee D(x)$ is not safe-range by definition. It is a contradiction. Therefore, $C(x)$ is safe-range and $D(x)$ is safe-range.

- Let us prove, that $\neg C(x)$ is safe-range if and only if $C(x)$ is not safe-range.

  $\Rightarrow$) Let $\neg C(x)$ be safe-range. Let us prove by contradiction. Let $C(x)$ be also safe-range. Then both $\neg C(x)$ and $C(x)$ are domain independent. But one can easily see (looking at the definition of domain independence), that it is impossible. Therefore, $C(x)$ is not safe-range. $\Leftarrow$) We need to prove, that if $C(x)$ is not safe-range, then $\neg C(x)$ is safe-range.

  Let us prove by induction on structure of the formula. Suppose, the item is true for any subformula of the formula $C(x)$.

  Suppose, $C(x)$ is not safe-range. Let us consider (using already proved items) all the possible cases, when $C(x)$ is not safe-range.

  - $C(x) = (\forall R.D)(x) = \forall y.\, R(x,y) \rightarrow D(y) \equiv \neg\exists y.R(x,y) \wedge \neg D(y)$ - not safe-range, where D is any (possibly complex) concept. Then $\neg C(x) = \exists y.R(x,y) \wedge \neg D(y)$ is safe-range by definition.
  - Suppose, $C(x) = (D \sqcap F)(x)$ is not safe-range. Then $D(x)$ is not safe-range and $F(x)$ is not safe-range. Since both $D(x)$ and $F(x)$ are subformulas of $C(x)$, by applying the current item we get: $\neg D(x)$ and $\neg F(x)$ are safe-range. $\neg C(x) \equiv \neg(D(x) \wedge F(x)) \equiv \neg D(x) \vee \neg F(x)$ - safe-range, because $\neg D(x)$ and $\neg F(x)$ are safe-range.
  - Suppose, $C(x) = (D \sqcup F)(x)$ is not safe-range. Then $D(x)$ is not safe-range or $F(x)$ is not safe-range. Since both $D(x)$ and $F(x)$ are subformulas of $C(x)$, by applying the current item we get: either $\neg D(x)$ or $\neg F(x)$ is safe-range. $\neg C(x) \equiv \neg(D(x) \vee F(x)) \equiv \neg D(x) \wedge \neg F(x)$ - safe-range, because either $\neg D(x)$ or $\neg F(x)$ is safe-range.
  - Suppose, $C(x) = \neg D(x)$ is not safe-range. We need to prove, that $\neg C(x) \equiv D(x)$ is safe-range. Let us prove by contradiction. Suppose, $D(x)$ is not safe-range. Then, since $D(x)$ is a subformula of $C(x)$, by applying the current item we get: $\neg D(x) \equiv C(x)$ is safe-range. It is a contradiction. Hence, $\neg C(x)$ is safe-range.

  The item is proved completely.

The proposition is proved completely. $\square$

Proposition 12. *All $\mathcal{ALCHOIQ}$ role inclusion axioms are safe-range.*

Proof. Let $S \sqsubseteq R$ be any role inclusion axiom in $\mathcal{ALCHOIQ}$. The formula $\neg\exists x, y.\, S(x, y)^* \wedge \neg R(x, y)^*$ is a first-order logic translation of the axiom, where $(x, y)^*$ stands for (x, y) if the preceding role is atomic and $(x, y)^*$ stands for (y, x) if the preceding role is inverse atomic. This formula is safe-range. $\square$

Guarded negation first-order logic. We recall the definition of *guarded negation first-order logic (GNFO)* given in the paper by Bárány et al. (2012). GNFO is a fragment of first-order logic consisting of all formulas generated by the following recursive definition:

$$\phi ::= \ R(t_1, \ldots, t_n) \mid t_1 = t_2 \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \exists x.\, \phi \mid \alpha \wedge \neg\phi \tag{7}$$

where each t_i is either a variable or a constant, α in $\alpha \wedge \neg\phi$ is an atomic formula (possibly an equality statement) containing all free variables of ϕ.

