unibz junior researcher

Domain Independence and Query Reformulations Under Constraints

Volha Kerhet

bu,press
bozen
bolzano
university
press

Freie Universität Bozen
Libera Università di Bolzano
Università Liedia de Bulsan

unibz junior researcher

Domain Independence and Query Reformulations Under Constraints

Volha Kerhet

bu,press
bozen
bolzano
university
press

Cover design: doc.bz / bu,press
Printer: Druckstudio Leo, Frangart-Frangarto

www.unibz.it/universitypress

ISBN 978-88-6046-134-6
E-ISBN 978-88-6046-135-3

Contents

The *unibz junior researcher* series

Especially at a time when universities are increasingly expected to produce tangible results, it is clear that one of their main tasks is to promote the work of their young scientists. The decision by the Free University of Bozen-Bolzano to publish the new series *unibz junior researcher*, enabling PhD students to present their research to a wider readership, is designed not so much to promote the work of individual scholars but rather to foster a common university culture. The idea is to publish studies which are exemplary, not just within the standards of the individual discipline, but also because of the wider significance of the issues they deal with and the way they are dealt with.

Due to the ever-increasing pressure in the academic world to publish papers in internationally renowned journals, there is a danger that a lot of research reaches out to only a narrow field of specialists. But we maintain that it is precisely the role of the university to ensure that knowledge is transmitted to a wider audience, that discussion between different areas of research is stimulated and that a dialogue with a wider readership beyond the university is established. This promotes a public sphere that is better informed and more competent in debating. The studies which are published in the *unibz junior researcher* series will serve future PhD students as reference points for participation in such a culture of research. Engaging in research in isolation from the general public simply ignores the requirements of our times: Universities need to open up and academics need to learn to transmit their knowledge at various levelsall the more so considering the increasing complexity of research topics and the higher demands of research methods. This is the only way to justify public investment in universities, only in this way can universities fulfil their public mandate and contribute to a competent dialogue over impending societal issues.

The first issues of this series convincingly fulfil these criteria. They present PhD research projects judged as excellent by the examining commissions. The Free University of Bozen-Bolzano's excellent research environment has contributed greatly to these results: The authors were able to approach their research topics in a measured way, under the close supervision of members of the respective PhD advisory commission, who were able to offer a range of perspectives on the relevant research methodology. Furthermore, the university's generous bursary scheme gives PhD students the opportunity to spend periods of study and research

abroad, and to thereby gain experience of how other universities conduct research on related topics. They could also present their research methodology and preliminary findings at international congresses a valuable experience in improving communicative competences. Finally, the regional setting of our university gave them access to a rich variety of empirical data which shows that South Tyrol, while being an alpine region, is by no means represents "periphery". Instead, the research projects demonstrate that regional study objects can have international relevance because the condensed dimensions allow processes to be brought into focus more readily and changes to be monitored more precisely. The region of South Tyrol is indeed affected by global change, as witnessed for instance in the environmental field, where its sensitive alpine landscape is particularly susceptible to harmful developments. So it is possible to see South Tyrol as a sort of laboratory where we can register warning signs earlier and experiment with appropriate counter measures. A greater density of transformation processes can equally be seen in the social field. As a traditional border area, South Tyrol has always been at the crossroads of different cultures. Its historical experience with multilingualism, with differrent political and legal frameworks and with the cultural interaction of very different reference points for identity, makes for a background against which some of todays major social challenges such as migration or the globalised economy, can be analysed and interpreted.
These chances for new socially-relevant scientific insights find expression in the PhD studies selected for this series. The university authorities hope that these publications will allow the wider public to gain insights into the quality of the work of these young researchers, and to recognize that the fruits of the financial investment in this university have direct beneficial effects on the local society. I congratulate the authors chosen for this series and wish them every success in their scientific career hoping they will remain intellectually and emotionally linked to their university and to South Tyrol.

Walter Lorenz
Rector (2008–2016)
Free University of Bozen-Bolzano

Acknowledgements

First of all, I would like to express my great thanks to my supervisor Prof. Enrico Franconi for his support and valuable advice, for helping me to find the topic of the thesis, for his understanding and patience. He is a wise teacher, who takes the individual characteristics of students into account and helps to broaden a vision, see other perspectives and choose the right direction. I am really grateful to him for his belief in me, his constant encouragement, inspiration and, in general, for taking care of his students.

I want to thank the KRDB Research Center and its members for providing such a fruitful research environment. Besides, it was a pleasure to work with such nice people. Special thanks go to Prof. Diego Calvanese for his understanding of my situation and meeting me halfway. Of course, I would like to deeply thank my closest colleague Nhung Ngo for the very productive collaboration and the creative views. I think we made a good team together.

Thanks also to the Free University of Bozen-Bolzano for giving me the possibility to travel a lot during my PhD. I also would like to thank Michael Zakharyaschev and Roman Kontchakov for the time I spent at Birkbeck, University of London, for useful discussions and advice.

I am grateful to my friends from Bolzano, Trento and Minsk. They always supported me and raised my spirit, and thanks to them I will have great memories of this period of my life. Last but not least, I thank my brother Alexey, who induced me to follow his footsteps and conduct scientific research in computer science, and my parents for all their love, care and understanding.

Preface

It is my pleasure to introduce the work of my PhD student Volha Kerhet, in which she addresses the problem of query reformulation with expressive ontologies over databases.
An ontology provides a conceptual view of a database and it is composed of constraints on a vocabulary extending the basic vocabulary forming the actual structure of the data. Querying a database using the terms in such a richer ontology vocabulary allows for more flexibility than using only the basic vocabulary of the relational database directly.
In this study Volha Kerhet investigates and develops 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 is called locally-closed world assumption, exact views, or DBox. A DBox is a set of ground atoms which semantically behaves like a database, i.e. the interpretation of the database predicates in the DBox is exactly equal to the database relations. The DBox predicates are closed, i.e. their extensions are the same in every interpretation, whereas the other predicates in the knowledge base are open, i.e. their extensions may vary among different interpretations. This study does not consider 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 this 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 support to decide the existence of a domain independent first-order equivalent reformulation of a query in terms of the database signature. It also provides an effective approach to construct the reformulation. The interest is 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. The framework points out several conditions on the ontologies and the queries to guarantee the existence of a safe-range reformulation. It is shown that these conditions are not infeasible in practice and an efficient method to ensure their validation is provided.

Standard theorem proving techniques can be used to compute the reformulation. As mentioned, domain independence is an important property of a query that guarantees that the answer value of the query remains the same regardless of the underlying domain of the interpretation. Unfortunately, checking domain independence of a first-order query is well known to be in general undecidable. There have been several attempts to define syntactic fragments of first-order logic characterising domain independent formulas that can easily be checked, but all of them are incomplete. In this work Volha Kerhet found a general way of reducing the problem of checking domain independence of an arbitrary first-order logic formula to checking a standard first-order logic entailment. This method can be applied in any decidable fragment where the formulas participating in the entailment can be expressed.

In order to be complete, the reformulation framework is applicable to ontologies and queries expressed in any fragment of first-order logic enjoying finitely controllable determinacy, a stronger property than the finite model property of the logic. If the logic employed does not enjoy finitely controllable determinacy this approach would become sound but incomplete, but still effectively implementable using standard theorem proving techniques. Volha Kerhet explores non-trivial applications where the framework is complete; in the present study she discusses the application with expressive description logics ontologies and concept queries. It is shown (i) how 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) how 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. So, there is the 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.

There is a strong interest in the query reformulation problem in classical relational database research as well as in modern knowledge representation studies. Differently from the mainstream research on query reformulation, which is mostly based on perfect or maximally contained rewritings with sound views (see, e.g., the ODBA approaches), this work focuses on exact rewritings with exact views, since it characterises more precisely the query answering problem with ontologies and databases, and it allows for very expressive ontology languages.

The core of Volha Kerhet's work has been published in the top-class Journal of Artificial Intelligence Research.

Enrico Franconi
Free University of Bozen-Bolzano
Faculty of Computer Science, KRDB Research Centre

1. Introduction

The goal of this work[1] is to formalise a precise and uniform integration between knowledge bases (also called *ontologies* or *constraints*) expressed in classical first-order logic, queries expressed in safe range first-order logic, and classical relational databases. We aim at extending relational database theory in an incremental fashion, namely our approach in absence of an ontology should behave exactly like classical relational database theories. Also, we don't want to consider non-classical variants of first-order logics, such as the one with active domain semantics and standard name assumption.

There is one principally crucial moment here. Knowledge representation formalisms that are used to express ontologies have *open world semantics*, typical of classical first-order logic. According to this, each bit of information that is not stated to be explicitly true is assumed to be unknown, that is, we have incomplete information. On the other hand databases have *closed world semantics*, according to which each bit of information that is not stated to be explicitly true is assumed to be $\mathbf{false}$, that is, information is complete. Thus, using ontologies together with databases, as they are, leads us to a fundamental mismatch between open and closed worlds. Our main goal is to develop and study a framework that builds a bridge between these worlds and allows the evaluation of queries over databases with respect to ontologies effectively and completely by using query reformulation.

More specifically, we study a general framework for the rewriting of a first-order query in presence of an arbitrary first-order logic knowledge base over a signature extending a database signature with additional predicates. The framework supports deciding the existence of a logically equivalent and – given the constraints – safe-range first-order reformulation (called *exact reformulation*) of a domain independent first-order query in terms of the database signature, and if such a reformulation exists, 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. We present a non-trivial application of the framework with ontologies in the very expressive decidable fragments of first-order logic represented by the description logics $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$, by providing effective means to compute safe-range first-order exact reformulations of queries.

---

1 Parts of this study have been published in Franconi, Kerhet, and Ngo (2012b, 2012a); Kerhet and Franconi (2012); Franconi, Kerhet, and Ngo (2013).

As a separate research direction, originating from the special requirement of SQL executability of a query, we study domain independence of a first-order formula and show for the first time a sound and complete procedure to check this property, by reducing this problem to an entailment problem in first-order logic.

1.1 Domain Independence

Domain independence (Avron, 2008) is an important property of a formula that guarantees that the truth value of the formula in an interpretation remains the same regardless of the underlying domain of the interpretation. An example of a domain independent formula would be $\exists x.\ P(x)$, since if it turns out to be true in some interpretation of the unary predicate symbol P with a specific domain, it is also true in all the structures sharing the same interpretation for the predicate P and any other compatible domain. On the other hand, the formula $\forall x.\ P(x)$ is not a domain independent formula, since if it is true in an interpretation with some domain, it is definitely not true in any structure sharing the same interpretation for the predicate P but with a larger domain. Traditionally all standard database constraints (e.g. tuple-generating dependencies, equality-generating dependencies) (Gyssens, 2009) and standard database query languages over databases with such constraints are domain independent; as a matter of fact they are expressible in relational algebra – a language meant specifically for relational database queries and constraints. The importance of checking domain independence of a formula stems from the use of first-order logic as a query language for databases. Since we allow a query to be an arbitrary first-order formula, its answer could be infinite (since the domain is not restricted to be finite in classical first-order logic) or it may depend on the domain.

Example 1.1. Consider the query $Q(x) = \neg Student(x)$ over the database $\mathcal{DB} = \{Student(a)\}$, with domain $\Delta_1 = \{a, b\}$ has the answer $\{x \mapsto b\}$, while with the extended domain $\Delta_1 = \{a, b, c\}$ has the different answer $\{x \mapsto b, x \mapsto c\}$; if an infinite domain is considered, the answer will be infinite even in the presence of such a finite database.

Therefore, it is only possible to deal with *domain independent* queries in relational databases. Indeed, the above open formula $\neg Student(x)$ turns out not to be domain independent.

Example 1.2. Let us try to "fix" the example above by inserting a so-called "guard" as a conjunct, which will restrict range of the free variable. That is, consider the query $Q(x) = Person(x) \wedge \neg Student(x)$. For any fixed database

this query gives the same answer with any compatible domain (even an infinite one). $Q(x)$ is domain independent.

Domain independence of constraints is also an important aspect. For non-domain independent constraints validation of the constraints in some interpretation may depend on the domain of the interpretation. Thus, the consistency of a database with respect to a set of non-domain independent constraints may depend not only on the extensions of the database tables but also on the domain.

Example 1.3. Consider a knowledge base:

$$\mathcal{KB}_1 = \{\exists x.\, \neg Student(x),\, \exists x.\, Person(x)\}$$

and consider the database $\mathcal{DB} = \{Student(a),\ Person(a)\}$. This database with domain $\Delta_1 = \{a\}$ is inconsistent with respect to $\mathcal{KB}_1$, while it is consistent with respect to the same knowledge base with domain $\Delta_2 = \{a, b\}$. It happens because the first sentence of the knowledge base is not domain independent.
Let us now "fix" the non-domain independent sentence in the knowledge base and make it domain independent by adding a "guard" as we did in the previous example. We obtain the following new knowledge base:

$$\mathcal{KB}_2 = \{\exists x.\, Person(x) \wedge \neg Student(x)\}.$$

This knowledge base is domain independent, and one can easily see that the database $\mathcal{DB}$ is always inconsistent with respect to this new knowledge base regardless the underlying domain. And if we "fix" the database to be consistent with respect to $\mathcal{KB}_2$ (by adding an assertion $Person(b)$, for example), this new database will be consistent with respect to the knowledge base with all possible domains.

Still, domain independence of constraints does not guarantee the answer to a query over a database with respect to these constraints to be finite. That is why one needs to assume domain independence of both constraints and query.

Example 1.4. Consider the knowledge base:

$$\begin{aligned}\mathcal{KB} = \{&\forall x, y.\, (hasTeacher(x, y) \rightarrow \exists z.\, hasTeacher(y, z)),\\ &\forall x, y, z.\, hasTeacher(x, y) \wedge hasTeacher(z, y) \rightarrow x = z,\\ &\exists x, y.\, (hasTeacher(x, y) \wedge \neg\exists z.\, hasTeacher(z, x))\}.\end{aligned}$$

This knowledge base models an infinite chain. The first sentence informally says that any teacher has a teacher. The second one says that every teacher has just one pupil (the one who is taught by somebody). The third sentence says that there is

a pupil who is not a teacher. All the sentences are domain independent, but one can easily see that this knowledge base requires any model of it to have an infinite domain. It means that there exists a non-domain independent query (for instance, one can consider $Q(x) = \neg Student(x)$), that gives an infinite answer over any database which is consistent with respect to this knowledge base.

The problem of checking whether a first-order logic formula is domain independent is undecidable (Di Paola, 1969; Abiteboul, Hull, & Vianu, 1995). The well known *safe-range* syntactic fragment of first-order logic 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. This transformation implements the idea that the range of every variable of domain independent formula should be restricted by some "guard" bounding its extension. For instance, like in the examples 1.2 and 1.3, a guard for a free variable could be a positive atomic conjunct using this variable. In Subsection 3.3.2 we consider a *relativisation of a formula to a predicate*, which is nothing other than a transformation of the original formula to a domain independent one by restricting the range of all variables of the formula to this predicate that plays a role of a "guard" in this case.

1.2 DBoxes

We consider knowledge representation systems (that is, first-order theories) where ground data is stored in a classical finite relational database, in a way that has been called the *locally-closed world* in the literature (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. We say that the DBox predicates are *closed*, i.e. their extensions are the same across all the interpretations, whereas predicates in classical first-order logic are *open*, i.e. their extensions may vary among different interpretations. Ground atoms in classical first-order logic have been also called *ABox* or *sound views*.
In spite of the fact that a DBox represents a database, we can not rely on standard relational database query evaluation technology if a query involves non-DBox predicates, since the non-DBox predicates may have different extensions in different models, and the domain of the interpretations is not necessarily restricted to the *active domain* (the set of all elements appearing in the database and in the query). As an example, given an ABox $\mathcal{A} = \{C(a)\}$ and alternatively a DBox

with the same data $\mathcal{D} = \{C(a)\}$, in any model $\mathcal{I}$ of the ABox $\mathcal{A}$ the extension of the predicate C includes a: $C^{\mathcal{I}} = \{a^{\mathcal{I}}, \ldots\}$; while in any model $\mathcal{I}$ of the DBox $\mathcal{D}$ the extension of C is exactly a: $C^{\mathcal{I}} = \{a^{\mathcal{I}}\}$.
So, a system composed of a first-order logic theory (the constraints) and a DBox may admit many models and we have to rely on certain answer semantics for answering queries. Given a set of constraints and a DBox, the *certain answer* is an answer which holds in every model of the constraints with the DBox predicates maintaining always the same extension as the given database.

Example 1.5 (DBox versus ABox). Let us compare the semantics of a DBox with the semantics of an ABox. As an example, consider a boolean negative query $\neg Student(\texttt{mary})$ over a given standard relational database, expressed by the DBox $\mathcal{D} = \{Student(\texttt{john})\}$. The answer to the query over the DBox is $\mathbf{true}$, because the only specified student is `john`. On the other hand, if we consider $\mathcal{D}$ as an ABox and evaluate the query over it, the answer will be $\mathbf{false}$ because the ABox specifies only the necessary facts but not all of them, and, hence, `mary` may still be a student in some model.

Note, that by adding an ontology on top of the DBox $\mathcal{D}$ from the previous example, the answer to the query is not supposed to change — since the query uses only the signature of the DBox and additional constraints are not supposed to change the meaning of the query — whereas if $\mathcal{D}$ were treated as an ABox the answer may change in presence of an ontology. For instance, if we consider $\mathcal{D}$ as an ABox and add an ontology $\{\forall x, y.\, Student(x) \wedge Student(y) \rightarrow x = y\}$ that says that there is at most one student, the answer to the query will be $\mathbf{true}$. This may be important from the application perspective: a DBox preserves the behaviour of legacy application queries over relational databases. A DBox represents faithfully a relational database, in that queries under classical relational database semantics (i.e. queries involving only database predicates) have the same answer under DBox semantics.
It has been shown by Franconi et al. (2011) that query answering under constraints with DBoxes may be strictly harder in data complexity than query answering under constraints with ABoxes.

1.3 The Proposed Framework

In our general framework an ontology is a set of first-order formulas, and queries are (possibly open) first-order formulas. An ontology provides a conceptual view of the database and it is composed of constraints on a vocabulary *extending* the basic vocabulary of the database. In other words the language for the ontology is combined from database predicates and possibly some additional predicates.

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. A database is represented by a DBox. As we mentioned already, in any model of the ontology the extension of each DBox predicate is exactly defined by the content of the corresponding table from the database. All the other predicates are *open* (just like in classical first-order logic semantics) and can have different extensions in different models. In our framework we focus on the necessary and sufficient conditions which guarantee domain independent reformulations of queries because we want to reduce the original query answering problem to a relational algebra evaluation (e.g., via SQL) over the original database (Abiteboul et al., 1995).
In addition to that, in our framework we focus on queries involving non-DBox predicates whose answer requires *just* the data stored in the DBox only. We call such queries *definable* from the database predicates. To answer them one needs to find the *logically equivalent* reformulation (the *exact* reformulation) of the query expressed in terms of the database predicates only. Given that first-order logic possesses the nice Beth definability property (Beth, 1953), we can find the precise conditions under which domain independent (in particular safe-range) exact reformulations exist.
The proposed framework:

1. supports deciding the existence of a safe-range first-order equivalent reformulation of a query in terms of the database signature (DBox predicates), and

2. it provides an effective approach to construct the reformulation.