Guarded negation fragment of $\mathcal{ALCHOIQ}$. Now we consider $\mathcal{ALCHOIQ}_{GN}$ - a guarded negation fragment of $\mathcal{ALCHOIQ}$ (i.e. an intersection of GNFO and $\mathcal{ALCHOIQ}$). We say, that

- a concept C is an $\mathcal{ALCHOIQ}_{GN}$ concept if C is an $\mathcal{ALCHOIQ}$ concept and the corresponding first-order logic translation $C(x)$ is expressed in GNFO;
- a concept inclusion axiom $C \sqsubseteq D$ is an $\mathcal{ALCHOIQ}_{GN}$ concept inclusion axiom if C and D are $\mathcal{ALCHOIQ}$ concepts and the formula $\neg\exists x.\, C(x) \wedge \neg D(x)$ (which is equivalent to the first-order translation of $C \sqsubseteq D$) is expressed in GNFO;
- a role inclusion axiom $S \sqsubseteq R$ is an $\mathcal{ALCHOIQ}_{GN}$ role inclusion axiom if S and R are roles (atomic or inverse atomic) and the formula $\neg\exists x, y.\, S(x, y)^* \wedge \neg R(x, y)^*$, where $(x, y)^*$ stands for (x, y) if the preceding role is atomic and $(x, y)^*$ stands for (y, x) if the preceding role is inverse atomic, is expressed in GNFO.

It is easy to see, that any $\mathcal{ALCHOIQ}$ role inclusion axiom is an $\mathcal{ALCHOIQ}_{GN}$ role inclusion axiom. Then because of Proposition 12 the following holds.

Proposition 13.
- *All $\mathcal{ALCHOIQ}_{GN}$ role inclusion axioms are safe-range.*
- *All safe-range role inclusion axioms in $\mathcal{ALCHOIQ}$ are in $\mathcal{ALCHOIQ}_{GN}$.*

From the definition of GNFO and $\mathcal{ALCHOIQ}$ it follows, that the complex concept C of the logic $\mathcal{ALCHOIQ}_{GN}$ is recursively defined as follows:

$$B ::= \ A \mid \{o\} \mid \ \geq nR$$

$$C ::= \ B \mid \ \geq nR.C \mid \ \geq nR.\neg C \mid B \sqcap \neg C \mid C \sqcap D \mid C \sqcup D \tag{8}$$

where A is an atomic concept, R is an atomic role or an inverse atomic role, and C and D are $\mathcal{ALCHOIQ}_{GN}$ concepts (possibly complex).

Note, that in general, according to the definition (7) of GNFO all formulas with *atleast* operator for $n \geq 2$ are not in GNFO because of non-guarded inequality statements $x_i \neq x_j$. We fix this by assuming that inequality relation is actually a special binary database predicate. This assumption is usual for databases.

Also strictly speaking $\geq nR \sqcap \neg C$ is not in GNFO. Indeed, the formula $(\exists x_1, \dots, x_n. R(x, x_1) \wedge \dots \wedge R(x, x_n) \wedge (x_1 \neq x_2) \wedge \dots \wedge (x_{n-1} \neq x_n)) \wedge \neg C(x)$ is not in GNFO ($R(x, y)$ here stands for $P(x, y)$ if R stands for an atomic role P, and $R(x, y)$ stands for $P(y, x)$ if R stands for an inverse atomic role P^-), but it can be easily transformed to a logically equivalent GNFO one by simply shifting the parentheses: $\exists x_1, \dots, x_n. (R(x, x_1) \wedge \dots \wedge R(x, x_n) \wedge (x_1 \neq x_2) \wedge \dots \wedge (x_{n-1} \neq x_n) \wedge \neg C(x))$. So, we can assume, that the formula $\geq nR \sqcap \neg C$ is in $\mathcal{ALCHOIQ}_{GN}$.

Proposition 14. *All $\mathcal{ALCHOIQ}_{GN}$ concepts are safe-range.*

Proof. Let us prove by induction on the structure of $\mathcal{ALCHOIQ}_{GN}$ concepts defined by (8).