Step 1 is in general undecidable for first-order logic, so, some decidable fragments should be studied. We present a non-trivial application of the framework with ontologies in the very expressive decidable first-order logic fragments represented by the description logics $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$.

Example 1.6. Let $\mathcal{T}$ be an ontology in $\mathcal{ALC}$ (TBox):

$$\begin{aligned}\mathcal{T} = \{&Person \sqsubseteq Man \sqcup Women,\\ &Man \sqsubseteq Person,\\ &Woman \sqsubseteq Person,\\ &Woman \sqsubseteq \neg Man\}.\end{aligned}$$

It defines a partition, namely every person is either man or women, and these sets are disjoint. Let $\mathcal{D}$ be a DBox:

$$\mathcal{D} = \{Person(\texttt{john}), Person(\texttt{maria}), Man(\texttt{john})\}.$$

We want to query all the instances of $Woman$. One can easily see that this query can be reformulated to the concept $Person \sqcap \neg Man$, which is logically equivalent to the atomic concept $Woman$ with respect to the ontology. This reformulation is safe-range and expressed in terms of DBox predicates and we can evaluate it over the database (e.g., via model checking). The answer is $\{\texttt{maria}\}$.

1.4 Determinacy and Exact Reformulations

The certain answer to an open query includes all the substitutions which make the query true in *all* the models of the ontology with the DBox: so, if a substitution would make the query true only in some models, 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 with the DBox. In this case, the query is not fully determined by the given source data; indeed, given the database (DBox) there is some answer which is possible, but not certain. Due to the indeterminacy of the query with respect to the data, the complexity of computing the certain answer in general increases up to the complexity of entailment in the fragment of first-order logic used to represent the ontology. In our framework we focus on the case when a query has the same answer over all the models of the ontology with the DBox, 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 decreases drastically (see Section 5.3). The *determinacy* of a query with respect to a source database (DBox) (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 DBox predicates) by Beth (1953). So, in our framework we are interested in queries that are implicitly definable by DBox predicates. Since first-order logic has the Beth definability property, for every formula which is implicitly definable from the DBox predicates there exists, with respect to the ontology, a logically equivalent formula expressed in terms of the DBox predicates only – an *exact reformulation*. The mainstream research on query reformulation (Halevy, 2001) is 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, and Zakharyaschev (2009)). Given an ontology, a database (DBox) and a query, the *perfect reformulation* is a formula that gives the same certain answer over the database (DBox) with respect to the ontology as the original query. So, in the case of perfect or maximally contained reformuations reformulated queries are guaranteed to give the same certain answer (or its best approximation if such rewriting does not exist) as the original arbitrary query. On

the other hand, we are looking for a reformulated query expressed in terms of DBox predicates that is logically equivalent to the original query with respect to the ontology. Moreover, by focusing 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 seen as database views over the original database (DBox), so that they can be subsequently composed in an arbitrary way. In other words, exact reformulations can be used in query composition without affecting the outcome, which is not the case for perfect reformulations. This may be important for legacy database applications.

Example 1.7 (Perfect reformulation, query composition). Consider an ontology:

$$\mathcal{KB} = \{\exists x.\, Person(x) \wedge \neg Woman(x),\\ \forall x.\, Woman(x) \rightarrow Person(x)\},$$

saying that there is a person who is not woman and every woman is person. Let $\{Man, Woman\}$ be database predicates and $\mathcal{Q}(x) = Person(x)$ be a query. Then $\widehat{\mathcal{Q}}(x) = Woman(x)$ is a perfect reformulation of $\mathcal{Q}$, because these queries have the same certain answer over any database with respect to the ontology. Let us try to compose a new query $\mathcal{Q}_1$ by using the original query $\mathcal{Q}(x)$:

$$\mathcal{Q}_1 = \exists x.\, \mathcal{Q}(x) \wedge \neg Woman(x) = \exists x.\, Person(x) \wedge \neg Woman(x).$$

For any DBox which is consistent with respect to $\mathcal{KB}$ answer to this composed query is $\mathbf{true}$. It is evident that we will get the same answer if we substitute $\mathcal{Q}(x)$ here with any formula, which is logically equivalent to $\mathcal{Q}(x)$ with respect to $\mathcal{KB}$. In other words exact reformulations do not change truth value of the composed query.
Let us now use a perfect reformulation of $\mathcal{Q}(x)$ instead of the query itself in this composition:

$$\mathcal{Q}_2 = \exists x.\, \widehat{\mathcal{Q}}(x) \wedge \neg Woman(x) = \exists x.\, Woman(x) \wedge \neg Woman(x) \equiv \mathbf{false}.$$

Thus, perfect reformulations cannot be used in query composition because they may change truth value of the composed query.

It should be noticed that every exact reformulation is perfect, but not vice-versa. 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). We say that a fragment has finitely controllable determinacy if any finitely determined query is also determined in unrestricted models (the reverse is trivially true). 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.

This framework has also been applied to devise the formal foundations of the problem of *view update*: informally speaking, a target view of some source database is updatable if the target predicates are definable by the source predicates (Franconi & Guagliardo, 2012, 2013; Feinerer, Guagliardo, & Franconi, 2014; Feinerer, Franconi, & Guagliardo, 2015).

1.5 Main Results

As a useful formal tool, we show for the first time a sound and complete procedure to check domain independence of a formula, by reducing this problem to an entailment problem in first-order logic. We also considered decidable applications of the method in a two-variable fragment of first-order logic and in prefix-vocabulary fragments $[\exists^*\forall^*, all, (0)]^{\mathbb{C}}_{=}$ and $[all, (w), (0)]^{\mathbb{C}, free}_{=}$.
Within our query rewriting setting, the framework characterises exactly the existence of a domain independent first-order equivalent reformulation of a query in terms of the DBox signature. It also provides an effective approach to construct the reformulation as a safe-range formula with sufficient conditions. 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.
We have explored non-trivial applications where the framework is complete; namely, the applications with $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$ description logics and concept queries are discussed. We show how (i) to check whether the answers to a given query with respect to 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.
This work extends the works on exact rewritings with exact views by Marx (2007) and Nash et al. (2010) by focusing on *safe-range reformulations* and on the conditions ensuring their existence, and by considering general first-order ontologies extending the database (DBox) signature, rather than just *local as view* constraints over the database predicates (Halevy, 2001).

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 set of *constants* and $\mathbb{P}$ is a set of *predicates* with associated arities. The arity of a predicate P we denote as $\text{AR}(P)$. We denote as $\sigma(\phi)$ the signature of the formula ϕ, that is all the predicates and constants occurring in ϕ. We denote with $\mathbb{P}_{\{\phi_1,\ldots,\phi_n\}}$ the set of all predicates occurring in the formulas $\phi_1, \ldots, \phi_n$, with $\mathbb{C}_{\{\phi_1,\ldots,\phi_n\}}$ the set of all constants occurring in the formulas $\phi_1, \ldots, \phi_n$; for the sake of brevity, instead of $\mathbb{P}_{\{\phi\}}$ (resp. $\mathbb{C}_{\{\phi\}}$) we write $\mathbb{P}_\phi$ (resp. $\mathbb{C}_\phi$). We denote with $\sigma(\phi_1, \ldots, \phi_n)$ the signature of the formulas $\phi_1, \ldots, \phi_n$, namely the union of $\mathbb{P}_{\{\phi_1,\ldots,\phi_n\}}$ and $\mathbb{C}_{\{\phi_1,\ldots,\phi_n\}}$.We denote the set of all variables appearing in ϕ as $\text{VAR}(\phi)$, and the set of the free variables appearing in ϕ as $\text{FREE}(\phi)$; we may use for ϕ the notation $\phi(\bar{x})$, where $\bar{x} = \text{FREE}(\phi)$ is the (possibly empty) set of free variables of the formula. The notation $\phi(\bar{x}, \bar{y})$ means $\text{FREE}(\phi) = \bar{x} \cup \bar{y}$. A formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ is in *prenex normal form*, if it is written as a string of quantifiers followed by a quantifier-free part. Every formula is equivalent to a formula in prenex normal form and can be converted into it in polynomial time (Kleene, 2002).

Let $\mathbb{X}$ be a countable set of variables we use. We define a *substitution* Θ to be a total function $\mathbb{X} \mapsto \mathbb{S}$ assigning an element of the set $\mathbb{S}$ to each variable in $\mathbb{X}$. We can see substitution as a countable set of assignments of elements from $\mathbb{S}$ to elements from $\mathbb{X}$. That is, if $\mathbb{X} = \{x_1, x_2, \ldots\}$, then $\Theta := \{x_1 \to s_1, x_2 \to s_2, \ldots\}$, where $s_1, s_2, \ldots$ are elements from $\mathbb{S}$ assigned to corresponding variables from $\mathbb{X}$ by Θ.

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 $\cdot^\mathcal{I} = \cdot^\mathcal{J}$. We use standard definitions of validity, satisfiability and entailment of a formula. An *extension* of $\phi(\bar{x})$ in interpretation $\mathcal{I} = \langle \Delta^\mathcal{I}, \cdot^\mathcal{I} \rangle$, denoted $(\phi(\bar{x}))^\mathcal{I}$, is the set of substitutions which satisfy ϕ in $\mathcal{I}$. That is,

$$(\phi(\bar{x}))^\mathcal{I} = \{\Theta \mid \mathcal{I}, \Theta \models \phi(\bar{x})\}.$$

If ϕ is closed, then the extension depends on whether ϕ holds in $\mathcal{I} = \langle \Delta^\mathcal{I}, \cdot^\mathcal{I} \rangle$ or not. Thus, for a closed formula ϕ, $(\phi)^\mathcal{I} = \{\Theta \mid \Theta : \mathbb{X} \mapsto \Delta^\mathcal{I}\}$ – the set of all possible substitutions assigning elements from the domain $\Delta^\mathcal{I}$ to variables $\mathbb{X}$ – if $\mathcal{I} \models \phi$, and $(\phi)^\mathcal{I} = \emptyset$, if $\mathcal{I} \not\models \phi$.

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 $\sigma' = \sigma(\phi)$, where ϕ is a formula, we call $adom(\sigma(\phi), \mathcal{I})$ a *semantic active domain of a formula* ϕ *in an interpretation* $\mathcal{I}$.

3. Domain Independence

We study the property of domain independence and its various syntactic subfragments. We introduce a new expressive subfragment of a domain independent fragment – *relational algebra guarded negation first-order logic* (RAGNFO) – that possesses a finite model property inherited from GNFO. As a main result we propose a method that allows us to reduce the problem of checking domain independence of a formula to checking standard first-order logic entailment and considered applications of the method in several fragments of first-order logic.
In this chapter we assume *standard name assumption* for constants, i.e. $c^{\mathcal{I}} = c$ for any interpretation $\mathcal{I}$ and any constant $c \in \mathbb{C}$. This is a usual assumption that is made when one deals with databases.
Similar to the definition given in Nash et al. (2010), in terms of classical logic a *database instance* $\mathcal{D}$ *over a schema* $\mathbb{P}$ *with underlying domain* Δ (possibly infinite) is an interpretation $\mathcal{D} = \langle \Delta, \cdot^{\mathcal{D}} \rangle$ such that the interpretation function is defined for the schema $\mathbb{P}$ and for any predicate $P \in \mathbb{P}$ of arity n, $P^{\mathcal{D}}$ is a finite subset of $\mathbb{C}^n$. It corresponds to standard terminology (see Abiteboul et al. (1995) and Grädel et al. (2005)). Instead of notation $P^{\mathcal{D}}$ for databases we will usually use $P(\mathcal{D})$ (classical notation for databases).

3.1 Importance of Domain Independence

It is well known that satisfiable SPC algebra (with selection, projection, and cartesian product), satisfiable SPJR (with selection, projection, join, renaming) algebra and conjunctive calculus queries are equivalent (Abiteboul et al., 1995), and they can only express domain independent formulas. By adding the *union* and *difference* operators to the algebras how should we extend the expressivity of the calculus by maintaining the equivalence? It may seem logical that the expressive power of union can be reached by the conjunctive calculus by adding *disjunction* similarly to conjunction. But it turns out that disjunction brings more power to conjunctive calculus than union brings to algebras. With disjunction it is possible to express queries that are not expressible in SPCU and SPJRU algebras. The point is that the union operator can only be applied to expressions having the same arity (in unnamed perspective) or sort (named perspective), while disjunction can connect any kinds of formulas. This extra power of disjunction brings some issues: using disjunction one can easily express queries giving infinite answers.

Example 3.1. Consider a calculus query $A(x) \vee A(y)$ over a database instance $\mathcal{D}$ with an infinite domain Δ. One can easily see that the answer to this query

over the database is $(A(\mathcal{I}) \times \Delta) \cup (\Delta \times A(\mathcal{I}))$, where $A(\mathcal{I})$ is an extension of A in $\mathcal{I}$. Since the domain is infinite the answer is also infinite that contradicts with the expectations of the user who normally wants to see a finite set of tuples as an answer.

The difference operator in algebras corresponds to the so-called *guarded negation* in calculus, which informally means that all variables appearing in a negated atom necessarily appear in (are guarded by) a *positive* atom (called *guard*). In our case the guard may not necessarily be atomic. And again, like in the case of union operator, using unrestricted (full of non-guarded) negation allows the expression of queries that are not expressible in algebras with difference. Such queries being asked over a standard relational database may again give unexpected answers: the answer may be infinite or it may depend on the underlying domain content – which is typically unknown by the user.

Example 3.2. Consider a calculus query $Person(x) \wedge \neg Parent(x)$ over some database with some underlying domain. The negation here is guarded and the query is equivalent to the relational algebra expression $Person - Parent$. If we remove the guard and consider the formula $\neg Parent(x)$ as a query, we will not find any equivalent expression in relational algebra. The answer to the first guarded query consists of all the tuples from the table $Person$ in the database that do not appear in the table $Parent$ and, hence, is always finite. In contrast, the answer to the second non-guarded query consists of all the tuples from the domain that do not appear in the table $Parent$, that is, it may contain *all kinds of objects* of the domain. Therefore the answer to the second query is infinite if the domain is infinite. But even if the domain is finite, the answer to the query $\neg Parent(x)$ will change if we add at least one new element to the domain (keeping the database unchangeable) - this new element will appear in the answer.

Thus, on the one hand adding disjunction and full negation to the conjunctive calculus essentially extends its expressive power, on the other hand it yields the aforementioned issues with the answer. One approach to solve these issues consists in considering an *active domain semantics*. Under the active domain semantics the answer to a query over the database is computed on the fixed finite *active domain* - the set of all constants occurring in the database and the query (formally in our terms this active domain for a database $\mathcal{D}$ over a schema $\mathbb{P}$ and a query q is defined as $adom(\mathbb{P}, \mathcal{D}) \cup \mathbb{C}_q$). The crucial disadvantage of this approach is that it makes the answer dependent on the active domain – the information that is not readily available to the user.
Another approach commonly taken is to consider only queries that give the same finite answers on all possible underlying domains. Such queries are called *domain independent*. Let us give a formal definition of this notion.

Two interpretations $\mathcal{I}_1 = \langle \Delta_1, \cdot^{\mathcal{I}_1} \rangle$ and $\mathcal{I}_2 = \langle \Delta_2, \cdot^{\mathcal{I}_2} \rangle$ are *compatible* iff they agree on the interpretations of all the predicates and constants. That is, for any predicate $P \in \mathbb{P}$ and constant $c \in \mathbb{C}$, the following holds:

$$P^{\mathcal{I}_2} = P^{\mathcal{I}_1}$$
$$c^{\mathcal{I}_2} = c^{\mathcal{I}_1}$$

Definition 3.3 (Domain independence). Let $\mathcal{I}_1 = \langle \Delta_1, \cdot^{\mathcal{I}_1} \rangle$ and $\mathcal{I}_2 = \langle \Delta_2, \cdot^{\mathcal{I}_2} \rangle$ be any two compatible interpretations, and ϕ be a formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$. ϕ is *domain independent*, if and only if

$$(\phi)^{\mathcal{I}_1} = (\phi)^{\mathcal{I}_2}$$

It is important to notice that these two approaches (considering active domain semantics and considering domain independent formulas) are equivalent. By definition, any domain independent query gives the same answer on any domain containing the active domain. And for any relational calculus query q_1 giving the answer ans on the active domain there is a domain independent query q_2 that gives the same answer ans on any domain, containing the active domain (see Abiteboul et al. (1995), Theorem 5.3.10).

3.2 Syntactic Fragments of Domain Independent Formulas

Domain independence is a semantic property of a formula. The problem of checking whether a FOL formula is domain independent is undecidable (Di Paola, 1969; Abiteboul et al., 1995). There have been several attempts to define syntactically restricted decidable subfragments of the domain independent fragment. The most important of them are: *range separable formulas* (Codd, 1972), *range restricted formulas* (Nicolas, 1982; Gelder & Topor, 1991), *evaluable formulas* (Demolombe, 1992; Gelder & Topor, 1991), *allowed formulas* (Topor, 1987; Gelder & Topor, 1991) and *safe-range formulas* (Abiteboul et al., 1995). The first three classes were introduced without the equality predicate. Range restricted formulas are originally defined as a subset of closed formulas in prenex conjunctive normal form. This definition was extended to open formulas and then formulas in disjunctive prenex normal form in Demolombe (1992). Safe-range characterisation also require preliminary transformation of a formula to a safe-range normal form. Definitions of range separable, range restricted and evaluable formulas can be modified so that all of them may contain equality. The way to do this for the last two classes was described in Gelder and Topor (1991). Below we recall the definitions of these classes.

The definition of evaluable formulas from Demolombe (1992) is too long. Here we use the simpler equivalent definition of evaluable formulas (with equality) given in Gelder and Topor (1991). First we need to define gen and con relations between variables and formulas. They possess the values $\mathbf{true}$ or $\mathbf{false}$ and are defined recursively by the following set of rules:

$$
\begin{array}{rll}
gen(x,\phi) & \textbf{if} & edb(\phi) \ \ \& \ \ x \in \text{FREE}(\phi) \\
gen(x,\neg\phi) & \textbf{if} & gen(x, pushnot(\neg\phi)) \\
gen(x,\exists y.\,\phi) & \textbf{if} & distinct(x,y) \ \ \& \ \ gen(x,\phi) \\
gen(x,\forall y.\,\phi) & \textbf{if} & distinct(x,y) \ \ \& \ \ gen(x,\phi) \\
gen(x,\phi\vee\varphi) & \textbf{if} & gen(x,\phi) \ \ \& \ \ gen(x,\varphi) \\
gen(x,\phi\wedge\varphi) & \textbf{if} & gen(x,\phi) \\
gen(x,\phi\wedge\varphi) & \textbf{if} & gen(x,\varphi)
\end{array}
$$

$$
\begin{array}{rll}
con(x,\phi) & \textbf{if} & edb(\phi) \ \ \& \ \ x \in \text{FREE}(\phi) \\
con(x,\phi) & \textbf{if} & x \notin \text{FREE}(\phi) \\
cen(x,\neg\phi) & \textbf{if} & con(x, pushnot(\neg\phi)) \\
con(x,\exists y.\,\phi) & \textbf{if} & distinct(x,y) \ \ \& \ \ con(x,\phi) \\
con(x,\forall y.\,\phi) & \textbf{if} & distinct(x,y) \ \ \& \ \ con(x,\phi) \\
con(x,\phi\vee\varphi) & \textbf{if} & con(x,\phi) \ \ \& \ \ con(x,\varphi) \\
con(x,\phi\wedge\varphi) & \textbf{if} & gen(x,\phi) \\
con(x,\phi\wedge\varphi) & \textbf{if} & gen(x,\varphi) \\
con(x,\phi\wedge\varphi) & \textbf{if} & con(x,\phi) \ \ \& \ \ con(x,\varphi)
\end{array}
$$

Here,

- $edb(\phi)$ is $\mathbf{true}$ if and only if ϕ is an atomic formula of one of the following forms:
  - $P(\bar{x})$, where P is a predicate;
  - $x = c$, where c is a constant;
- $distinct(x, y)$ is $\mathbf{true}$ if x and y are distinct variables.
- Function $pushnot$ is defined as follows:

$$
\begin{aligned}
pushnot(\neg(\phi\wedge\varphi)) &= \neg\phi\vee\neg\varphi \\
pushnot(\neg(\phi\vee\varphi)) &= \neg\phi\wedge\neg\varphi
\end{aligned}
$$

$$pushnot(\neg\exists x.\,\phi) = \forall x.\,\neg\phi$$
$$pushnot(\neg\forall x.\,\phi) = \exists x.\,\neg\phi$$
$$pushnot(\neg\neg\phi) = \phi$$
$$pushnot(\neg(x = y)) = (x \neq y)$$
$$pushnot(\neg(x \neq y)) = (x = y).$$

– & means 'and'.

Now we can define evaluable and allowed formulas.

Definition 3.4 (**Evaluable formula**)**.** Formula ϕ in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ is evaluable if

– for every $x \in \text{FREE}(\phi)$, $gen(x, \phi)$ is $\mathbf{true}$;

– for every subformula $\exists x.\,\varphi$ of ϕ, $con(x, \varphi)$ is $\mathbf{true}$;

– for every subformula $\forall x.\,\varphi$ of ϕ, $con(x, \neg\varphi)$ is $\mathbf{true}$.

Definition 3.5 (**Allowed formula**)**.** Formula ϕ in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ is allowed if

– for every $x \in \text{FREE}(\phi)$, $gen(x, \phi)$ is $\mathbf{true}$;

– for every subformula $\exists x.\,\varphi$ of ϕ, $gen(x, \varphi)$ is $\mathbf{true}$;

– for every subformula $\forall x.\,\varphi$ of ϕ, $gen(x, \neg\varphi)$ is $\mathbf{true}$.

The following theorem was proved in Gelder and Topor (1991).

Theorem 3.6. Every allowed formula is evaluable.

It was also proved in Gelder and Topor (1991) that any evaluable formula is logically equivalent to some allowed formula, and any allowed formula can be transformed into a relational algebra expression. It means that any evaluable formula is domain independent. On the other hand any domain independent formula is logically equivalent to an evaluable formula (Demolombe, 1992). Hence, a domain independent fragment is equally expressive to an evaluable fragment. It should be noticed that the advantage of Evaluable Fragment is that in order to check if a formula belongs to this fragment one does not need to transform it to any normal form. Moreover, Gelder and Topor (1991) argue that an evaluable fragment is the largest decidable syntactic fragment of the domain independent fragment.

Definition 3.7 (**Range restricted formulas on conjunctive normal form**)**.** Let $\phi = Q\varphi$ be a formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ in conjunctive normal form, where Q is the

list of quantifiers and $\varphi = C_1 \wedge C_2 \wedge \ldots \wedge C_n$, where each $C_i = L_{i1} \vee L_{i2} \vee \ldots \vee L_{ip_i}$, and L_{ij} is a literal, that is, a positive or negative atomic formula. ϕ is a range restricted formula if

– for each free variable x there is a conjunct C_i such that all its literals L_{ij} are positive and contain x;

– for each existentially quantified variable x, if x occurs in a negative literal then there is a conjunct C_i such that all its literals L_{ij} are positive and contain x;

– for each universally quantified variable x, if x occurs in a conjunct C_i, then one of its literals L_{ij} is negative and contains x.

Definition 3.8 (**Range restricted formulas on disjunctive normal form**)**.** Let $\phi = Q\varphi$ be a formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ in disjunctive normal form, where Q is the list of quantifiers and $\varphi = D_1 \vee D_2 \vee \ldots \vee D_n$, where each $D_i = L_{i1} \wedge L_{i2} \wedge \ldots \wedge L_{ip_i}$, and L_{ij} is a literal, that is a positive or negative atomic formula. ϕ is a range restricted formula if

– for each free variable x, for each disjunct D_i there is one of its literals L_{ij} which is positive and contains x;

– for each existentially quantified variable x, for each disjunct D_i, if x occurs in D_i then there is at least one of its literals L_{ij} which is positive and contains x;

– for each universally quantified variable x, if x occurs in a positive literal then there is a disjunct D_i such that all its literals L_{ij} are negative and contain x.

Theorem 3.9. If ϕ is a formula in conjunctive or disjunctive normal form, then ϕ is range restricted if and only if ϕ is evaluable.

This result was proved in Demolombe (1992) for formulas without equality. As we mentioned already it is not so difficult to extend the definitions of range restricted formulas so that they allow equality, and the same result remains valid in this case (Gelder & Topor, 1991).
It may be the case that after transformation of an evaluable formula in conjunctive or disjunctive normal form it becomes not evaluable and, hence, not range restricted.

Example 3.10. Consider an evaluable formula $\phi = \exists x, y.(A(y) \vee ((B(x) \vee C(x)) \wedge \neg D(x)))$. The formula $\varphi = \exists x, y.\,((A(y) \vee B(x) \vee C(x)) \wedge (A(y) \vee \neg D(x)))$ is a conjunctive normal form of the formula ϕ, but it is not evaluable, since it is not range restricted.

A common idea behind each of these syntactic fragments is that each of them in some particular sense reflects the idea that each variable of a domain independent formula has a restricted range. Consider for example a *safe-range fragment* commonly used in database research. Intuitively, a formula is safe-range if and only if its variables are bounded by positive predicates or equalities.
We recall here formal definitions (Abiteboul et al., 1995) of safe-range formula. First, formula should be transformed to a *safe-range normal form*, denoted by SRNF. A formula ϕ in in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ can be transformed to $\text{SRNF}(\phi)$ by the following steps:

– Variable substitution: no distinct pair of quantifiers may employ same variable;

– Remove universal quantifiers;

– Remove implications;

– Push negation;

– Flatten 'and's and 'or's.

Definition 3.11 (**Range restriction of a formula**)**.** Range restriction of a formula ϕ in a safe-range normal form, denoted $rr(\phi)$, is a subset of $\text{FREE}(\phi)$ or $\bot$ recursively defined as follows:

– $\phi = R(t_1, \ldots, t_n)$, where each t_i is either a variable or a constant : $rr(\phi)$ is a set of variables in $t_1, \ldots, t_n$;

– $\phi = (x = c)$ or $\phi = (c = x)$, where c is a constant : $rr(\phi) = \{x\}$;

– $\phi = (x = y)$: $rr(\phi) = \emptyset$;

– $\phi = \phi_1 \wedge \phi_2$: $rr(\phi) = rr(\phi_1) \cup rr(\phi_2)$;

– $\phi = \phi_1 \vee \phi_2$: $rr(\phi) = rr(\phi_1) \cap rr(\phi_2)$;

– $\phi = \phi_1 \wedge (x = y)$: $rr(\phi) = rr(\phi_1)$ if $\{x, y\} \cap rr(\phi_1) = \emptyset$; $rr(\phi) = rr(\phi_1) \cup \{x, y\}$ otherwise;

– $\phi = \neg\phi_1$: $rr(\phi) = \emptyset \cap rr(\phi_1)$;

– $\phi = \exists x \phi_1$: $rr(\phi) = rr(\phi_1) \setminus \{x\}$ if $x \in rr(\phi_1)$; $rr(\phi) = \bot$ otherwise,

where $\bot \cup Z = \bot \cap Z = \bot \setminus Z = Z \setminus \bot = \bot$ for any range restriction of a formula Z.

Definition 3.12 (Safe-range formula). A formula ϕ in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ is safe-range iff $rr(\text{SRNF}(\phi)) = \text{FREE}(\phi)$.

It was proved in Abiteboul et al. (1995) that a safe-range fragment is equally expressive to a domain independent fragment. This result is known as Codd's theorem.
To sum up, the following relationships between the mentioned classes of domain independent formulas were investigated (we omit the word 'fragment' here for the sake of readability):

$$\begin{gathered}\textit{Range separable} \subset \textit{Evaluable} \subset \textit{Domain independent} \\ \textit{Range restricted} = \textit{Evaluable} \cap \textit{Prenex normal form} \\ \textit{Allowed} \subset \textit{Evaluable} \subset \textit{Domain independent} \\ \textit{Safe-range} \subset \textit{Evaluable} \subset \textit{Domain independent},\end{gathered}$$

where *Prenex normal form* means the formulas in conjunctive or disjunctive normal form. Each of these set inclusions is strict (see Example 3.13). Moreover, it was proved that any domain independent formula has a logically equivalent evaluable one, allowed one and safe-range one. In other words, each of the evaluable, allowed and safe-range fragments have the same expressive power as domain independent fragment. Since range restricted fragment is actually equal to evaluable formulas in prenex normal form, we can conclude that range restricted fragment is also equally expressive to domain independent fragment. That is,

$$\begin{gathered}\textit{Range restricted} \equiv \textit{Evaluable} \equiv \textit{Allowed} \equiv \\ \equiv \textit{Safe-range} \equiv \textit{Domain independent},\end{gathered}$$

where '$\equiv$' means 'has the same expressive power as'.

Example 3.13.

- $A(x, y) \wedge B(x)) \vee (C(x, y) \wedge D(y))$ is evaluable, but not range separable.
- $\exists x, y.(A(y) \vee ((B(x) \vee C(x)) \wedge \neg D(x)))$ is evaluable, but not range restricted (see Example 3.10). Equivalent range restricted formula is $\exists x, y.\, (A(y) \vee ((B(x) \wedge \neg D(x)) \vee (C(x) \wedge \neg D(x)))$.
- $\exists x.((A(x, y) \vee B(y)) \wedge \neg C(y))$ is evaluable, but not allowed. Equivalent allowed formula is $(\exists x.\, (A(x, y)) \vee B(y)) \wedge \neg C(y)$.
- $\exists x, y.(A(x) \vee B(y))$ is evaluable, but not safe-range. Equivalent safe-range formula is $\exists x.\, A(x) \vee \exists y.\, B(y)$.
- $\exists x.(P(x) \vee \neg P(x))$ is domain independent (it is a tautology), but not evaluable. Equivalent evaluable formula is $\forall x.(P(x) \vee \neg P(x))$.

3.2.1 Relational algebra first-order logic

As we mentioned in Section 3.2 the equality between satisfiable SPC algebra (or satisfiable SPJR algebra in the named case) and conjunctive queries is violated, when we add union ($\cup$) and difference ($-$) operators to the algebra and disjunction ($\vee$) and full negation ($\neg$) to conjunctive queries. It happens because disjunction and full negation bring more expressive power to conjunctive queries than, correspondingly, union and difference bring to algebra: first-order logic formulas $A(x) \vee B(y)$ and $\neg A(x)$ are not expressible in relational algebra.
In this subsection we introduce a new fragment of first-order logic implementing the idea of replacing $\neg$ and $\vee$ operators, which bring extra expressivity to the first-order logic, with restricted versions of these operators, such that their semantics will correspond to semantics of $-$ and $\cup$ operators in relational algebra respectively. Then it is logical to expect such language to be equally expressive to relational algebra and, hence, domain independent fragment. We call this fragment *relational algebra first-order logic* (RAFO). RAFO is a fragment of FOL defined by the following grammar:

$$\phi ::= \; R(t_1, \ldots, t_n) \mid x = c \mid c_1 = c_2 \mid \phi_1 \wedge \phi_2 \mid \phi \wedge (x = y) \mid \phi_1(\bar{x}) \vee \phi_2(\bar{x}) \mid$$

$$\phi_1(\bar{x}, \bar{y}) \wedge \neg \phi_2(\bar{y}) \mid \exists x.\, \phi$$

where R is a predicate, each t_i is either a variable or a constant, c, c_1, c_2 are constants, ϕ_1 and ϕ_2 are RAFO formulas, in the formula $\phi \wedge (x = y)$, $\text{FREE}(\phi) \cap \{x, y\} \neq \emptyset$, in the formula $\exists x.\, \phi$, $x \in \text{FREE}(\phi)$. In this logic instead of full negation we have a *generally guarded negation*, which requires any (fully) negated RAFO formula to be guarded by another RAFO formula; instead of full version of disjunction we have only *disjunction of union-compatible formulas*, that is, formulas with the same sets of free variables.
The following proposition holds.

Proposition 3.14. RAFO is safe-range.

That is,

$$\textit{RAFO} \subset \textit{Safe-range} \subset \textit{Domain independent}$$

As we expected there is a straightforward translation between RAFO and relational algebra and, hence, the following theorem takes place.

Theorem 3.15. RAFO $\equiv$ Domain independent.

3.2.2 Relational algebra guarded negation first-order logic

Domain independent fragment does not have finite model property.

Example 3.16. Consider the following theory:

$$\begin{aligned}\mathcal{T} = \{&\forall x, y.\, R(x, y) \to \exists z.\, R(y, z),\\ &\forall x, y, z.\, R(x, y) \wedge R(y, z) \to R(x, z),\\ &\neg\exists x.\, R(x, x),\\ &\exists x, y.\, R(x, y)\}.\end{aligned}$$

It is domain independent (all the sentences are safe-range) and have only infinite models.

We consider a *guarded negation first-order logic* (GNFO) (Bárány, ten Cate, & Segoufin, 2011; Bárány, ten Cate, & Otto, 2012), which is a fragment of FOL that is "good" in a sense that it has a number of useful properties: (i) it is decidable – the satisfiability problem is 2-EXPTIME-complete, both on arbitrary interpretations and on finite interpretations; (ii) it has *finite model property*; (iii) it has *tree model property*.
Roughly speaking, GNFO is a fragment of FOL in which all occurrences of negation are *guarded* by an atomic predicate. Formally it consists of all formulas generated by the following recursive definition:

$$\phi ::= \; R(t_1, \ldots, t_n) \mid t_1 = t_2 \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \exists x.\, \phi \mid \alpha \wedge \neg\phi \qquad (3.1)$$

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.
GNFO in general does not enjoy a property of domain independence. For instance, GNFO formulas $A(x) \vee B(y)$ and $(x = x) \wedge \neg A(x)$ are not domain independent. In order to bring one more useful property – domain independence – we consider a class of domain independent formulas in GNFO. We define a new fragment that we call *relational algebra guarded negation first-order logic* and denote as RAGNFO:

$$\phi ::= \; R(t_1, \ldots, t_n) \mid x = c \mid c_1 = c_2 \mid \phi_1 \wedge \phi_2 \mid \phi \wedge (x = y) \mid \phi_1(\bar{x}) \vee \phi_2(\bar{x}) \mid$$

$$R(\bar{x}, \bar{y}) \wedge \neg\phi(\bar{y}) \mid \exists x.\, \phi$$

where R is a predicate, each t_i is either a variable or a constant, c, c_1, c_2 are constants, ϕ_1 and ϕ_2 are RAGNFO formulas, in the formula $\phi \wedge (x = y)$, $\text{FREE}(\varphi) \cap \{x, y\} \neq \emptyset$, in the formula $\exists x.\, \phi$, $x \in \text{FREE}(\phi)$. It immediately follows from the definitions that RAGNFO is equally expressive to intersection of RAFO and GNFO. Then because of Theorem 3.15,

$$\textit{Domain independent} \cap \textit{GNFO} \equiv \textit{RAGNFO}.$$

Therefore, *RAGNFO* is a quite expressive syntactic subfragment of domain independent fragment, which has finite model property.
A similar result was obtained in Bárány et al. (2012) (Theorem 2.2). The authors introduced a GN-RA – guarded negation fragment of relational algebra – and proved its equivalence with the domain independent fragment of GNFO.
Note, that domain independent fragment of GNFO is *equally expressive* but not *equal* to RAGNFO.

Example 3.17. Consider a formula $\exists x, y.\, (A(x) \vee B(y))$. This formula is in GNFO and domain independent, bit it is not in RAFO, since formulas $A(x)$ and $B(y)$ are not "union-compatible". So, it is not in RAGNFO. On the other hand, the formula $\exists x.\, A(x) \vee \exists y.\, B(y)$, which is logically equivalent to the original one, is in RAGNFO.

3.3 Checking Domain Independence

In order to introduce our technique for checking domain independence, we introduce a notion of *active domain theory* in Subsection 3.3.1, which starts by defining some other auxiliary notions needed for the definition of active domain theory. Then, in Subsection 3.3.2 our new technique for checking domain independence is presented.

3.3.1 Active domain theory

First, we define an *index set* of an n-ary predicate P in a formula ϕ and denote it as $Ind(P, \phi)$. Each element of this set, *index* $\mathbf{i}$, corresponds to some occurrence of the predicate P in ϕ and is represented by an n-ary vector of natural numbers and constants: $\mathbf{i} = (\mathbf{i}_1, \ldots, \mathbf{i}_n)$. Each natural number (constant) $\mathbf{i}_j$ in this vector corresponds to a particular variable (constant) in this particular occurrence of the predate P in ϕ. That is, the same number (constant) corresponds to the same variable (constant), and the greatest number among numeric components of the index $\mathbf{i}$, denoted as $max(\mathbf{i})$, is the number of distinct variables in the corresponding occurrence of the predicate P in the formula ϕ. That is, if all the components of the index $\mathbf{i}$ are constants (only constants appear in the corresponding occurrence of the predicate P), $max(\mathbf{i}) := 0$.

Example 3.18. Let us consider a formula $\phi = \exists x.\, P(x, x, y) \vee \exists z.\, P(z, y, y)$. Index set of the predicate P in the formula ϕ is $Ind(P, \phi) = \{(1, 1, 2), (1, 2, 2)\}$.

The following algorithm formally defines an *index set* of an n-ary predicate P in a formula ϕ.