1. $A, \{o\}, \geq nR, \geq nR.C, \geq nR.\neg C$ (C is an $\mathcal{ALCHOIQ}_{GN}$ concept) are safe-range because of the item 1 of Proposition 11.
2. For any atomic concept A, any individual o any role R and any natural number n the concepts $A \sqcap \neg C$, $\{o\} \sqcap \neg C$ and $\geq nR \sqcap \neg C$ are safe-range because of the item 3 of Proposition 11 and since A, $\{o\}$ and $\geq nR$ are safe-range by the first item.
3. Suppose, that $\mathcal{ALCHOIQ}_{GN}$ concepts C and D are safe-range. Then the concepts $C \sqcap D$ and $C \sqcup D$ are safe-range by the items 2 and 3 of Proposition 11 respectively.

The proposition is proved. ☐

Lemma 4. *For any safe-range concept C in $\mathcal{ALCHOIQ}$ the following holds:*

$$C \sqsubseteq B_1 \sqcup \dots \sqcup B_n,$$

where B_i appears as a subconcept in C and is one of the following concepts:

- *an atomic concept A;*
- *$\{o\}$, where o is an individual name;*
- *$\geq nR$, where R is an atomic role or inverse atomic role, n is a natural number.*

Proof. Let us prove the proposition by induction for all safe-range concepts of $\mathcal{ALCHOIQ}$.

- $A, \{o\}, \geq nR, \geq nR.C$ are safe-range by Proposition 11. $A \sqsubseteq A$, $\{o\} \sqsubseteq \{o\}$, $\geq nR \sqsubseteq \geq nR$, $\geq nR.C \sqsubseteq \geq nR$.

Suppose now that C is a complex safe-range concept and the proposition holds for all safe-range subconcepts of C.

1. $C = C_1 \sqcap C_2$ - safe-range. Then either C_1 or C_2 is safe-range. Let C_1 be safe-range. Hence, $C_1 \sqsubseteq B_1 \sqcup \dots \sqcup B_m$, where B_i is a concept of the aforementioned type. Then $C_1 \sqcap C_2 \sqsubseteq C_1 \sqsubseteq B_1 \sqcup \dots \sqcup B_m$.
2. $C = C_1 \sqcup C_2$ - safe-range. Then C_1 and C_2 are safe-range. Hence, $C_1 \sqsubseteq B_1 \sqcup \dots \sqcup B_k$ and $C_2 \sqsubseteq B_{k+1} \sqcup \dots \sqcup B_m$, where B_i is a concept of the aforementioned type. Then $C_1 \sqcup C_2 \sqsubseteq (B_1 \sqcup \dots \sqcup B_k) \sqcup (B_{k+1} \sqcup \dots \sqcup B_m) \sqsubseteq B_1 \sqcup \dots \sqcup B_m$.

3. $C = \neg D$ is safe-range. By Proposition 11 it is possible if and only if D is not safe-range. That is one of the following cases takes place.

   - $D = D_1 \sqcap D_2$. Then $\neg D \equiv \neg D_1 \sqcup \neg D_2$. And we reduced this case to the item 2.
   - $D = D_1 \sqcup D_2$. Then $\neg D \equiv \neg D_1 \sqcap \neg D_2$. And we reduced this case to the item 1.
   - $D = \neg D_1$. Then $\neg D \equiv \neg\neg D_1 \equiv D_1$. Hence, D_1 is a safe-range subconcept of $\neg D$. Then the proposition holds for D_1 and, hence, also for C, because $C \equiv \neg D \equiv D_1$.

The lemma is proved completely. □

Lemma 5. *For any $\mathcal{ALCHOIQ}$ concept C there exists an $\mathcal{ALCHOIQ}_{GN}$ concept C' such that either $C \equiv C'$ or $C \equiv \neg C'$.*

Proof. Suppose that the lemma holds for all $\mathcal{ALCHOIQ}$ subconcepts of the $\mathcal{ALCHOIQ}$ concept C. Let us prove it for C.

1. Base. $A, \{o\}, \geq nR$ are $\mathcal{ALCHOIQ}_{GN}$ concepts by the definition of $\mathcal{ALCHOIQ}_{GN}$ concept (8).

2. $C = \geq nR.D$ and D' is an $\mathcal{ALCHOIQ}_{GN}$ concept such that $D \equiv D'$ or $D \equiv \neg D'$. Then $C \equiv \geq nR.D'$ or $C \equiv \geq nR.\neg D'$. $\geq nR.D'$ and $\geq nR.\neg D'$ are both $\mathcal{ALCHOIQ}_{GN}$ concepts. Hence, the item is proved.

3. $C = \neg D$ and D' is an $\mathcal{ALCHOIQ}_{GN}$ concept such that $D \equiv D'$ or $D \equiv \neg D'$. Then $C \equiv \neg D'$ or $C \equiv \neg\neg D' \equiv D'$. The item is proved.

4. $C = C_1 \sqcap C_2$ and C'_1 is an $\mathcal{ALCHOIQ}_{GN}$ concept such that $C_1 \equiv C'_1$ or $C_1 \equiv \neg C'_1$, C'_2 is an $\mathcal{ALCHOIQ}_{GN}$ concept such that $C_2 \equiv C'_2$ or $C_2 \equiv \neg C'_2$. Consider all possible cases.

   (a) $C_1 \equiv C'_1$ and $C_2 \equiv C'_2$. Then $C \equiv C'$, where $C' = C'_1 \sqcap C'_2$ is an $\mathcal{ALCHOIQ}_{GN}$ concept (because C'_1 and C'_2 are $\mathcal{ALCHOIQ}_{GN}$ concepts).

   (b) $C_1 \equiv \neg C'_1$ and $C_2 \equiv \neg C'_2$. Then $C \equiv \neg C'_1 \sqcap \neg C'_2 \equiv \neg(C'_1 \sqcup C'_2) = \neg C'$, where $C' = C'_1 \sqcup C'_2$ is an $\mathcal{ALCHOIQ}_{GN}$ concept (because C'_1 and C'_2 are $\mathcal{ALCHOIQ}_{GN}$ concepts).

   (c) $C_1 \equiv C'_1$ and $C_2 \equiv \neg C'_2$ (the case when $C_1 \equiv \neg C'_1$ and $C_2 \equiv C'_2$ is the similar one). Then $C \equiv C'_1 \sqcap \neg C'_2$. Since C'_1 is an $\mathcal{ALCHOIQ}_{GN}$ concept by Proposition 14 it is safe-range and, hence, by Lemma 4 $C'_1 \sqsubseteq B_1 \sqcup \ldots \sqcup B_n$, where each B_i is either an atomic concept A or $\{o\}$ or $\exists R$. Then $C'_1 \equiv C'_1 \sqcap (B_1 \sqcup \ldots \sqcup B_n)$ and, hence, $C \equiv C'_1 \sqcap (B_1 \sqcup \ldots \sqcup B_n) \sqcap \neg C'_2 \equiv C'_1 \sqcap (B_1 \sqcap \neg C'_2 \sqcup \ldots \sqcup B_n \sqcap \neg C'_2)$. Each disjunct $B_i \sqcap \neg C'_2$ is an $\mathcal{ALCHOIQ}_{GN}$ concept (because C_2 is $\mathcal{ALCHOIQ}_{GN}$ concept and by the definition (8) of $\mathcal{ALCHOIQ}_{GN}$ concepts). Then $C' = C'_1 \sqcap (B_1 \sqcap \neg C'_2 \sqcup \ldots \sqcup B_n \sqcap \neg C'_2)$ is an $\mathcal{ALCHOIQ}_{GN}$ concept. $C \equiv C'$. The item is proved.

5. $C = C_1 \sqcup C_2 \equiv \neg(\neg C_1 \sqcap \neg C_2)$. This case is reduced to the items 3 and 4.