Algorithm 1 Index Set

Input: $P \in \mathbb{P}$, $\phi \in \mathcal{FOL}(\mathbb{C}, \mathbb{P})$
Output: $Ind(P, \phi)$ – *index set of the predicate* P *in the formula* ϕ

$Ind(P, \phi) := \emptyset$
for the next occurrence $P(t_1, \ldots, t_n)$ of P in ϕ **do**
  // each t_i *is either a variable or a constant*
  // it may be the case that some t_i *and* t_j *denote the same variable or constant*
  // define coordinates of the index corresponding to this occurrence
  // of the predicate P*:*
  if $t_1 \in \mathbb{C}$ **then**
    $\mathbf{i}_1 := t_1$
    $l := 0$
  else
    $\mathbf{i}_1 := 1$
    $l := 1$
  end if
  // l *is the current number of distinct variables*
  for $j = 2, \ldots, n$ **do**
    if $t_j \in \mathbb{C}$ **then**
      $\mathbf{i}_j := t_j$
    else
      $k := 1$
      while t_j is distinct from t_k **do**
        $k := k + 1$
      end while
      if $k = j$ **then**
        $l := l + 1$
        $\mathbf{i}_j := l$
      else
        $\mathbf{i}_j := \mathbf{i}_k$
      end if
    end if
  end for
  // define the index itself:
  $\mathbf{i} := (\mathbf{i}_1, \ldots, \mathbf{i}_n)$
  // add the index to the index set:
  $Ind(P, \phi) := Ind(P, \phi) \cup \{\mathbf{i}\}$
end for

Example 3.19. Note that index set of the predicate P in the formula $\psi = = \exists x.\, P(x,x,y) \vee \exists z.\, P(y,y,z)$ (that is similar to the formula ϕ from the example 3.18) consists of just one index: $Ind(P,\psi) = \{(1,1,2)\}$.

Example 3.20. Consider a formula with constants: $\varphi = \exists x.\, P(x,c,y) \vee \vee \exists z.\, P(y,y,z)$ (that is similar to the formula ψ from the example 3.19). Then $Ind(P,\varphi) = \{(1,c,2),(1,1,2)\}$.

Now we define an *active domain theory for the formula* ϕ. First, we introduce a new unary predicate $Adom_\phi$. According to the idea, active domain theory for the formula ϕ, denoted as $\mathcal{A}_\phi$, encodes that interpretation of $Adom_\phi$ in any model of the theory is never empty and contains a so-called *semantic active domain* of the formula ϕ in this model, where a semantic active domain of a formula in an interpretation (formally defined below) is a set that consists of all elements appearing in interpretations of all the occurrences of all predicates in the formula and interpretations of all the constants in the formula in this interpretation. Formally, $\mathcal{A}_\phi$ is built according to the following rules:

1. For every n-ary predicate P in the formula ϕ and every index $\mathbf{i} \in Ind(P,\phi)$, which contains variables (not only constants),
$$\forall x_1,\ldots,x_{max(\mathbf{i})}.\, P(x_{\mathbf{i}_1},\ldots,x_{\mathbf{i}_n}) \to \bigwedge_{j=1}^{max(\mathbf{i})} Adom_\phi(x_j) \in \mathcal{A}_\phi,$$
where, $x_c := c$ for any $c \in \mathbb{C}$.
In other words, $Adom_\phi$ contains any domain element that occurs in the interpretation of each occurrence of each predicate in the formula ϕ.

2. $Adom_\phi$ contains all the constants from ϕ:
$$\bigwedge_{c \in \sigma(\phi) \cap \mathbb{C}} Adom_\phi(c) \in \mathcal{A}_\phi.$$

3. $Adom_\phi$ is not empty:
$$\exists x_1.\, Adom_\phi(x_1) \in \mathcal{A}_\phi.$$

4. There are no other sentences in $\mathcal{A}_\phi$.

Definition 3.21 (**Semantic active domain**)**.** *Semantic active domain* of a formula $\phi \in \mathcal{FOL}(\mathbb{C},\mathbb{P})$ in an interpretation $\mathcal{I} = \langle \Delta^\mathcal{I}, \cdot^\mathcal{I} \rangle$, denoted $adom(\phi,\mathcal{I})$, is a set of elements of the domain $\Delta^\mathcal{I}$ defined as follows:
$$adom(\phi,\mathcal{I}) := \bigcup_{P \in \sigma(\phi) \cap \mathbb{P}} \bigcup_{\mathbf{i} \in Ind(P,\phi)} \bigcup_{(a_{\mathbf{i}_1},\ldots,a_{\mathbf{i}_n}) \in P^\mathcal{I}} \{a_{\mathbf{i}_1},\ldots,a_{\mathbf{i}_n}\} \cup \bigcup_{c \in \sigma(\phi) \cap \mathbb{C}} \{c^\mathcal{I}\},$$
where $a_c := c^\mathcal{I}$ for any $c \in \mathbb{C}$.

It is easy to see then that for any formula ϕ from $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ and for any interpretation $\mathcal{I}$ which is a model of $\mathcal{A}_\phi$ we have:

$$adom(\phi, \mathcal{I}) \subseteq Adom_\phi^{\mathcal{I}}.$$

In order to understand the idea of the active domain theory of a formula let us consider an example.

Example 3.22. Let us take the formula from the example 3.18:
$\phi = \exists x.\, P(x, x, y) \vee \exists z.\, P(z, y, y)$. $Ind(P, \phi) = \{(1, 1, 2), (1, 2, 2)\}$.
Then $\mathcal{A}_\phi =$

$$\begin{aligned}\{&\forall x_1, x_2.\, P(x_1, x_1, x_2) \rightarrow Adom_\phi(x_1) \wedge Adom_\phi(x_2),\\ &\forall x_1, x_2.\, P(x_1, x_2, x_2) \rightarrow Adom_\phi(x_1) \wedge Adom_\phi(x_2),\\ &\exists x_1.\, Adom_\phi(x_1)\}\end{aligned}$$

And consider any interpretation $\mathcal{I} = \langle \{a, b, c\}, \cdot^{\mathcal{I}} \rangle$, such that

$$P^{\mathcal{I}} = \{(a, a, b), (b, a, a), (a, b, c)\}.$$

Let us compute the semantic active domain of ϕ in $\mathcal{I}$. $adom(\phi, \mathcal{I}) = \{a, b\}$ by the definition. Then if we specify the interpretation $\mathcal{I}$ by assigning some subset of the domain $\{a, b, c\}$ containing the semantic active domain $\{a, b\}$ to the predicate $Adom_\phi$, we will necessarily obtain a model of $\mathcal{A}_\phi$. Thus, the following interpretations are models of the active domain theory for ϕ:

$$\mathcal{I}_1 = \langle \{a, b, c\}, \cdot^{\mathcal{I}_1} \rangle,\ P^{\mathcal{I}_1} = \{(a, a, b), (b, a, a), (a, b, c)\},\ Adom_\phi^{\mathcal{I}_1} = \{a, b\};$$
$$\mathcal{I}_2 = \langle \{a, b, c\}, \cdot^{\mathcal{I}_2} \rangle,\ P^{\mathcal{I}_2} = \{(a, a, b), (b, a, a), (a, b, c)\},\ Adom_\phi^{\mathcal{I}_2} = \{a, b, c\}.$$

3.3.2 Technique for checking domain independence

In this section we show how to prove domain independence of a formula ϕ by proving a validity of a first-order logic entailment.
Let $\phi(\bar{x})$ be a formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ with free variables $\bar{x}$, and let P be an unary predicate. Then *relativisation of the formula* $\phi(\bar{x})$ *to the predicate* P is a formula $\phi(\bar{x})|_P$ defined as follows:

$$\phi(\bar{x})|_P := \phi(\bar{x})|_P^{quan} \wedge \bigwedge_{x \in \bar{x}} P(x),$$

where $\phi(\bar{x})|_P^{quan}$ can be computed recursively applying the following *relativising rules*:

1. if $\phi(\bar{x})$ is quantifier-free formula, then $\phi(\bar{x})|_P^{quan} := \phi(\bar{x})$;

2. if $\phi(\bar{x}) = \exists x.\, \psi(\bar{x}, x)$, then $\phi(\bar{x})|_P^{quan} := \exists x.\, P(x) \wedge \psi(\bar{x}, x)|_P^{quan}$;

3. if $\phi(\bar{x}) = \forall x.\, \psi(\bar{x}, x)$, then $\phi(\bar{x})|_P^{quan} := \forall x.\, P(x) \rightarrow \psi(\bar{x}, x)|_P^{quan}$;

4. if $\phi(\bar{x}) = \neg\psi(\bar{x})$, then $\phi(\bar{x})|_P^{quan} := \neg(\psi(\bar{x})|_P^{quan})$;

5. if $\phi(\bar{x}) = \psi(\bar{y}) \circ \varphi(\bar{z})$, where $\bar{x} = \bar{y} \cup \bar{z}$ and $\circ$ stands for any of $\wedge$, $\vee$, $\rightarrow$, then
$\phi(\bar{x})|_P^{quan} := \psi(\bar{y})|_P^{quan} \circ \varphi(\bar{z})|_P^{quan}$.

Theorem 3.23. For any formula $\phi(\bar{x})$ in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ and any unary predicate $P \in \mathbb{P}$, $\phi(\bar{x})|_P$ is safe-range.

Example 3.24. Let us compute relativisation of the formula $\phi(z) = \forall x.\, A(x) \rightarrow \exists y.\, (B(x, z) \vee C(x, y, z))$ to the predicate P.

$$\begin{aligned}\phi(z)|_P &= (\forall x.\, A(x) \rightarrow \exists y.\, (B(x, z) \vee C(x, y, z)))|_P^{quan} \wedge P(z) = \\ &= (\forall x.\, P(x) \rightarrow (A(x) \rightarrow \exists y.\, (B(x, z) \vee C(x, y, z)))|_P^{quan}) \wedge P(z) = \\ &= (\forall x.\, P(x) \rightarrow (A(x) \rightarrow (\exists y.\, B(x, z) \vee C(x, y, z))|_P^{quan})) \wedge P(z) = \\ &= (\forall x.\, P(x) \rightarrow (A(x) \rightarrow \exists y.\, (P(y) \wedge (B(x, z) \vee C(x, y, z))))) \wedge P(z).\end{aligned}$$

Theorem 3.25 (Checking domain independence). Let $\phi(\bar{x})$ be a formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$. Then $\phi(\bar{x})$ is domain independent iff

$$\mathcal{A}_\phi \models \forall \bar{x}\, .\, \phi(\bar{x}) \leftrightarrow \phi(\bar{x})|_{Adom_\phi}. \tag{3.2}$$

Example 3.26. Let us consider the formula $\phi = \exists x.\, (A(x) \vee B(a))$ from Topor (1987) that was already mentioned and apply the theorem to it. This formula is not safe-range but it is domain independent, because it is logically equivalent to $(\exists x.\, A(x)) \vee B(a)$, which is safe-range and, hence, domain independent. Such kinds of formulas are of particular interest for us.
It follows from the definition of $\mathcal{A}_\phi$ that

$$\mathcal{A}_\phi \models \forall x.\, A(x) \rightarrow Adom_\phi(x).$$

Hence,

$$\mathcal{A}_\phi \models \exists x.\, A(x) \leftrightarrow \exists x. Adom_\phi(x) \wedge A(x). \tag{3.3}$$

Again, from the definition of $\mathcal{A}_\phi$,

$$\mathcal{A}_\phi \models \exists x. Adom_\phi(x).$$

Hence,

$$\mathcal{A}_\phi \models B(a) \leftrightarrow \exists x. Adom_\phi(x) \wedge B(a). \tag{3.4}$$

As we mentioned already,

$$\models \exists x.\,(A(x) \vee B(a)) \leftrightarrow (\exists x.\,A(x)) \vee B(a). \tag{3.5}$$

It follows from (3.3) and (3.4) that

$$\mathcal{A}_\phi \models (\exists x.\,A(x)) \vee B(a) \leftrightarrow (\exists x. Adom_\phi(x) \wedge A(x)) \vee (\exists x. Adom_\phi(x) \wedge B(a)).$$

Then

$$\mathcal{A}_\phi \models (\exists x.\,A(x)) \vee B(a) \leftrightarrow \exists x. Adom_\phi(x) \wedge (A(x) \vee B(a)).$$

Hence, from (3.5) and the statement above we have:

$$\mathcal{A}_\phi \models \exists x.\,(A(x) \vee B(a)) \leftrightarrow \exists x. Adom_\phi(x) \wedge (A(x) \vee B(a)).$$

That is,

$$\mathcal{A}_\phi \models \phi \leftrightarrow \phi|_{Adom_\phi}$$

Therefore by applying the theorem we proved that the formula ϕ is domain independent.

3.4 Applications to Various Fragments of First-Order Logic

In this section we consider some applications of our technique (Theorem 3.25). Using this technique makes particular sense when the formula to be checked for domain independence is not safe-range. The following algorithm based on Theorem 3.25 can be used for checking domain independence of a formula from some fragment of first-order logic:

Algorithm 2 Checking domain independence

Input: A formula $\phi(\bar{x})$ from some fragment of $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$

Output: If (3.6) is valid, $\phi(\bar{x})$ is domain independent. Otherwise, $\phi(\bar{x})$ is not domain independent.

1: Construct active domain theory for $\phi(\bar{x})$: $\mathcal{A}_\phi$
2: Construct a relativisation of $\phi(\bar{x})$: $\phi(\bar{x})|_{Adom_\phi}$
3: Construct a conjunction of all sentences in $\mathcal{A}_\phi$: α_ϕ.
4: Consider the following sentence:

$$\alpha_\phi \rightarrow (\forall \bar{x}\,.\,\phi(\bar{x}) \leftrightarrow \phi(\bar{x})|_{Adom_\phi}). \tag{3.6}$$

If this sentence is expressible in some decidable fragment of $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$, check its validity.

In practice this algorithm is useful and can be applied for a formula $\phi(\bar{x})$ from any not-safe-range fragment of $\mathcal{FOL}(\mathbb{C},\mathbb{P})$. The important requirement is that the sentence (3.6) can be expressed in some decidable fragment. In the next subsections of this section we consider applications of this algorithm for formulas in such fragments of $\mathcal{FOL}(\mathbb{C},\mathbb{P})$, for which the sentence (3.6) is necessarily expressible in some decidable fragment. Thus, for these fragments we have a practical way to check domain independence by using the algorithm.

3.4.1 Two-variable fragment

Two-variable fragment of first-order logic $\mathcal{FO}^2(\mathbb{C},\mathbb{P})$ (with equality) consists of all formulas from $\mathcal{FOL}(\mathbb{C},\mathbb{P})$ (with equality) containing at most two distinct variables. The first decidability result for this fragment without equality was obtained by Scott. He showed that it is decidable (Scott, 1962). Decidability of the two-variable fragment with equality was proved by Mortimer together with the finite model property (1975). The satisfiability problem for $\mathcal{FO}^2(\mathbb{C},\mathbb{P})$ with equality is NEXPTIME-complete (Grädel, Kolaitis, & Vardi, 1997). It should be noted that equality makes no difference to the complexity of the problem for this fragment.

Following the relativising rules one can easily see that $\phi(\bar{x})|_{Adom_\phi}$ is expressed in $\mathcal{FO}^2(\mathbb{C},\mathbb{P})$ whenever $\phi(\bar{x})$ is in $\mathcal{FO}^2(\mathbb{C},\mathbb{P})$. On the other hand, for any $\phi(\bar{x}) \in \mathcal{FO}^2(\mathbb{C},\mathbb{P})$ and every predicate P from $\phi(\bar{x})$ and every index $\mathbf{i} \in Ind(P,\phi)$ we have: $max(\mathbf{i}) \leq 2$. Hence, by the definition of the active domain theory for a formula, α_ϕ is expressed in $\mathcal{FO}^2(\mathbb{C},\mathbb{P})$ if ϕ is expressed in $\mathcal{FO}^2(\mathbb{C},\mathbb{P})$.

Then $\alpha_\phi \rightarrow (\forall\bar{x}\,.\,\phi(\bar{x}) \leftrightarrow \phi(\bar{x})|_{Adom_\phi})$ is expressed in $\mathcal{FO}^2(\mathbb{C},\mathbb{P})$ whenever $\phi(\bar{x})$ is in $\mathcal{FO}^2(\mathbb{C},\mathbb{P})$. Besides, $\mathcal{FO}^2(\mathbb{C},\mathbb{P})$ contains domain independent formulas that are not safe-range. As an example one can consider the formula from Example 3.26 or a tautology $\exists x.\, A(x) \vee \neg A(x)$. Thus, a two-variable fragment of first-order logic is a reasonable application of our technique.

3.4.2 Prefix-vocabulary classes

First, let us recall what prefix-vocabulary classes are.

Prefix-vocabulary classes are classes of first-order logic that are determined by the quantifier prefix and the vocabulary of relation and function symbols. The classical decision problem for all such classes is solved. That is, we have a partition of the family of prefix-vocabulary classes into decidable and undecidable (Börger, Grädel, & Gurevich, 1997).

Prefix-vocabulary classes are denoted as $[\Pi, (p_1, p_2, \ldots), (f_1, f_2, \ldots)]_{(=)}$, where

– Π stands for a word over $\{\exists, \forall, \exists^*, \forall^*\}$ that denotes a set of quantifier prefixes, where '*' means any number of the preceding quantifier;

– $p_n, f_n \leq \omega$ are respectively numbers of available relation and function symbols of arity n (n is a natural number);

– presence (absence) of '=' indicates that the formulas in the fragment may (cannot) contain equality.

In other words, $[\Pi, (p_1, p_2, \ldots), (f_1, f_2, \ldots)]_{(=)}$ is a set of all formulas without (with) equality of the form $\pi.\varphi$, where $\pi \in \Pi$, φ is quantifier-free, the number of n-ary predicate symbols in φ is less than or equal to p_n, the number of n-ary function symbols in φ is less than or equal to f_n and φ has no constants and no free variables.

Example 3.27. $[\exists^*\forall\exists^*, (\omega, 1), all]_=$ is the class of all first-order sentences of the form $\exists x_1, \ldots, x_m, \forall y, \exists z_1, \ldots, z_n. \varphi$, where φ is quantifier free and

– may contain at most one binary predicate and cannot contain predicates of higher arities,

– may contain any number of unary predicates,

– may contain any number of function symbols of any arity higher than one,

– may contain equality.

Let $[\Pi, (p_1, p_2, \ldots), (f_1, f_2, \ldots)]^{\mathbb{C}, free}_{(=)}$ be fragments that are the same as $[\Pi, (p_1, p_2, \ldots), f_1, f_2, \ldots)]_{(=)}$, but where the constants from $\mathbb{C}$, and free variables are allowed (one of the parameters $\mathbb{C}$ and *free* may be absent).
It is mentioned in Börger et al. (1997) that decidability class for

$$[\Pi, (p_1, p_2, \ldots), (f_1, f_2, \ldots)]^{\mathbb{C}, free}_{(=)}$$

is the same as for

$$[\Pi, (p_1, p_2, \ldots), (f_1, f_2, \ldots)]_{(=)}.$$

As usual, given a function-free first-order logic with equality $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$. We consider prefix-vocabulary fragments of $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ as applications of our technique.