The lemma is proved completely. □

Corollary 2. *For any $\mathcal{ALCHOIQ}$ concept C and any concept B, which is either an atom A or $\{o\}$ or $\geq nR$, the concept $B \sqcap C$ is equivalent to some $\mathcal{ALCHOIQ}_{GN}$ concept.*

Proof. By Lemma 5 there exists an $\mathcal{ALCHOIQ}_{GN}$ concept C' such that either $C \equiv C'$ or $C \equiv \neg C'$. Then $B \sqcap C \equiv B \sqcap C'$ or $B \sqcap C \equiv B \sqcap \neg C'$. Both $B \sqcap C'$ and $B \sqcap \neg C'$ are $\mathcal{ALCHOIQ}_{GN}$ concepts (by the definition (8) of $\mathcal{ALCHOIQ}_{GN}$ concepts). Hence, the corollary is proved. □

Proposition 15. *Any safe-range $\mathcal{ALCHOIQ}$ concept is equivalent to some $\mathcal{ALCHOIQ}_{GN}$ concept.*

Proof. Let C be any safe-range $\mathcal{ALCHOIQ}$ concept. By Lemma 4 $C \sqsubseteq B_1 \sqcup \ldots \sqcup B_n$, where each B_i is either an atom A or $\{o\}$ or $\geq nR$. Then $C \equiv C \sqcap (B_1 \sqcup \ldots \sqcup B_n) \equiv B_1 \sqcap C \sqcup \ldots \sqcup B_n \sqcap C$. By the corollary 2 for each disjunct $B_i \sqcap C$ there exists an $\mathcal{ALCHOIQ}_{GN}$ concept D_i such that $B_i \sqcap C \equiv D_i$. Then $C \equiv D_1 \sqcup \ldots \sqcup D_n$. The concept $D_1 \sqcup \ldots \sqcup D_n$ is an $\mathcal{ALCHOIQ}_{GN}$ concept as a disjunction of $\mathcal{ALCHOIQ}_{GN}$ concepts. Hence, the proposition is proved. □

Proposition 16. *All $\mathcal{ALCHOIQ}_{GN}$ concept inclusion axioms are safe-range.*

Proof. Let $C \sqsubseteq D$ be any concept inclusion axiom in $\mathcal{ALCHOIQ}_{GN}$. It means that the corresponding first-order logic translation $\neg \exists x. C(x) \wedge \neg D(x)$ is in GNFO. Hence, $C(x) \wedge \neg D(x)$ is in GNFO or, that is the same, $C \sqcap \neg D$ is in $\mathcal{ALCHOIQ}_{GN}$. It is easy to see, that $\neg \exists x. C(x) \wedge \neg D(x)$ is safe-range if and only if the formula $C(x) \wedge \neg D(x)$ is safe-range, that is if and only if the corresponding $\mathcal{ALCHOIQ}_{GN}$ concept $C \sqcap \neg D$ is safe-range. But by Proposition 14 any $\mathcal{ALCHOIQ}_{GN}$ concept is safe-range. The proposition is proved. □

Lemma 6. *For any safe-range $\mathcal{ALCHOIQ}$ concept C and any $\mathcal{ALCHOIQ}$ concept D the concept $C \sqcap D$ is equivalent to some $\mathcal{ALCHOIQ}_{GN}$ concept $C' \sqcap D'$, where C' and D' are $\mathcal{ALCHOIQ}_{GN}$ concepts.*

Proof. Since C is safe-range by Lemma 4 $C \sqsubseteq B_1 \sqcup \ldots \sqcup B_n$, where each B_i is either an atomic concept A or $\{o\}$ or $\exists R$. Then $C \equiv C \sqcap (B_1 \sqcup \ldots \sqcup B_n)$ and, hence, $C \sqcap D \equiv C \sqcap (B_1 \sqcup \ldots \sqcup B_n) \sqcap D \equiv C \sqcap (B_1 \sqcap D \sqcup \ldots \sqcup B_n \sqcap D)$. By the corollary 2 each disjunct $B_n \sqcap D$ is an $\mathcal{ALCHOIQ}_{GN}$ concept. Hence, $D' := B_1 \sqcap D \sqcup \ldots \sqcup B_n \sqcap D$ is an $\mathcal{ALCHOIQ}_{GN}$ concept. Since C is s safe-range by Proposition 15 there exists an $\mathcal{ALCHOIQ}_{GN}$ concept C' such that $C \equiv C'$. Then $C \sqcap D \equiv C' \sqcap D'$, and $C' \sqcap D'$ is an $\mathcal{ALCHOIQ}_{GN}$ concept, where C' and D' are $\mathcal{ALCHOIQ}_{GN}$ concepts. □