3.4.2.1 Application in $\exists^*$-prefix class

Let us consider $[\exists^*, all, (0)]^{\mathbb{C}}_{=}$ – a class of existential first-order formulas with equality and without functions.

Proposition 3.28. If sentence ϕ is expressible in $[\exists^*, all, (0)]^{\mathbb{C}}_{=}$, then $\alpha_\phi \to (\phi \leftrightarrow \phi|_{Adom_\phi})$ is expressible in $[\exists^*\forall^*, all, (0)]^{\mathbb{C}}_{=}$.

The class $[\exists^*, all, (0)]^{\mathbb{C}}_{=}$ is decidable because $[\exists^*, all, (0)]_{=}$ is decidable as a subfragment of existential first-order formulas with equality $[\exists^*, all, all]_{=}$ (Börger et al., 1997). The class $[\exists^*\forall^*, all, (0)]^{\mathbb{C}}_{=}$ is decidable because a corresponding so-called *Ramsey class* $[\exists^*\forall^*, all, (0)]_{=}$ is decidable (Börger et al., 1997). Decidability of $[\exists^*\forall^*, all, (0)]^{\mathbb{C}}_{=}$ is more important for us since we need to check satisfiability of the formula $\alpha_\phi \rightarrow (\phi \leftrightarrow \phi|_{Adom_\phi})$ expressed in it. The formula $\phi = \exists x.\,(A(x) \vee B(a))$ from Example 3.26 is domain independent but not safe-range and expressed in $[\exists^*, all, (0)]^{\mathbb{C}}_{=}$. Then it makes sense to apply our technique to $[\exists^*, all, (0)]^{\mathbb{C}}_{=}$.

3.4.2.2 Application in Löwenheim class

Let us consider a set of first-order formulas over unary predicates with equality and without functions in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$. Let us denote this fragment $\mathcal{F}$ for this subsection. Then the set of all prenex normal forms of all formulas from this fragment is actually a prefix-vocabulary class $[all, (w), (0)]^{\mathbb{C},free}_{=}$, which is decidable, since the class $[all, (w), (0)]_{=}$, which is called *Löwenheim class with equality*, is decidable (Van Heigenoort, 1977; Börger et al., 1997). So, without loss of generality we can assume that we consider the fragment $[all, (w), (0)]^{\mathbb{C},free}_{=}$.
Let $\phi(\bar{x})$ be a formula from $\mathcal{F}$ (or $[all, (w), (0)]^{\mathbb{C},free}_{=}$). Again, following the relativising rules and definition of the active domain theory for a formula one can see that $\alpha_\phi \rightarrow (\forall \bar{x}\,.\,\phi(\bar{x}) \leftrightarrow \phi(\bar{x})|_{Adom_\phi})$ is expressed in $\mathcal{F}$. Then after a standard transformation of this sentence to prenex normal form we obtain a logically equivalent formula, which is expressed in $[all, (w), (0)]^{\mathbb{C},free}_{=}$. And again, the formula $\exists x.\,(A(x) \vee B(a))$ from Example 3.26 can be considered as an example of a safe-range but not domain independent formula expressed in $[all, (w), (0)]^{\mathbb{C},free}_{=}$. It means that our method is applicable in $[all, (w), (0)]^{\mathbb{C},free}_{=}$.

3.5 Proofs of Section 3.2

3.5.1 Auxiliary definitions

We recall the formal semantics of the *standard relational algebra* RA (see, e.g., Abiteboul et al. (1995) for details). Let $\mathcal{I}$ be a database instance of a database schema $\mathcal{R}$ over an underlying domain Δ. Algebra expressions are either atomic expressions (*atomic relations*, *constant singletons*) or complex expressions built according to inductive formation rules based on standard unary and binary operators (*selection*, *projection*, *cartesian product*, *union* and *difference*). The semantics of an algebra expression e is inductively defined as the transformation of database instances $\mathcal{I}$ to a set of tuples $e(\mathcal{I})$ as follows:

– *Atomic relation* - R - (where $R \in \mathcal{R}$)

$$R(\mathcal{I}) = \mathcal{I}(R).$$

– *Constant singleton* - $\langle c \rangle$ - (where c is a constant)

$$\langle c \rangle(\mathcal{I}) = \{\langle c \rangle\}.$$

– *Selection* - $\sigma_{i=c}(e)$, $\sigma_{i=j}(e)$ - (where c is a constant, m is the arity of e, and $i, j \leq m$)

$$\sigma_{i=c}(e)(\mathcal{I}) = \{s \text{ is a } m\text{-tuple} \mid s \in e(\mathcal{I}) \text{ and } s(i) = c\},$$
$$\sigma_{i=j}(e)(\mathcal{I}) = \{s \text{ is a } m\text{-tuple} \mid s \in e(\mathcal{I}) \text{ and } s(i) = s(j)\}.$$

– *Projection* - $\pi_{i_1,\ldots,i_k}(e)$ - (where m is the arity of e, and $\{i_1, \ldots, i_k\} \subseteq [1 \ldots m]$)

$$\pi_{i_1,\ldots,i_k}(e)(\mathcal{I}) =$$
$$= \{s \text{ is a } k\text{-tuple} \mid \text{exists } s' \in e(\mathcal{I}) \text{ s.t. for all } 1 \leq j \leq k.\ s(j) = s'(i_j)\}.$$

– *Cartesian product* - $e \times e'$ - (where n, m are the arities of e, e')

$$(e \times e')(\mathcal{I}) = \{s \text{ is a } (n+m)\text{-tuple} \mid \text{exists } t \in e(\mathcal{I}), t' \in e'(\mathcal{I}) \text{ s.t.}$$
$$\begin{array}{lll} \text{for all} & 1 \leq j \leq n. & s(j) = t(j) \text{ and} \\ \text{for all} & 1 + n \leq j \leq (n+m). & s(j) = t'(j-n)\}. \end{array}$$

– *Union/Difference* - $e \cup e'$, $e - e'$ - (where m is the arity of e and e')

$$(e \cup e')(\mathcal{I}) = \{s \text{ is a } m\text{-tuple} \mid s \in e(\mathcal{I}) \text{ or } s \in e'(\mathcal{I})\},$$
$$(e - e')(\mathcal{I}) = \{s \text{ is a } m\text{-tuple} \mid s \in e(\mathcal{I}) \text{ and } s \notin e'(\mathcal{I})\}.$$

3.5.2 Proof of Proposition 3.14

Proof. Let us prove the proposition by induction on the structure of formula.

– **Base.**

1. $\phi = R(t_1, \ldots, t_n)$, where each t_i is either a variable or a constant. ϕ is in safe-range normal form.
   $rr(R(t_1, \ldots, t_n)) = \text{FREE}(R(t_1, \ldots, t_n))$. Hence, ϕ is safe-range.

2. $\phi = (x = c)$, where c is a constant. ϕ is in safe-range normal form.
   $rr(x = c) = \{x\} = \text{FREE}(x = c)$. Hence, ϕ is safe-range.

3. $\phi = (c_1 = c_2)$, where c_1 and c_2 are a constants. ϕ is in safe-range normal form.
   $rr(c_1 = c_2) = \emptyset = \text{FREE}(c_1 = c_2)$. Hence, ϕ is safe-range.

– $\phi = \varphi \wedge (x = y)$, where $\text{FREE}(\varphi) \cap \{x, y\} \neq \emptyset$. Suppose that the proposition holds for the formula φ, i.e. $rr(\text{SRNF}(\varphi)) = \text{FREE}(\varphi)$.
We have: $\text{SRNF}(\phi) = \text{SRNF}(\varphi) \wedge (x = y)$. Then $rr(\phi) = rr(\text{SRNF}(\phi)) = rr(\text{SRNF}(\varphi)) \cup \{x, y\} = \text{FREE}(\varphi) \cup \{x, y\} = \text{FREE}(\phi)$. Hence, ϕ is safe-range.

– $\phi = \varphi_1 \wedge \varphi_2$. Suppose that the proposition holds for the formulas φ_1 and φ_2, i.e. $rr(\varphi_1) = \text{FREE}(\varphi_1)$, $rr(\varphi_2) = \text{FREE}(\varphi_1)$.
One can easily see, taking into account safe-range transformation rules, that $rr(\phi) = rr(\varphi_1) \cup rr(\varphi_2) = \text{FREE}(\varphi_1) \cup \text{FREE}(\varphi_2) = \text{FREE}(\phi)$. Hence, ϕ is safe-range.

– $\phi = \exists x.\, \varphi$, where $x \in \text{FREE}(\varphi)$. Suppose that the proposition holds for the formula φ, i.e. $rr(\text{SRNF}(\varphi)) = \text{FREE}(\varphi)$.
We have: $rr(\phi) = rr(\text{SRNF}(\phi)) = rr(\exists x.\, \text{SRNF}(\varphi)) = rr(\text{SRNF}(\varphi)) \setminus \{x\} = \text{FREE}(\varphi) \setminus \{x\} = \text{FREE}(\phi)$. Hence, ϕ is safe-range.

– $\phi = \varphi_1(\bar{x}) \vee \bar{\varphi}_2(\bar{x})$. Suppose that the proposition holds for the formulas $\varphi_1(\bar{x})$ and $\varphi_2(\bar{x})$, i.e. $rr(\varphi_1(\bar{x})) = \text{FREE}(\varphi_1(\bar{x})) = \bar{x}$, $rr(\varphi_2(\bar{x})) = \text{FREE}(\varphi_1(\bar{x})) = \bar{x}$.
One can easily see, taking into account safe-range transformation rules, that $rr(\phi) = rr(\varphi_1(\bar{x})) \cap rr(\varphi_2(\bar{x})) = \bar{x} \cap \bar{x} = \bar{x} = \text{FREE}(\phi)$. Hence, ϕ is safe-range.

– $\phi = \varphi_1(\bar{x}, \bar{y}) \wedge \neg\varphi_2(\bar{y})$. Suppose that the proposition holds for the formulas $\varphi_1(\bar{x}, \bar{y})$ and $\varphi_2(\bar{y})$, i.e. $rr(\varphi_1(\bar{x}, \bar{y})) = \text{FREE}(\varphi_1(\bar{x}, \bar{y})) = \bar{x} \cup \bar{y}$, $rr(\varphi_2(\bar{y})) = \text{FREE}(\varphi_1(\bar{y})) = \bar{y}$.
One can easily see, taking into account safe-range transformation rules, that $rr(\phi) = rr(\varphi_1(\bar{x}, \bar{y})) \cup rr(\neg\varphi_2(\bar{y})) = \bar{x} \cup \bar{y} \cup (\emptyset \cap rr(\varphi_2(\bar{y}))) = \bar{x} \cup \bar{y} \cup (\emptyset \cap \bar{y}) = \bar{x} \cup \bar{y} \cup \emptyset = \bar{x} \cup \bar{y} = \text{FREE}(\phi)$. Hence, ϕ is safe-range.

The proposition is proved. □

3.5.3 Proof of Theorem 3.15

Let us first prove the following lemma.

Lemma 3.29. RA $\sqsubseteq$ RAFO.

Proof. We need to prove that for each n-ary RA expression e there is an equivalent RAFO formula ϕ_e, i.e. with a little abuse of notation $e(\mathcal{I}) = (\phi_e)^{\mathcal{I}}$ for

any $\mathcal{I}$, that can be seen either as a relational database (on the left-hand side) or as an interpretation with the same domain (on the right-hand side). Let us prove the lemma by induction on the structure of the RA expression e.

– **Base.**

1. $e = R$, where R is a relation name of arity n. Then one can easily see that $\phi_e = R(x_1, \ldots, x_n) \in$ RAFO (R here can be seen as an n-ary predicate).
2. $e = \langle c \rangle$, where c is a constant. Then $\phi_e = (x = c) \in$ RAFO.

– $e = \sigma_{i=c}(e_1)$, where $e_1 \in$ RA, c is a constant and $i \leq n$. Suppose that for e_1 there exists an equivalent formula $\phi_{e_1} \in$ RAFO. Then $\phi_e = \phi_{e_1} \wedge (x_i = c)$. $\phi_e \in$ RAFO because $\phi_{e_1} \in$ RAFO.

– $e = \sigma_{i=j}(e_1)$, where e_1 is an RA expression of arity n and $i, j \leq n$. Suppose that for e_1 there exists an equivalent formula $\phi_{e_1} \in$ RAFO. Then $\phi_e = \phi_{e_1} \wedge (x_i = x_j)$. $\phi_e \in$ RAFO because $\phi_{e_1} \in$ RAFO, $x_i \in$ FREE(ϕ_e) and $x_j \in$ FREE(ϕ_e).

– $e = \pi_{i_1,\ldots,i_n}(e_1)$, where e_1 is an RA expression of arity m and $i_1, \ldots, i_n \leq m$. Suppose that for e_1 there exists an equivalent formula $\phi_{e_1} \in$ RAFO. Then

$$\phi_e(x_i, \ldots, x_n) =$$
$$\exists y_1, \ldots, y_m . (\phi_{e_1}(y_1, \ldots, y_m) \wedge (x_1 = y_{i_1}) \wedge \ldots \wedge (x_n = y_{i_n})).$$

$\phi_e(x_i, \ldots, x_n) \in$ RA because $\phi_{e_1} \in$ RA, each $y_{i_j} \in \{y_1, \ldots, y_m\}$.

– $e = e_1 \times e_2$, where e_1 and e_2 are RA expressions of arities l and m respectively. Suppose that for e_1 there exists an equivalent formula $\phi_{e_1} \in$ RAFO and for e_2 there exists an equivalent formula $\phi_{e_2} \in$ RAFO. Then $\phi_e(x_1, \ldots, x_l, x_{l+1}, \ldots, x_{l+m}) = \phi_{e_1}(x_i, \ldots, x_l) \wedge \phi_{e_2}(x_{l+1}, \ldots, x_{l+m})$. $\phi_e \in$ RAFO because $\phi_{e_1} \in$ RAFO and $\phi_{e_2} \in$ RAFO.

– $e = e_1 \cup e_2$, where e_1 and e_2 are RA expressions of the same arity n. Suppose that for e_1 there exists an equivalent formula $\phi_{e_1} \in$ RAFO and for e_2 there exists an equivalent formula $\phi_{e_2} \in$ RAFO. Then $\phi_e(x_1, \ldots, x_n) = \phi_{e_1}(x_1, \ldots, x_n) \vee \phi_{e_2}(x_1, \ldots, x_n)$. $\phi_e(x_1, \ldots, x_n) \in$ RAFO, because $\phi_{e_1}(x_1, \ldots, x_n) \in$ RAFO, $\phi_{e_2}(x_1, \ldots, x_n) \in$ RAFO and ϕ_{e_1} and ϕ_{e_2} have the same arity n.

– $e = e_1 - e_2$, where e_1 and e_2 are RA expressions of the same arity n. Suppose that for e_1 there exists an equivalent formula $\phi_{e_1} \in$ RAFO and for e_2 there exists an equivalent formula $\phi_{e_2} \in$ RAFO. Then $\phi_e(x_1, \ldots, x_n) = \phi_{e_1}(x_1, \ldots, x_n) \wedge$

$\neg\phi_{e_2}(x_1, \ldots, x_n)$. $\phi_e(x_1, \ldots, x_n) \in$ RAFO, because
$\phi_{e_1}(x_1, \ldots, x_n) \in$ RAFO, $\phi_{e_2}(x_1, \ldots, x_n) \in$ RAFO and $\text{FREE}(\phi_{e_2}) =$
$= \{x_1, \ldots, x_n\} \subseteq \text{FREE}(\phi_{e_1}) = \{x_1, \ldots, x_n\}$.

The lemma is proved. □

It is known that *Safe-range* $\subset$ *Domain independent* and *RA* $\equiv$ *Domain independent* (Abiteboul et al., 1995). Then the Theorem 3.15 follows from the Proposition 3.14 and Lemma 3.29.

3.6 Proofs of Sections 3.3 and 3.4

3.6.1 Proof of Theorem 3.23

Proof. Without loss of generality assume that $\phi(\bar{x})$ is in safe-range normal form (if it is not we, first, transform $\phi(\bar{x})$ to $\text{SRNF}(\phi(\bar{x}))$). Recall:

$$\phi(\bar{x})|_P := \phi(\bar{x})|_P^{quan} \wedge \bigwedge_{x \in \bar{x}} P(x)$$

Let us prove the theorem by induction on a structure of a formula.

1. **Base.** $\phi(\bar{x}) = R(t_1, \ldots, t_n)$, where each t_i is either a variable from $\bar{x}$ or a constant (remember, that $\bar{x} = \text{FREE}(\phi)$). Then

   $$R(t_1, \ldots, t_n)|_P = R(t_1, \ldots, t_n) \wedge \bigwedge_{x \in \bar{x}} P(x)$$

   by definition. Then

   $$rr(\phi(\bar{x})|_P) = rr(R(t_1, \ldots, t_n)) \cup \bigcup_{x \in \bar{x}} rr(P(x)) = \bar{x} = \text{FREE}(\phi).$$

   Hence, $\phi(\bar{x})|_P$ is safe-range by definition.

2. $\phi(\bar{x}) = \exists y.\, \psi(\bar{x}, y)$, where $\psi(\bar{x}, y)|_P$ is safe-range by supposition. Then $\phi(\bar{x})|_P = (\exists y.\, P(y) \wedge \psi(\bar{x}, y)|_P^{quan}) \wedge \bigwedge_{x \in \bar{x}} P(x)$. $rr(\phi(\bar{x})|_P) = (\{y\} \cup rr(\psi(\bar{x}, y)|_P^{quan}) \setminus \{y\}) \cup \bar{x}$. Since $\psi(\bar{x}, y)|_P$ is safe-range, $rr(\psi(\bar{x}, y)|_P^{quan}$ cannot be $\perp$ by definition. That is, $rr(\phi(\bar{x})|_P) = \bar{x} = \text{FREE}(\phi)$. Hence, $\phi(\bar{x})|_P$ is safe-range by definition. Note that this means that for any formula $\phi(\bar{x})$ in safe-range normal form, $rr(\phi(\bar{x}))|_P^{quan}$ can never be $\perp$.

3. $\phi(\bar{x}) = \neg\psi(\bar{x})$, where $\psi(\bar{x})|_P$ is safe-range by supposition. Then $\phi(\bar{x})|_P := \neg(\psi(\bar{x})|_P^{quan}) \wedge \bigwedge\limits_{x\in\bar{x}} P(x)$. $rr(\phi(\bar{x})|_P) = (\emptyset \cap rr(\psi(\bar{x})|_P^{quan})) \cup \bigcup\limits_{x\in\bar{x}} rr(P(x))$. Since $rr(\psi(\bar{x})|_P^{quan})$ can never be $\bot$, we have that $rr(\phi(\bar{x})|_P) = \bar{x} = \text{FREE}(\phi)$. Hence, $\phi(\bar{x})|_P$ is safe-range by definition.