Proposition 17. *Any safe-range $\mathcal{ALCHOIQ}$ concept inclusion axiom $C \sqsubseteq D$ can be transformed to a concept inclusion axiom $C' \sqsubseteq \neg D'$, where C' and D' are $\mathcal{ALCHOIQ}_{GN}$.*

Proof. Let $C \sqsubseteq D$ be any safe-range $\mathcal{ALCHOIQ}$ concept inclusion axiom. Then the corresponding formula $\neg \exists x. C(x) \wedge \neg D(x)$ is safe-range. Then the first-order logic formula $C(x) \wedge \neg D(x)$ is safe-range, or, that is the same, the $\mathcal{ALCHOIQ}$ concept $C \sqcap \neg D$ is safe-range. By Proposition 11 we have that C is safe-range or $\neg D$ is safe-range.

- C is safe-range. Then by Lemma 6 there exist two $\mathcal{ALCHOIQ}_{GN}$ concepts C' and D' such that $C \sqcap \neg D$ is logically equivalent to the $\mathcal{ALCHOIQ}_{GN}$ concept $C' \sqcap D'$. Then $\neg\exists x.\, C(x) \wedge \neg D(x)$ is logically equivalent to $\neg\exists x.\, C'(x) \wedge D'(x)$. Hence, $C \sqsubseteq D$ is logically equivalent to $C' \sqsubseteq \neg D'$ (C' and D' are $\mathcal{ALCHOIQ}_{GN}$ concepts).

- $\neg D$ is safe-range. The proof is similar to the previous item.

The proposition is proved completely. $\square$

Proposition 18. *For any two $\mathcal{ALCHOIQ}_{GN}$ concepts C and D the axiom $C \sqsubseteq \neg D$ is an $\mathcal{ALCHOIQ}_{GN}$ concept inclusion axiom.*

Proof. The axiom $C \sqsubseteq \neg D$ is logically equivalent to the first-order logic formula $\neg\exists x.\, C(x) \wedge D(x)$, where $C(x)$ and $D(x)$ are in GNFO. Then $\neg\exists x.\, C(x) \wedge D(x)$ is also in GNFO. Hence, by the definition of $\mathcal{ALCHOIQ}_{GN}$ concept inclusion axiom the axiom $C \sqsubseteq \neg D$ is an $\mathcal{ALCHOIQ}_{GN}$ concept inclusion axiom. $\square$

Propositions 17 and 18 imply the following.

Proposition 19. *Any safe-range $\mathcal{ALCHOIQ}$ concept inclusion axiom is equivalent to some $\mathcal{ALCHOIQ}_{GN}$ concept inclusion axiom.*

We consider a connection between safe-range fragment of $\mathcal{ALCHOIQ}$ and guarded negation fragment of $\mathcal{ALCHOIQ}$, that is $\mathcal{ALCHOIQ}_{GN}$. When we say "fragment", we mean a set of TBox assertions (concept and role inclusion axioms) and concepts (open formulas) of $\mathcal{ALCHOIQ}$ satisfying a particular property (e.g safe-range or guarded negation). Taking into account propositions 14, 15, 16, 19 and 13, we have the following theorem.

Proposition 20. *The safe-range fragment of $\mathcal{ALCHOIQ}$ and $\mathcal{ALCHOIQ}_{GN}$ are equally expressive.*

This proves Theorem 10:

Theorem 10 (**Expressive power equivalence**). *The domain independent fragment of $\mathcal{ALCHOIQ}$ and $\mathcal{ALCHOIQ}_{GN}$ are equally expressive.*

Theorem 11. *$\mathcal{ALCHOIQ}_{GN}$ TBoxes have finitely controllable determinacy of concept queries.*

Proof. We need to prove, that for any $\mathcal{ALCHOIQ}_{GN}$ TBox $\mathcal{T}$ (ontology), any concept query $\mathcal{Q}$ in $\mathcal{ALCHOIQ}_{GN}$ and any set of database predicates $\mathbb{P}_{\mathcal{DB}}$, whenever the query is finitely determined by the database predicates under the ontology then it is also determined in unrestricted models.

Suppose, that $\mathcal{Q}$ is finitely determined by $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{T}$. Then from Theorem 2 it follows, that $\mathcal{T} \cup \widetilde{\mathcal{T}} \models_{\mathit{fin}\_\mathbb{P}_{\mathcal{DB}}} \mathcal{Q} \sqsubseteq \widetilde{\mathcal{Q}}$, where $\models_{\mathit{fin}\_\mathbb{P}_{\mathcal{DB}}}$ means entailment over models with a finite interpretation to the database predicates. Hence, in particular $\mathcal{T} \cup \widetilde{\mathcal{T}} \models_{\mathit{fin}} \mathcal{Q} \sqsubseteq \widetilde{\mathcal{Q}}$, where $\models_{\mathit{fin}}$ means entailment over finite models. Hereafter let τ be one sentence, that is a first-order logic translation of a conjunction of all axioms in the TBox $\mathcal{T}$. Then from the aforementioned entailment we have:

$$\models_{\mathit{fin}} (\neg\tau \vee \neg\widetilde{\tau}) \vee (\neg\exists x.\, \mathcal{Q}(x) \wedge \neg\widetilde{\mathcal{Q}}(x)). \tag{9}$$

By Proposition 14 $\mathcal{Q}(x)$ is safe-range. Hence, $\mathcal{Q}(x) \wedge \neg\widetilde{\mathcal{Q}}(x)$ is safe-range, hence the $\mathcal{ALCHOIQ}$ concept $\mathcal{Q} \sqcap \neg\widetilde{\mathcal{Q}}$ is safe-range and, hence, by Proposition 15 there exists an $\mathcal{ALCHOIQ}_{GN}$ concept C' such that $\mathcal{Q} \sqcap \neg\widetilde{\mathcal{Q}} \equiv C'$. Then $\neg\exists x.\, \mathcal{Q}(x) \wedge \neg\widetilde{\mathcal{Q}}(x) \leftrightarrow \neg\exists x. C'(x)$ and the following holds:

$$\models_{\mathit{fin}} (\neg\tau \vee \neg\widetilde{\tau}) \vee (\neg\exists x. C'(x)). \tag{10}$$

$\neg\exists x. C'(x)$ is in GNFO, because $C'(x)$ is in GNFO. Since all the axioms in $\mathcal{T}$ are $\mathcal{ALCHOIQ}_{GN}$ TBox axioms, the sentences τ and $\widetilde{\tau}$ are in GNFO. Then the sentence $\neg\tau \vee \neg\widetilde{\tau}$ is in GNFO. Therefore the right hand side of the entailment (10) is in GNFO. Then $\neg((\neg\tau \vee \neg\widetilde{\tau}) \vee (\neg\exists x. C'(x)))$ is also in GNFO and by the entailment (10) does not have a finite model. Then, since GNFO has the finite model property, $\neg((\neg\tau \vee \neg\widetilde{\tau}) \vee (\neg\exists x. C'(x)))$ is unsatisfiable. Hence, we have:

$$\models (\neg\tau \vee \neg\widetilde{\tau}) \vee (\neg\exists x. C'(x)).$$

Since $\neg\exists x. C'(x) \leftrightarrow \neg\exists x.\, \mathcal{Q}(x) \wedge \neg\widetilde{\mathcal{Q}}(x)$, the following holds:

$$\models (\neg\tau \vee \neg\widetilde{\tau}) \vee (\neg\exists x.\, \mathcal{Q}(x) \wedge \neg\widetilde{\mathcal{Q}}(x)).$$

Then $\mathcal{T} \cup \widetilde{\mathcal{T}} \models \mathcal{Q} \sqsubseteq \widetilde{\mathcal{Q}}$. By Theorem 2 it means, that the query $\mathcal{Q}$ is determined in unrestricted models by the database predicates $\mathbb{P}_{\mathcal{DB}}$ under the ontology $\mathcal{T}$.