4. $\phi(\bar{x}) = \psi(\bar{y}) \circ \varphi(\bar{z})$, where $\bar{x} = \bar{y} \cup \bar{z}$, $\circ$ stands for any of $\wedge$, $\vee$ and $\psi(\bar{y})|_P$ and $\varphi(\bar{z})|_P$ are both safe-range. Then $\phi(\bar{x})|_P := (\psi(\bar{y})|_P^{quan} \circ \varphi(\bar{z})|_P^{quan}) \wedge \bigwedge\limits_{x\in\bar{x}} P(x)$. Then $rr(\phi(\bar{x})|_P) = (rr(\psi(\bar{y})|_P^{quan}) \bullet rr(\varphi(\bar{z})|_P^{quan})) \cup$ $\cup \bigcup\limits_{x\in\bar{x}} rr(P(x))$, where $\bullet$ stands for $\cup$ when $\circ$ stands for $\wedge$, and $\bullet$ stands for $\cap$ when $\circ$ stands for $\vee$. Since none of the $rr(\psi(\bar{y})|_P^{quan})$, $rr(\varphi(\bar{z})|_P^{quan})$ is $\bot$, $rr(\phi(\bar{x})|_P) = \bar{x} = \text{FREE}(\phi)$. Hence, $\phi(\bar{x})|_P$ is safe-range by definition.

The theorem is proved. □

3.6.2 Auxiliary definitions, propositions and lemmas

Here we give some auxiliary definitions, propositions and lemmas that we need to prove the main theorem (Theorem 3.25).

Definition 3.30 (**ϕ-compatible interpretations**). Let ϕ be a formula from $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$. Two interpretations $\mathcal{I}_1 = \langle \Delta_1, \cdot^{\mathcal{I}_1} \rangle$ and $\mathcal{I}_2 = \langle \Delta_2, \cdot^{\mathcal{I}_2} \rangle$ are *ϕ-compatible* iff for any predicate P appearing in ϕ and constant c appearing in ϕ the following holds:

$$c^{\mathcal{I}_1} = c^{\mathcal{I}_2},$$

$$\bigcup_{\mathbf{i}\in Ind(P,\phi)} \{(a_{\mathbf{i}_1}, \ldots, a_{\mathbf{i}_n}) \mid (a_{\mathbf{i}_1}, \ldots, a_{\mathbf{i}_n}) \in P^{\mathcal{I}_1}\} =$$

$$= \bigcup_{\mathbf{i}\in Ind(P,\phi)} \{(a_{\mathbf{i}_1}, \ldots, a_{\mathbf{i}_n}) \mid (a_{\mathbf{i}_1}, \ldots, a_{\mathbf{i}_n}) \in P^{\mathcal{I}_2}\},$$

where $a_c := c^{\mathcal{I}_1} = c^{\mathcal{I}_2}$ for any $c \in \mathbb{C}$.

Note that any two compatible interpretations are always ϕ-compatible for any formula ϕ.

Proposition 3.31. Formula ϕ in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ is domain independent iff for any two ϕ-compatible interpretations $\mathcal{I}_1 = \langle \Delta_1, \cdot^{\mathcal{I}_1} \rangle$ and $\mathcal{I}_2 = \langle \Delta_2, \cdot^{\mathcal{I}_2} \rangle$, such that if $\mathcal{C} \neq \emptyset$, then $\Delta_1 \cap \Delta_2 \neq \emptyset$, we have:

$$(\phi)^{\mathcal{I}_1} = (\phi)^{\mathcal{I}_2}.$$

Proof. " $\Leftarrow$ " Let $\mathcal{I}_1 = \langle \Delta_1, \cdot^{\mathcal{I}_1} \rangle$ and $\mathcal{I}_2 = \langle \Delta_2, \cdot^{\mathcal{I}_2} \rangle$ be any two compatible interpretations. Then $\mathcal{I}_1$ and $\mathcal{I}_2$ are ϕ-compatible. Two cases are possible:

1. $\mathbb{C} = \emptyset$. By supposition we have, that $(\phi)^{\mathcal{I}_1} = (\phi)^{\mathcal{I}_2}$.

2. $\mathbb{C} \neq \emptyset$. Since $\mathcal{I}_1$ and $\mathcal{I}_2$ are compatible, for any constant c from $\mathbb{C}$ we have: $c^{\mathcal{I}_1} = c^{\mathcal{I}_2}$. Let $e := c^{\mathcal{I}_1} = c^{\mathcal{I}_2}$. Then it is evident that $e \in \Delta_1 \cap \Delta_2$. Hence, $\Delta_1 \cap \Delta_2 \neq \emptyset$. Again by supposition we have: $(\phi)^{\mathcal{I}_1} = (\phi)^{\mathcal{I}_2}$.

Therefore, ϕ is domain independent by definition.
" $\Rightarrow$ " Suppose, ϕ is domain independent. Suppose also that $\mathcal{I}_1 = \langle \Delta_1, \cdot^{\mathcal{I}_1} \rangle$ and $\mathcal{I}_2 = \langle \Delta_2, \cdot^{\mathcal{I}_2} \rangle$ are any two ϕ-compatible interpretations such that $\Delta_1 \cap \Delta_2 \neq \emptyset$ if the set of all constants $\mathbb{C}$ is not empty. We need to prove that $(\phi)^{\mathcal{I}_1} = (\phi)^{\mathcal{I}_2}$. Let us construct two interpretations $\mathcal{I}_1' = \langle \Delta_1, \cdot^{\mathcal{I}_1'} \rangle$ and $\mathcal{I}_2' = \langle \Delta_2, \cdot^{\mathcal{I}_2'} \rangle$ such that:

– for any $P \in \sigma(\phi) \cap \mathbb{P}$,
$P^{\mathcal{I}_1'} = \bigcup\limits_{\mathbf{i} \in Ind(P,\phi)} \{(a_{\mathbf{i}_1}, \ldots, a_{\mathbf{i}_n}) \mid (a_{\mathbf{i}_1}, \ldots, a_{\mathbf{i}_n}) \in P^{\mathcal{I}_1}\} = P^{\mathcal{I}_2'}$;

– for any $c \in \sigma(\phi) \cap \mathbb{C}$, $c^{\mathcal{I}_1'} = c^{\mathcal{I}_1} = c^{\mathcal{I}_2'}$;

– for any $P \in \mathbb{P} \setminus \sigma(\phi)$: $P^{\mathcal{I}_1'} = \emptyset = P^{\mathcal{I}_2'}$;

– if $\mathbb{C} \neq \emptyset$, then for any constant $c \in \mathbb{C} \setminus \sigma(\phi)$: $c^{\mathcal{I}_1'} = c^{\mathcal{I}_2'} = e$, where e is any element in a set $\Delta_1 \cap \Delta_2$, which is not empty by supposition.

By construction $\mathcal{I}_1'$ and $\mathcal{I}_2'$ are compatible. Since ϕ is domain independent,

$$(\phi)^{\mathcal{I}_1'} = (\phi)^{\mathcal{I}_2'}. \tag{3.7}$$

Let us consider interpretations $\mathcal{I}_1$ and $\mathcal{I}_1'$. They have the same domain Δ_1 and they are ϕ-compatible by definition. Then it is evident that for any substitution $\Theta : \mathbb{X} \mapsto \Delta_1$ we have: $\mathcal{I}_1, \Theta \models \phi$ iff $\mathcal{I}_1', \Theta \models \phi$. So, by definition of extension of a formula:

$$(\phi)^{\mathcal{I}_1} = (\phi)^{\mathcal{I}_1'}. \tag{3.8}$$

Similarly

$$(\phi)^{\mathcal{I}_2} = (\phi)^{\mathcal{I}_2'}. \tag{3.9}$$

So, it follows from (3.7), (3.8) and (3.9) that $(\phi)^{\mathcal{I}_1} = (\phi)^{\mathcal{I}_2}$. The proposition is proved completely.

The proposition is proved. □

Lemma 3.32. Let $\phi(\bar{x})$ be any (possibly closed) formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$, $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ be any model of $\mathcal{A}_\phi$, and $\mathcal{I}' = \langle Adom_\phi^{\mathcal{I}}, \cdot^{\mathcal{I}'} \rangle$ be any ϕ-compatible with $\mathcal{I}$ interpretation. Then

$$(\phi(\bar{x}))^{\mathcal{I}'} = (\phi(\bar{x})|_{Adom_\phi})^{\mathcal{I}} \tag{3.10}$$

Proof. First, we need to note that the statement of the lemma is well defined, because for any model $\mathcal{I}$ of $\mathcal{A}_\phi$ there always exists a ϕ-compatible interpretation $\mathcal{I}' = \langle Adom_\phi^{\mathcal{I}}, \cdot^{\mathcal{I}'} \rangle$ that can be constructed, for example, as follows:

- for any $P \in \sigma(\phi) \cap \mathbb{P}$,
  $P^{\mathcal{I}'} := \bigcup\limits_{\mathbf{i} \in Ind(P,\phi)} \{(a_{\mathbf{i}_1}, \ldots, a_{\mathbf{i}_n}) \mid (a_{\mathbf{i}_1}, \ldots, a_{\mathbf{i}_n}) \in P^{\mathcal{I}}\} \subseteq Adom_\phi^{\mathcal{I}}$,
  where $a_c := c^{\mathcal{I}}$ for any $c \in \mathbb{C}$;
- for any $c \in \sigma(\phi) \cap \mathbb{C}$, $c^{\mathcal{I}'} := c^{\mathcal{I}} \in Adom_\phi^{\mathcal{I}}$;
- for any $P \in \mathbb{P} \setminus \sigma(\phi)$: $P^{\mathcal{I}'} := \emptyset$;
- for any $c \in \mathbb{C} \setminus \sigma(\phi)$: $c^{\mathcal{I}'} := e$, where e is any element from $Adom_\phi^{\mathcal{I}}$, which is not empty because $\mathcal{I}$ is a model of $\mathcal{A}_\phi$.

We need to prove that

$$(\phi(\bar{x}))^{\mathcal{I}'} = (\phi(\bar{x})|^{quan}_{Adom_\phi} \wedge \bigwedge_{x \in \bar{x}} Adom_\phi(x))^{\mathcal{I}}$$

For any substitution Θ from $(\phi(\bar{x}))^{\mathcal{I}'}$ we have that $\Theta : \mathbb{X} \mapsto Adom_\phi^{\mathcal{I}}$, because $Adom_\phi^{\mathcal{I}}$ is domain of $\mathcal{I}'$. Hence, $\Theta \subseteq \bigcap\limits_{x \in \bar{x}} (Adom_\phi(x))^{\mathcal{I}}$.

We will prove the lemma by induction on the structure of formula.

1. **Base.**

   a) $\phi(\bar{x}) = P(\bar{c}, \bar{x})$ is an atomic formula, in which all constants from $\bar{c} = \{c_1, \ldots, c_k\}$ and all variables from $\bar{x} = \{x_1, \ldots, x_n\}$ occur at some places.
   "$\subseteq$" Let Θ be any substitution from $(P(\bar{c}, \bar{x}))^{\mathcal{I}'}$, i.e. $\mathcal{I}', \Theta \models P(\bar{c}, \bar{x})$, and let $a_i = \Theta(x_i)$, $i = 1, \ldots, n$; $\bar{a} = \{a_1, \ldots, a_n\}$; $\bar{c}^{\mathcal{I}'} = \{c_1^{\mathcal{I}'}, \ldots, c_k^{\mathcal{I}'}\}$. Then $(\bar{c}^{\mathcal{I}'}, \bar{a}) \in P^{\mathcal{I}'}$, where position of each $c_i^{\mathcal{I}'}$ in $(\bar{c}^{\mathcal{I}'}, \bar{a})$ corresponds to the position of c_i in $P(\bar{c}, \bar{x})$ and position of each a_j in $(\bar{c}^{\mathcal{I}'}, \bar{a})$ corresponds to the position of x_j in $P(\bar{c}, \bar{x})$. Then, since $\mathcal{I}'$ and $\mathcal{I}$ are ϕ-compatible, $(\bar{c}^{\mathcal{I}}, \bar{a}) \in P^{\mathcal{I}}$. Hence, $\mathcal{I}, \Theta \models P(\bar{c}, \bar{x})$. So, $\Theta \in (P(\bar{c}, \bar{x}))^{\mathcal{I}}$. Moreover, as we mentioned, $\Theta \in (Adom_\phi(x_1) \wedge \ldots \wedge Adom_\phi(x_n))^{\mathcal{I}}$. Then

$\Theta \in (P(\bar{c},\bar{x}))^{\mathcal{I}} \cap (Adom_\phi(x_1) \wedge \ldots \wedge Adom_\phi(x_n))^{\mathcal{I}} = (P(\bar{c},\bar{x}) \wedge Adom_\phi(x_1) \wedge \ldots \wedge Adom_\phi(x_n))^{\mathcal{I}} = (\phi(\bar{x})|_{Adom_\phi})^{\mathcal{I}}$.
"$\supseteq$" Let now Θ be any substitution from $(P(\bar{c},\bar{x}) \wedge Adom_\phi(x_1) \wedge \ldots \wedge Adom_\phi(x_n))^{\mathcal{I}} = (\phi(\bar{x})|_{Adom_\phi})^{\mathcal{I}}$ and such that $a_i = \Theta(x_i)$, $i = 1,\ldots,n$. To prove that $\Theta \in (P(\bar{c},\bar{x}))^{\mathcal{I}'}$ we basically need to follow the proof "$\subseteq$" in opposite direction.

b) $\phi(x) = (x = c)$, where c is a constant. Let Θ be any substitution from $(x = c)^{\mathcal{I}'}$ such that $\Theta(x) = a$. That is $\mathcal{I}', \Theta \models x = c$. Then $a = c^{\mathcal{I}'}$, and $a = c^{\mathcal{I}}$, because $\mathcal{I}'$ and $\mathcal{I}$ are ϕ-compatible. Then $\mathcal{I}, \Theta \models x = c$, that is $\Theta \in (x = c)^{\mathcal{I}}$ and as we mentioned $\Theta \in (Adom_\phi(x))^{\mathcal{I}}$. Then $\Theta \in (x = c)^{\mathcal{I}} \cap (Adom_\phi(x))^{\mathcal{I}} = ((x = c) \wedge Adom_\phi(x))^{\mathcal{I}} = (\phi(x)|_{Adom_\phi})^{\mathcal{I}}$.
Let now Θ be any substitution from $((x = c) \wedge Adom_\phi(x))^{\mathcal{I}}$ such that $\Theta(x) = a$. To prove that $\Theta \in (x = c)^{\mathcal{I}'}$ we basically need to follow the proof "$\subseteq$" in opposite direction.

c) Cases $\phi(x,y) = (x = y)$, and $\phi = (c_1 = c_2)$ (c_1 and c_2 are constants) can be proved similarly to the case (b).

2. $\phi(\bar{x},\bar{y},\bar{z}) = \psi(\bar{y},\bar{x}) \wedge \varphi(\bar{x},\bar{z})$, $\text{FREE}(\phi) = \bar{x} \cup \bar{y} \cup \bar{z}$, $\text{FREE}(\psi) = \bar{y} \cup \bar{x}$, $\text{FREE}(\varphi) = \bar{x} \cup \bar{z}$. Suppose that the lemma holds for the formulas ψ and φ,that is

– for any model $\mathcal{I}_1 = \langle \Delta_1, \cdot^{\mathcal{I}_1} \rangle$ of $\mathcal{A}_\psi$ and any ψ-compatible with $\mathcal{I}_1$ interpretation $\mathcal{I}' = \langle Adom_\psi^{\mathcal{I}_1}, \cdot^{\mathcal{I}'} \rangle$ we have:

$$(\psi(\bar{y},\bar{x}))^{\mathcal{I}'} = (\psi(\bar{y},\bar{x})|^{quan}_{Adom_\psi} \wedge \bigwedge_{y \in \bar{y}} Adom_\psi(y) \wedge \bigwedge_{x \in \bar{x}} Adom_\psi(x))^{\mathcal{I}_1} \quad (3.11)$$

– for any model $\mathcal{I}_2 = \langle \Delta_2, \cdot^{\mathcal{I}_2} \rangle$ of $\mathcal{A}_\varphi$ and any φ-compatible with $\mathcal{I}_2$ interpretation $\mathcal{I}'' = \langle Adom_\varphi^{\mathcal{I}_2}, \cdot^{\mathcal{I}''} \rangle$ we have:

$$(\varphi(\bar{x},\bar{z}))^{\mathcal{I}''} = (\varphi(\bar{x},\bar{z})|^{quan}_{Adom_\varphi} \wedge \bigwedge_{x \in \bar{x}} Adom_\varphi(x) \wedge \bigwedge_{z \in \bar{z}} Adom_\varphi(z))^{\mathcal{I}_2} \quad (3.12)$$

Let $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ be any model of $\mathcal{A}_\phi$ and let $\mathcal{I}^* = \langle Adom_\phi^{\mathcal{I}}, \cdot^{\mathcal{I}^*} \rangle$ be any ϕ-compatible with $\mathcal{I}$ interpretation. Let $\bar{\mathcal{I}} = \langle \Delta, \cdot^{\bar{\mathcal{I}}} \rangle$ be an interpretation such that $Adom_\psi^{\bar{\mathcal{I}}} = Adom_\varphi^{\bar{\mathcal{I}}} = Adom_\phi^{\mathcal{I}}$ and $\cdot^{\bar{\mathcal{I}}}$ interpret all the other predicates and constants as $\cdot^{\mathcal{I}}$. Then $\bar{\mathcal{I}}$ is a model of $\mathcal{A}_\phi$ and since $\sigma(\phi) = \sigma(\psi) \cup \sigma(\varphi)$,

$\bar{\mathcal{I}}$ is a model of $\mathcal{A}_\psi$ and $\mathcal{A}_\varphi$ by construction and

$$(\psi(\bar{y},\bar{x})|^{quan}_{Adom_\psi})^{\bar{\mathcal{I}}} = (\psi(\bar{y},\bar{x})|^{quan}_{Adom_\phi})^{\bar{\mathcal{I}}} = (\psi(\bar{y},\bar{x})|^{quan}_{Adom_\phi})^{\mathcal{I}},$$
$$(\varphi(\bar{x},\bar{z})|^{quan}_{Adom_\varphi})^{\bar{\mathcal{I}}} = (\varphi(\bar{x},\bar{z})|^{quan}_{Adom_\phi})^{\bar{\mathcal{I}}} = (\varphi(\bar{x},\bar{z})|^{quan}_{Adom_\phi})^{\mathcal{I}}.$$

Thus, by (3.11) we have:
$(\psi(\bar{y},\bar{x}))^{\mathcal{I}^*} = (\psi(\bar{y},\bar{x})|^{quan}_{Adom_\psi} \wedge \bigwedge\limits_{y\in\bar{y}} Adom_\psi(y) \wedge \bigwedge\limits_{x\in\bar{x}} Adom_\psi(x))^{\bar{\mathcal{I}}} =$
$= (\psi(\bar{y},\bar{x})|^{quan}_{Adom_\psi})^{\bar{\mathcal{I}}} \cap \bigcap\limits_{y\in\bar{y}} (Adom_\psi(y))^{\bar{\mathcal{I}}} \cap \bigcap\limits_{x\in\bar{x}} (Adom_\psi(x))^{\bar{\mathcal{I}}} =$
$= (\psi(\bar{y},\bar{x})|^{quan}_{Adom_\phi})^{\mathcal{I}} \cap \bigcap\limits_{y\in\bar{y}} (Adom_\phi(y))^{\mathcal{I}} \cap \bigcap\limits_{x\in\bar{x}} (Adom_\phi(x))^{\mathcal{I}}$
$= (\psi(\bar{y},\bar{x})|^{quan}_{Adom_\phi} \wedge \bigwedge\limits_{y\in\bar{y}} Adom_\phi(y) \wedge \bigwedge\limits_{x\in\bar{x}} Adom_\phi(x))^{\mathcal{I}}.$
Similarly, by (3.12) we have:
$(\varphi(\bar{x},\bar{z}))^{\mathcal{I}^*} = (\varphi(\bar{x},\bar{z})|^{quan}_{Adom_\phi} \wedge \bigwedge\limits_{x\in\bar{x}} Adom_\phi(x) \wedge \bigwedge\limits_{z\in\bar{z}} Adom_\phi(z))^{\mathcal{I}}.$
Then $(\phi(\bar{x},\bar{y},\bar{z})^{\mathcal{I}^*} = (\psi(\bar{y},\bar{x}) \wedge \varphi(\bar{x},\bar{z}))^{\mathcal{I}^*} = (\psi(\bar{y},\bar{x})^{\mathcal{I}^*} \cap (\varphi(\bar{x},\bar{z}))^{\mathcal{I}^*} =$
$= (\psi(\bar{y},\bar{x})|^{quan}_{Adom_\phi} \wedge \varphi(\bar{x},\bar{z})|^{quan}_{Adom_\phi} \wedge \bigwedge\limits_{y\in\bar{y}} Adom_\phi(y) \wedge \bigwedge\limits_{x\in\bar{x}} Adom_\phi(x) \wedge$
$\bigwedge\limits_{z\in\bar{z}} Adom_\phi(z))^{\mathcal{I}} = (\phi(\bar{x},\bar{y},\bar{z})|^{quan}_{Adom_\phi} \wedge \bigwedge\limits_{x\in\bar{x}\cup\bar{y}\cup\bar{z}} Adom_\phi(x))^{\mathcal{I}} =$
$(\phi(\bar{x},\bar{y},\bar{z})|_{Adom_\phi})^{\mathcal{I}}.$

The case is proved.

3. $\phi(\bar{x}) = \neg\psi(\bar{x})$. Suppose that the lemma holds for the formula $\psi(\bar{x})$. Let $\mathcal{I} = \langle\Delta, \cdot^{\mathcal{I}}\rangle$ be any model of $\mathcal{A}_\phi$ and $\mathcal{I}' = \langle Adom^{\mathcal{I}}_\phi, \cdot^{\mathcal{I}'}\rangle$ be any ϕ-compatible with $\mathcal{I}$ interpretation. Since $\sigma(\psi) = \sigma(\phi)$, we have: $\mathcal{I}$ is a model of $\mathcal{A}_\psi$ as well, $Adom^{\mathcal{I}}_\phi = Adom^{\mathcal{I}}_\psi$, and, hence, $(Adom_\phi(\bar{x}))^{\mathcal{I}} = (Adom_\psi(\bar{x}))^{\mathcal{I}}$, $(\psi(\bar{x})|^{quan}_{Adom_\phi})^{\mathcal{I}} = (\psi(\bar{x})|^{quan}_{Adom_\psi})^{\mathcal{I}}$. Then, since the lemma holds for $\psi(\bar{x})$, we have:

$$(\psi(\bar{x}))^{\mathcal{I}'} = (\psi(\bar{x})|^{quan}_{Adom_\psi} \wedge \bigwedge_{x\in\bar{x}} Adom_\psi(x))^{\mathcal{I}}.$$

Then $(\phi(\bar{x}))^{\mathcal{I}'} = (\neg\psi(\bar{x}))^{\mathcal{I}'} =$
$= \bigcap\limits_{x\in\bar{x}} (Adom_\phi(x))^{\mathcal{I}} \setminus (\psi(\bar{x}))^{\mathcal{I}'} = \bigcap\limits_{x\in\bar{x}} (Adom_\phi(x))^{\mathcal{I}} \setminus (\psi(\bar{x})|^{quan}_{Adom_\psi} \wedge$
$\wedge \bigwedge\limits_{x\in\bar{x}} Adom_\psi(x))^{\mathcal{I}} = \bigcap\limits_{x\in\bar{x}} (Adom_\phi(x))^{\mathcal{I}} \setminus ((\psi(\bar{x})|^{quan}_{Adom_\psi})^{\mathcal{I}} \cap$

$\cap \bigcap_{x\in\bar{x}}(Adom_{\psi}(x))^{\mathcal{I}} =$
$= \bigcap_{x\in\bar{x}}(Adom_{\phi}(x))^{\mathcal{I}} \setminus ((\psi(\bar{x})|^{quan}_{Adom_{\phi}})^{\mathcal{I}} \cap \bigcap_{x\in\bar{x}}(Adom_{\phi}(x))^{\mathcal{I}}) =$
$\bigcap_{x\in\bar{x}}(Adom_{\phi}(x))^{\mathcal{I}} \setminus (\psi(\bar{x})|^{quan}_{Adom_{\phi}})^{\mathcal{I}} = (\{\Theta \mid \Theta : \mathbb{X} \mapsto \Delta\}^n \setminus (\psi(\bar{x})|^{quan}_{Adom_{\phi}})^{\mathcal{I}})$
$\cap \bigcap_{x\in\bar{x}}(Adom_{\phi}(x))^{\mathcal{I}} = (\neg(\psi(\bar{x})|^{quan}_{Adom_{\phi}}))^{\mathcal{I}} \cap \bigcap_{x\in\bar{x}}(Adom_{\phi}(x))^{\mathcal{I}} =$
$= ((\neg\psi(\bar{x}))|^{quan}_{Adom_{\phi}} \wedge \bigwedge_{x\in\bar{x}} Adom_{\phi}(x))^{\mathcal{I}} = (\phi(\bar{x})|_{Adom_{\phi}})^{\mathcal{I}}.$

The case is proved.

4. $\phi(\bar{x},\bar{y},\bar{z}) = \psi(\bar{y},\bar{x}) \vee \varphi(\bar{x},\bar{z})$. Suppose that the lemma holds for the formulas $\psi(\bar{y},\bar{x})$ and $\varphi(\bar{x},\bar{z})$. Let $\mathcal{I} = \langle\Delta, \cdot^{\mathcal{I}}\rangle$ be any model of $\mathcal{A}_{\phi}$ and $\mathcal{I}' = \langle Adom^{\mathcal{I}}_{\phi}, \cdot^{\mathcal{I}'}\rangle$ be any ϕ-compatible with $\mathcal{I}$ interpretation. Then $(\phi(\bar{x},\bar{y},\bar{z}))^{\mathcal{I}'} = (\psi(\bar{y},\bar{x}) \vee \varphi(\bar{x},\bar{z})^{\mathcal{I}'} = (\neg(\neg\psi(\bar{y},\bar{x}) \wedge \neg\varphi(\bar{x},\bar{z}))^{\mathcal{I}'}$. By the cases that are proved already we have: $(\neg(\neg\psi(\bar{y},\bar{x}) \wedge \neg\varphi(\bar{x},\bar{z}))^{\mathcal{I}'} = ((\neg(\neg\psi(\bar{y},\bar{x}) \wedge \neg\varphi(\bar{x},\bar{z})))|_{Adom_{\phi}})^{\mathcal{I}} = ((\psi(\bar{y},\bar{x}) \vee \varphi(\bar{x},\bar{z}))|_{Adom_{\phi}})^{\mathcal{I}} =$
$= (\phi(\bar{x},\bar{y},\bar{z})|_{Adom_{\phi}})^{\mathcal{I}}$.

The case is proved.

5. $\phi(\bar{x},\bar{y},\bar{z}) = \psi(\bar{y},\bar{x}) \to \varphi(\bar{x},\bar{z})$. This case can be proved similarly to the case 4.

6. Let $\phi(\bar{x}) = \exists y.\, \psi(y,\bar{x})$, where $\text{FREE}(\phi(\bar{x})) = \bar{x}$, $\text{FREE}(\psi(y,\bar{x})) = y \cup \bar{x}$. Suppose that the lemma holds for the formula $\psi(y,\bar{x})$. Let $\mathcal{I} = \langle\Delta, \cdot^{\mathcal{I}}\rangle$ be any model of $\mathcal{A}_{\phi}$ and $\mathcal{I}' = \langle Adom^{\mathcal{I}}_{\phi}, \cdot^{\mathcal{I}'}\rangle$ be any ϕ-compatible with $\mathcal{I}$ interpretation. Since $\sigma(\psi) = \sigma(\phi)$, we have: $\mathcal{I}$ is a model of $\mathcal{A}_{\psi}$ as well, $\mathcal{I}'$ is ψ-compatible with $\mathcal{I}$, $Adom^{\mathcal{I}}_{\phi} = Adom^{\mathcal{I}}_{\psi}$, and, hence, $(Adom_{\phi}(x))^{\mathcal{I}} = (Adom_{\psi}(x))^{\mathcal{I}}$, $(\psi(\bar{x})|^{quan}_{Adom_{\phi}})^{\mathcal{I}} = (\psi(\bar{x})|^{quan}_{Adom_{\psi}})^{\mathcal{I}}$. Then, since the lemma holds for $\psi(y,\bar{x})$, we have:

$$(\psi(y,\bar{x}))^{\mathcal{I}'} = (\psi(y,\bar{x})|^{quan}_{Adom_{\phi}} \wedge Adom_{\phi}(y) \wedge \bigwedge_{x\in\bar{x}} Adom_{\phi}(\bar{x}))^{\mathcal{I}}.$$

"$\subseteq$" $\Theta \in (\phi(\bar{x}))^{\mathcal{I}'} = (\exists y.\, \psi(y,\bar{x}))^{\mathcal{I}'}$. Hence, there exists $a \in Adom^{\mathcal{I}}_{\phi}$ such that
$\mathcal{I}', \Theta[y \to a] \models \psi(y,\bar{x})$, where $\Theta[y \to a]$ is a substitution that is the same as

Θ except that it assigns element a to variable y. Then

$$\Theta[y \to a] \in (\psi(y, \bar{x}))^{\mathcal{I}'} = (\psi(y, \bar{x})|^{quan}_{Adom_\phi} \wedge Adom_\phi(y) \bigwedge_{x \in \bar{x}} Adom_\phi(\bar{x}))^{\mathcal{I}}.$$

Hence,

$$\Theta \in (\exists y.(Adom_\phi(y) \wedge \psi(y, \bar{x})|^{quan}_{Adom_\phi}) \bigwedge_{x \in \bar{x}} Adom_\phi(\bar{x}))^{\mathcal{I}} = (\phi(\bar{x})|_{Adom_\phi})^{\mathcal{I}}.$$

"$\supseteq$" Let $\Theta \in (\phi(\bar{x})|_{Adom_\phi})^{\mathcal{I}}$. By following the previous proof in opposite direction we can get that $\Theta \in (\phi(\bar{x}))^{\mathcal{I}'}$.

The case is proved.

7. $\phi(\bar{x}) = \forall y.\, \psi(y, \bar{x})$. Suppose that the lemma holds for the formula $\psi(y, \bar{x})$. Let $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ be any model of $\mathcal{A}_\phi$ and $\mathcal{I}' = \langle Adom^{\mathcal{I}}_\phi, \cdot^{\mathcal{I}'} \rangle$ be any ϕ-compatible with $\mathcal{I}$ interpretation. Since $\sigma(\psi) = \sigma(\phi)$, we have: $\mathcal{I}$ is a model of $\mathcal{A}_\psi$ as well, $\mathcal{I}'$ is ψ-compatible with $\mathcal{I}$, $Adom^{\mathcal{I}}_\phi = Adom^{\mathcal{I}}_\psi$, and, hence, $(Adom_\phi(x))^{\mathcal{I}} = (Adom_\psi(x))^{\mathcal{I}}$, $(\psi(\bar{x})|^{quan}_{Adom_\phi})^{\mathcal{I}} = (\psi(\bar{x})|^{quan}_{Adom_\psi})^{\mathcal{I}}$. Then taking into account the previous cases we have: $(\phi(\bar{x}))^{\mathcal{I}'} = (\forall y.\, \psi(y, \bar{x}))^{\mathcal{I}'} =$

$$= (\neg\exists y.\, \neg\psi(y, \bar{x}))^{\mathcal{I}'} = ((\neg\exists y.\, \neg\psi(y, \bar{x}))|^{quan}_{Adom_\psi} \wedge \bigwedge_{x \in \bar{x}} Adom_\psi(x))^{\mathcal{I}} =$$
$$= (\neg\exists y.\, (Adom_\psi(y) \wedge \neg(\psi(y, \bar{x}))|^{quan}_{Adom_\psi}) \wedge \bigwedge_{x \in \bar{x}} Adom_\psi(x))^{\mathcal{I}} =$$
$$= (\forall y.\, (\neg Adom_\phi(y) \vee \psi(y, \bar{x})|^{quan}_{Adom_\phi}) \wedge \bigwedge_{x \in \bar{x}} Adom_\phi(x))^{\mathcal{I}} =$$
$$= (\forall y.\, (Adom_\phi(y) \to \psi(y, \bar{x})|^{quan}_{Adom_\phi}) \wedge \bigwedge_{x \in \bar{x}} Adom_\phi(x))^{\mathcal{I}} = (\phi(\bar{x})|_{Adom_\phi})^{\mathcal{I}}.$$

The case is proved.

The lemma is proved. □

3.6.3 Proof of Theorem 3.25

Proof. "⇒" Let $\phi(\bar{x})$ be domain independent. $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ is a model of $\mathcal{A}_\phi$ and $\mathcal{I}' =$
$\langle Adom^{\mathcal{I}}_\phi, \cdot^{\mathcal{I}'} \rangle$ be any ϕ-compatible with $\mathcal{I}$ (it exists as we mentioned in the previous proof). Then $\emptyset \neq Adom^{\mathcal{I}}_\phi \subseteq \Delta$. Hence, since $\phi(\bar{x})$ is domain independent, by Proposition 3.31 we have: $(\phi(\bar{x}))^{\mathcal{I}} = (\phi(\bar{x}))^{\mathcal{I}'}$. By Lemma 3.32, $(\phi(\bar{x}))^{\mathcal{I}'} = (\phi(\bar{x})|_{Adom_\phi})^{\mathcal{I}}$. Then $(\phi(\bar{x}))^{\mathcal{I}} = (\phi(\bar{x})|_{Adom_\phi})^{\mathcal{I}}$. It means that for any substitution Θ, $\mathcal{I}, \Theta \models \phi(\bar{x})$ iff $\mathcal{I}, \Theta \models \phi(\bar{x})|_{Adom_\phi}$. That is,

$\mathcal{I}, \Theta \models \phi(\bar{x}) \leftrightarrow \phi(\bar{x})|_{Adom_\phi}$. That is $\mathcal{I} \models \forall \bar{x}.\, \phi(\bar{x}) \leftrightarrow \phi(\bar{x})|_{Adom_\phi}$. Recall that $\mathcal{I}$ is any model of $\mathcal{A}_\phi$. Then $\mathcal{A}_\phi \models \forall \bar{x}.\, \phi(\bar{x}) \leftrightarrow \phi(\bar{x})|_{Adom_\phi}$.
"$\Leftarrow$" Suppose, that $\mathcal{A}_\phi \models \forall \bar{x}.\, \phi(\bar{x}) \leftrightarrow \phi(\bar{x})|_{Adom_\phi}$. Let us prove that $\phi(\bar{x})$ is domain independent.
Let $\mathcal{I}_1 = \langle \Delta_1, \cdot^{\mathcal{I}_1} \rangle$ be any interpretation. Consider interpretation $\mathcal{I}^* = \langle \Delta_1, \cdot^{\mathcal{I}^*} \rangle$ such that the interpretation function $\cdot^{\mathcal{I}^*}$ assigns everything exactly as $\cdot^{\mathcal{I}_1}$ except the interpretation of the predicate $Adom_\phi$. Let $Adom_\phi^{\mathcal{I}^*}$ be any non empty set containing $adom(\phi, \mathcal{I}_1)$. Then it is easy to see that $\mathcal{I}^*$ is a model of $\mathcal{A}_\phi$. Interpretations $\mathcal{I}_1$ and $\mathcal{I}^*$ have the same domain Δ_1 and they interpret all the predicates (except $Adom_\phi$) and constants in the same way. Then, evidently, $(\phi(\bar{x}))^{\mathcal{I}_1} = (\phi(\bar{x}))^{\mathcal{I}^*}$ (because $Adom_\phi \notin \sigma(\phi)$).
Since $\mathcal{I}^*$ is a model of $\mathcal{A}_\phi$, by supposition we have:

$$\mathcal{I}^* \models \forall \bar{x}.\, \phi(\bar{x}) \leftrightarrow \phi(\bar{x})|_{Adom_\phi}.$$

That is, for any $\Theta : \mathbb{X} \mapsto \Delta$ we have: $\mathcal{I}^*, \Theta \models \phi(\bar{x}) \leftrightarrow \phi(\bar{x})|_{Adom_\phi}$. That is $(\phi(\bar{x}))^{\mathcal{I}^*} = (\phi(\bar{x})|_{Adom_\phi})^{\mathcal{I}^*}$. By Lemma 3.32 we have: $(\phi(\bar{x})|_{Adom_\phi})^{\mathcal{I}^*} = (\phi(\bar{x}))^{\mathcal{I}'}$, where $\mathcal{I}' = \langle Adom_\phi^{\mathcal{I}^*}, \cdot^{\mathcal{I}'} \rangle$ is any ϕ-compatible with $\mathcal{I}^*$ and $\mathcal{I}_1$ interpretation. Summing up we have: $(\phi(\bar{x}))^{\mathcal{I}_1} = (\phi(\bar{x}))^{\mathcal{I}^*} = (\phi(\bar{x})|_{Adom_\phi})^{\mathcal{I}^*} = (\phi(\bar{x}))^{\mathcal{I}'}$.
Together with $\mathcal{I}_1$ let us consider any other ϕ-compatible with $\mathcal{I}_1$ interpretation $\mathcal{I}_2 = \langle \Delta_2, \cdot^{\mathcal{I}_2} \rangle$. Then similarly $(\phi(\bar{x}))^{\mathcal{I}_2} = (\phi(\bar{x}))^{\mathcal{I}'}$. Hence, $(\phi(\bar{x}))^{\mathcal{I}_1} = (\phi(\bar{x}))^{\mathcal{I}'} = (\phi(\bar{x}))^{\mathcal{I}_2}$. Then by Proposition 3.31 formula $\phi(\bar{x})$ is domain independent.