The proposition is proved. $\square$

References

Abiteboul, S., Hull, R., & Vianu, V. (1995). *Foundations of Databases.* Addison-Wesley.

Artale, A., Calvanese, D., Kontchakov, R., & Zakharyaschev, M. (2009). The DL-Lite family and relations. *J. Artif. Intell. Res. (JAIR)*, *36*, 1–69.

Avron, A. (2008). Constructibility and decidability versus domain independence and absoluteness. *Theor. Comput. Sci.*, *394*, 144–158.

Bárány, V., Gottlob, G., & Otto, M. (2010). Querying the guarded fragment. In *Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science (LICS 2010)*, pp. 1–10.

Bárány, V., ten Cate, B., & Otto, M. (2012). Queries with guarded negation (full version). *CoRR*, *abs/1203.0077*.

Beth, E. (1953). On Padoa's method in the theory of definition. *Indagationes Mathematicae*, *15*, 330–339.

Craig, W. (1957). Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory. *J. Symb. Log.*, *22*(3), 269–285.

Etzioni, O., Golden, K., & Weld, D. S. (1997). Sound and efficient closed-world reasoning for planning. *Artif. Intell.*, *89*, 113–148.

Fan, W., Geerts, F., & Zheng, L. (2012). View determinacy for preserving selected information in data transformations. *Inf. Syst.*, *37*, 1–12.

Fitting, M. (1996). *First-order logic and automated theorem proving* (2nd edition). Springer.

Franconi, E., Ibanez-Garcia, Y. A., & Seylan, İnanç. (2011). Query answering with DBoxes is hard. *Electronic Notes in Theoretical Computer Science, Elsevier*, *278*, 71–84.

Franconi, E., Kerhet, V., & Ngo, N. (2012a). Exact query reformulation over SHOQ DBoxes. In *Proc. of the 2012 International workshop on Description Logics (DL-2012)*.

Franconi, E., Kerhet, V., & Ngo, N. (2012b). Exact query reformulation with first-order ontologies and databases. In *Logics in Artificial Intelligence - 13th European Conference, JELIA 2012*, pp. 202–214.

Franconi, E., Ngo, N., & Sherkhonov, E. (2012c). The definability abduction problem for data exchange. In *Web Reasoning and Rule Systems - 6th International Conference RR 2012*.

Gurevich, Y. (1984). Toward logic tailored for computational complexity. In *Computation and Proof Theory*, Vol. 1104, pp. 175–216. Springer.

Halevy, A. Y. (2001). Answering queries using views: A survey. *The VLDB Journal*, *10*, 270–294.

Marx, M. (2007). Queries determined by views: pack your views. In *Proceedings of the 26th ACM symposium on Principles of Database Systems*, PODS '07, pp. 23–30.

McCune, W. (2005–2011). Prover9 and Mace4. `http://www.cs.unm.edu/~mccune/prover9`.

Nash, A., Segoufin, L., & Vianu, V. (2010). Views and queries: Determinacy and rewriting. *ACM Trans. Database Syst.*, *35*, 21:1–21:41.

Rosati, R. (2011). On the finite controllability of conjunctive query answering in databases under open-world assumption. *J. Comput. Syst. Sci.*, *77*(3), 572–594.

Seylan, İnanç., Franconi, E., & de Bruijn, J. (2009). Effective query rewriting with ontologies over DBoxes. In *Proc. of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*, pp. 923–925.

ten Cate, B., Franconi, E., & Seylan, İnanç. (2011). Beth definability in expressive description logics. In *Proc. of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011)*, pp. 1099–1106.

ten Cate, B., Franconi, E., & Seylan, İnanç. (2013). Beth definability in expressive description logics. *Journal of Artificial Intelligence Research (JAIR)*, *48*, 347–414.