The theorem is proved. □

3.6.4 Proof of Proposition 3.28

Proof. $\alpha_\phi = \bigwedge\limits_{P \in \sigma(\phi)} \bigwedge\limits_{\mathbf{i} \in Ind(P,\phi)} (\forall x_1, \ldots, x_{max(\mathbf{i})}.\, P(x_{\mathbf{i}_1}, \ldots, x_{\mathbf{i}_{\text{AR}(P)}}) \rightarrow$

$\rightarrow \bigwedge\limits_{j=1}^{max(\mathbf{i})} Adom_\phi(x_j)) \wedge \bigwedge\limits_{c \in \sigma(\phi) \cap \mathbb{C}} Adom_\phi(c) \wedge \exists x_1.\, Adom_\phi(x_1)$. Let $\mathbf{m}(\phi)$ be the maximal number in the set $\{max(\mathbf{i}) \mid \mathbf{i} \in Ind(P, \phi),\, P \in \sigma(\phi)\}$. Then one can easily check that

$\alpha_\phi \equiv (\forall x_1, \ldots, x_{\mathbf{m}(\phi)}.\, \bigvee\limits_{P \in \sigma(\phi)} \bigvee\limits_{\mathbf{i} \in Ind(P,\phi)} P(x_{\mathbf{i}_1}, \ldots, x_{\mathbf{i}_{\text{AR}(P)}}) \rightarrow$

$\rightarrow \bigwedge\limits_{j=1}^{\mathbf{m}(\phi)} Adom_\phi(x_j)) \wedge \bigwedge\limits_{c \in \sigma(\phi) \cap \mathbb{C}} Adom_\phi(c) \wedge \exists w.\, Adom_\phi(w)$.

Suppose $\phi = \exists \bar{y}.\, \varphi(\bar{y}) \in [\exists^*, all, (0)]^{\mathbb{C}}_{=}$, where $\bar{y} = y_1, \ldots, y_n$ and $\varphi(\bar{y})$ is quantifier free. Then $\phi \leftrightarrow \phi|_{Adom_\phi} \equiv (\neg\phi \vee \phi|_{Adom_\phi}) \wedge (\phi \vee \neg\phi|_{Adom_\phi}) \equiv$

$\equiv (\neg\exists \bar{y}.\, \varphi(\bar{y}) \vee (\exists \bar{y}.\, \varphi(\bar{y}) \wedge \bigwedge\limits_{y \in \bar{y}} Adom_\phi(y))) \wedge (\exists \bar{y}.\, \varphi(\bar{y}) \vee (\neg\exists \bar{y}.\, \varphi(\bar{y}) \wedge$

$\wedge \bigwedge\limits_{y \in \bar{y}} Adom_\phi(y))) \equiv (\forall \bar{y}.\, \neg\varphi(\bar{y}) \vee (\exists \bar{z}.\, \varphi(\bar{z}) \wedge \bigwedge\limits_{z \in \bar{z}} Adom_\phi(z))) \wedge$

$\wedge (\exists \bar{u}.\, \varphi(\bar{u}) \vee \forall \bar{v}.\, \neg(\varphi(\bar{v}) \wedge \bigwedge\limits_{v \in \bar{v}} Adom_\phi(v))) \equiv \exists \bar{z}. \exists \bar{u}. \forall \bar{y}. \forall \bar{v}.\, (\neg\varphi(\bar{y}) \vee$

$\vee (\varphi(\bar{z}) \wedge \bigwedge\limits_{z \in \bar{z}} Adom_\phi(z))) \wedge (\varphi(\bar{u}) \vee \neg(\varphi(\bar{v}) \wedge \bigwedge\limits_{v \in \bar{v}} Adom_\phi(v))).$

Then $\alpha_\phi \rightarrow (\phi \leftrightarrow \phi|_{Adom_\phi}) \equiv \neg\alpha_\phi \vee (\phi \leftrightarrow \phi|_{Adom_\phi}) \equiv$

$\equiv \bigvee\limits_{c \in \sigma(\phi) \cap \mathbb{C}} \neg Adom_\phi(c) \vee \forall w.\, \neg Adom_\phi(w) \vee$

$\vee \exists x_1, \ldots, x_{\mathbf{m}(\phi)}. \neg(\bigvee\limits_{P \in \sigma(\phi)} \bigvee\limits_{\mathbf{i} \in Ind(P,\phi)} P(x_{\mathbf{i}_1}, \ldots, x_{\mathbf{i}_{\text{AR}(P)}}) \rightarrow$

$\rightarrow \bigwedge\limits_{j=1}^{\mathbf{m}(\phi)} Adom_\phi(x_j)) \vee \exists \bar{z}. \exists \bar{u}. \forall \bar{y}. \forall \bar{v}.\, (\neg\varphi(\bar{y}) \vee (\varphi(\bar{z}) \wedge \bigwedge\limits_{z \in \bar{z}} Adom_\phi(z))) \wedge$

$\wedge (\varphi(\bar{u}) \vee \neg(\varphi(\bar{v}) \wedge \bigwedge\limits_{v \in \bar{v}} Adom_\phi(v))) \equiv$

$\equiv \exists x_1, \ldots, x_{\mathbf{m}(\phi)}. \exists \bar{z}. \exists \bar{u}. \forall w. \forall \bar{y}. \forall \bar{v}.\, (\neg(\bigvee\limits_{P \in \sigma(\phi)} \bigvee\limits_{\mathbf{i} \in Ind(P,\phi)} P(x_{\mathbf{i}_1}, \ldots, x_{\mathbf{i}_{\text{AR}(P)}}) \rightarrow$

$\rightarrow \bigwedge\limits_{j=1}^{\mathbf{m}(\phi)} Adom_\phi(x_j)) \vee ((\neg\varphi(\bar{y}) \vee (\varphi(\bar{z}) \wedge \bigwedge\limits_{z \in \bar{z}} Adom_\phi(z))) \wedge (\varphi(\bar{u}) \vee$

$\vee \neg(\varphi(\bar{v}) \wedge \bigwedge\limits_{v \in \bar{v}} Adom_\phi(v)))) \vee \bigvee\limits_{c \in \sigma(\phi) \cap \mathbb{C}} \neg Adom_\phi(c) \vee \forall w.\, \neg Adom_\phi(w)) \in$

$\in [\exists^* \forall^*, all, (0)]^{\mathbb{C}}_{=}.$

The proposition is proved. □

4. Domain Independent Fragments of Description Logics

A domain independent fragment of first-order logic is in general undecidable (i.e. determining satisfiability of a domain independent formula is undecidable). Later we will consider description logics as possible applications of our query rewriting framework. Since, as we mentioned already, domain independence is a desirable property for a reformulation, it makes sense to study domain independent fragments of Description Logics. In this chapter we consider domain independent fragments of $\mathcal{SHOIQ}$, $\mathcal{SHOQ}$ and $\mathcal{ALCHOI}$ and prove variants of Codd's theorem for them.

We show that domain independent fragment of $\mathcal{SHOIQ}$ does not have the finite model property, as the logic itself, while domain independent fragments of both $\mathcal{SHOQ}$ and $\mathcal{ALCHOI}$ have this property. The first one inherits it from $\mathcal{SHOQ}$ itself. The second one – from the guarded negation first-order logic, because it turned out that all domain independent axioms and concepts in $\mathcal{ALCHOI}$ can be expressed in GNFO, which has the finite model property (Bárány et al., 2011).

We also give recursive definitions of domain independent concepts of $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$ and thus provide convenient syntactic characterizations of domain independent fragments of these logics.

As usual, a description logic has two main components: a *TBox* and an *ABox*. The first one consists of a finite set of a *concept inclusion axioms* of the form $C \sqsubseteq D$ and/or *role inclusion axioms* of the form $R \sqsubseteq S$, where C, D are *concepts* and R, S are *roles* defined according to the grammar of this logic. All these axioms that can potentially appear in a TBox are called *TBox axioms*. Each concept and role has a first-order logic translation, i.e. has an equivalent first-order logic formula. Here, we will use mainly the first component – TBox.

A description logic $\mathcal{L}$ is said to have the *finite model property* if for every $\mathcal{L}$-concept C and every $\mathcal{L}$-TBox $\mathcal{T}$, if C is satisfiable with respect to $\mathcal{T}$ then there exists some finite interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathcal{T}$ and $C^{\mathcal{I}} \neq \emptyset$.

We say that a fragment $\mathcal{L}_1$ of some description logic is *equal* to a fragment $\mathcal{L}_2$ of another (possibly the same one) description logic if

– the set of TBox axioms of $\mathcal{L}_1$ is equal to the set of TBox axioms of $\mathcal{L}_2$;

– the set of concepts of $\mathcal{L}_1$ is equal to the set of concepts of $\mathcal{L}_2$.

We say that a fragment $\mathcal{L}_1$ of some description logic is *equally expressive* to a fragment $\mathcal{L}_2$ of another (possibly the same one) description logic if

– for any TBox axiom in $\mathcal{L}_1$ there exists a logically equivalent TBox axiom in $\mathcal{L}_2$ and vice versa;

– for any concept in $\mathcal{L}_1$ there exists a logically equivalent concept in $\mathcal{L}_2$ and vice versa.

For any description logic $\mathcal{L}$, which is a fragment of $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$, we denote a first-order translation of a concept C to $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ as $C(x)$.
We call any axiom (concept) in some description logic safe-range (domain independent), if the corresponding first-order translation of the axiom (concept) is safe-range (domain independent) formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$. In particular,

– we call any concept in some description logic safe-range (domain independent), if the corresponding first-order translation of the concept is safe- range (domain independent) formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$;

– we call any concept inclusion axiom $C \sqsubseteq D$ safe-range (domain independent), if the corresponding first-order translation $\neg\exists x.\, C(x) \wedge \neg D(x)$ is safe-range (domain independent) formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$, where $C(x)$ and $D(x)$ are first-order translations of the concepts C and D respectively;

– we call any role inclusion axiom $S \sqsubseteq R$ safe-range (domain independent), if the corresponding first-order translation $\neg\exists x, y.\, S(x, y) \wedge \neg R(x, y)$ is safe-range (domain independent) formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$, where formulas $S(x, y)$ and $R(x, y)$ are first-order translations of the roles S and R respectively;

– we call any transitivity axiom $Trans(R)$ safe-range (domain independent), if the corresponding first-order translation $\neg\exists x, y, z.\, R(x, y) \wedge R(y, z) \wedge \neg R(x, z)$ is safe-range (domain independent) formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$, where $S(x, y)$ and $R(x, y)$ are first-order translations of the roles S and R respectively.

Safe-range (domain independent) fragment of some description logic is a set of safe-range (domain independent) concepts and safe-range (domain independent) TBox axioms in this logic.

4.1 Domain Independent Fragment of $\mathcal{SHOIQ}$

$\mathcal{SHOIQ}$ is an extension of the description logic $\mathcal{ALC}$ with role hierarchies ($\mathcal{H}$), transitive roles ($\mathcal{S}$), individuals ($\mathcal{O}$), inverse roles ($\mathcal{I}$), and qualified cardinality restrictions ($\mathcal{Q}$); it is the logic at the basis of OWL. The syntax and semantics of $\mathcal{SHOIQ}$ concept and role expressions are summarised in the Table 4.1, 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 standard first-order logic translation of the $\mathcal{SHOIQ}$ concepts and roles is given in the Table 4.2, where π_x is a mapping that maps each $\mathcal{SHOIQ}$ concept to a corresponding first-order logic formula with one free variable x and $\pi_{x,y}$ is a mapping that maps each $\mathcal{SHOIQ}$ role to a corresponding first-order logic formula with two free variables x and y. A TBox in $\mathcal{SHOIQ}$ is a set of concept inclusion axioms $C \sqsubseteq D$, role inclusion axioms $R \sqsubseteq S$ and transitivity axioms $Trans(R)$ (where C, D are concepts and R, S are roles) with the usual description logics semantics.

Table 4.1 – Syntax and semantics of $\mathcal{SHOIQ}$ concepts and roles

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

Table 4.2 – The standard inductive first-order logic translation of $\mathcal{SHOIQ}$ concepts and roles

$\mathcal{SHOIQ}$	First-order logic
$\pi_x(A)$	$A(x)$
$\pi_{x,y}(P)$	$P(x,y)$
$\pi_{x,y}(P^-)$	$P(y,x)$
$\pi_x(C \sqcap D)$	$\pi_x(C) \wedge \pi_x(D)$
$\pi_x(C \sqcup D)$	$\pi_x(C) \vee \pi_x(D)$
$\pi_x(\neg C)$	$\neg \pi_x(C)$
$\pi_x(\{o\})$	$x = o$
$\pi_x(\geq nR)$	$\exists x_1, \ldots, x_n . \pi_{x,x_1}(R) \wedge \ldots \wedge \pi_{x,x_n}(R) \wedge$ $\wedge (x_1 \neq x_2) \wedge \ldots \wedge (x_{n-1} \neq x_n)$
$\pi_x(\leq nR)$	$\forall x_1, \ldots, x_{n+1} . \pi_{x,x_1}(R) \wedge \ldots \wedge \pi_{x,x_{n+1}}(R) \rightarrow$ $\rightarrow (x_1 = x_2) \vee \ldots \vee (x_n = x_{n+1})$
$\pi_x(\geq nR.C)$	$\exists x_1, \ldots, x_n . \pi_{x,x_1}(R) \wedge \ldots \wedge \pi_{x,x_n}(R) \wedge$ $\wedge \pi_{x_1}(C) \wedge \ldots \wedge \pi_{x_n}(C) \wedge (x_1 \neq x_2) \wedge \ldots \wedge (x_{n-1} \neq x_n)$
$\pi_x(\leq nR.C)$	$\forall x_1, \ldots, x_{n+1} . \pi_{x,x_1}(R) \wedge \ldots \wedge \pi_{x,x_{n+1}}(R) \wedge$ $\wedge \pi_{x_1}(C) \wedge \ldots \wedge \pi_{x_{n+1}}(C) \rightarrow (x_1 = x_2) \vee \ldots \vee (x_n = x_{n+1})$

In this section we reveal a connection between domain independent and safe-range fragments of $\mathcal{SHOIQ}$. Concept inclusion axioms in $\mathcal{SHOIQ}$ ontologies may not be safe-range: for example, the axiom $\neg\, \texttt{male} \sqsubseteq \texttt{female}$ is not safe-range, because the corresponding first-order logic formula $\neg \exists x . \neg \texttt{male}(x) \wedge \neg \texttt{female}(x)$ 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 $\neg D(x)$ is not safe-range: just observe the logically equivalent first-order translation $\neg \exists x . C(x) \wedge \neg D(x)$ (which is actually in a safe-range normal form).
The following proposition provides recursive rules deciding whether an $\mathcal{SHOIQ}$ concept is safe-range.

Proposition 4.1. Let A be an atomic concept, let C and D be $\mathcal{SHOIQ}$ 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.

Corollary 4.2. A $\mathcal{SHOIQ}$ concept inclusion axiom $C \sqsubseteq D$ is safe-range if and only if C is safe-range or D is not safe-range.

It turned out that the description logic has such strict syntax that the safe-range fragment of it is not only equally expressive, but even ***equal*** to the domain independent fragment whenever we consider satisfiable axioms.
Let $\mathcal{SHOIQ}^*$ be a fragment of $\mathcal{SHOIQ}$, where only satisfiable TBox axioms are allowed. Then the following theorem takes place.

Theorem 4.3. The domain independent fragment of $\mathcal{SHOIQ}^*$ is **equal** to safe-range fragment of $\mathcal{SHOIQ}^*$.

Note that the Theorem 4.3 holds only for a fragment, where only satisfiable concept inclusion axioms are allowed as TBox assertions, that is in the fragment $\mathcal{SHOIQ}^*$. There are unsatisfiable domain independent concept inclusion axioms in $\mathcal{SHOIQ}$ which are not safe-range. For example, $A \sqcup \neg A \sqsubseteq A \sqcap \neg A$ is one of them. Below we prove that for such axioms there exist logically equivalent safe-range concept inclusion axioms in $\mathcal{SHOIQ}$. That is, safe-range and domain independent fragments of $\mathcal{SHOIQ}$ are equally expressive. In order to prove this we need to assume that the set of constants (individual names) $\mathbb{C}$ is not empty.

Theorem 4.4 (**Version of Codd's theorem**). If $\mathbb{C} \neq \emptyset$, then domain independent fragment of $\mathcal{SHOIQ}$ is equally expressive to safe-range fragment of $\mathcal{SHOIQ}$.

Unfortunately, $\mathcal{SHOIQ}$ does not have finite model property. Neither does its domain independent fragment.

Example 4.5. Let $\mathcal{T}$ be a TBox in $\mathcal{SHOIQ}$:

$$\begin{aligned}\mathcal{T} = \{&A \sqsubseteq \exists R,\\ &\exists R^- \sqsubseteq A,\\ &B \sqsubseteq \exists R,\\ &B \sqsubseteq \neg A,\\ &\top \sqsubseteq \leq 1R^-\}.\end{aligned}$$

This TBox may be rewritten as $\{A \sqsubseteq \geq 1R,\ \geq 1R^- \sqsubseteq A,\ B \sqsubseteq \geq 1R,\ B \sqsubseteq \neg A,$ $A \sqcup \neg A \sqsubseteq \neg \geq 2R^-\}$ so that we can really see that it is expressed in $\mathcal{SHOIQ}$.

Then, by using Corollary 4.2 and Proposition 4.1 one can see that $\mathcal{T}$ is safe-range. $\mathcal{T}$ is also satisfiable, and in any model $\mathcal{I}$ of $\mathcal{T}$ such that $B^{\mathcal{I}} \neq \emptyset$ we have an infinite number of instances of A, each one connected by R to the next one. Hence, safe-range (domain independent) fragment of $\mathcal{SHOIQ}$ does not have finite model property.

Note that in the example above it is even enough to consider logic $\mathcal{ALCFI}$ – a strict sublogic of $\mathcal{SHOIQ}$.

4.2 Syntactic Characterizations of Domain Independent Fragments of $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$

Each of the description logics $\mathcal{SHOQ}$ and $\mathcal{ALCHOI}$ is a sublogic of the description logic $\mathcal{SHOIQ}$. $\mathcal{SHOQ}$ corresponds to the $\mathcal{SHOIQ}$ without inverse roles (Table 4.3). For more details see e.g. Horrocks and Sattler (2001). Similarly, $\mathcal{SHOIQ}$ without transitivity axioms and cardinality restrictions forms the description logic $\mathcal{ALCHOI}$ (Table 4.4). Hence, the following propositions and theorems immediately follow from the Proposition 4.1, Theorem 4.3 and Theorem 4.4.

Table 4.3 – Syntax and semantics of $\mathcal{SHOQ}$ concepts and roles

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

Table 4.4 – Syntax and semantics of $\mathcal{ALCHOI}$ concepts and roles

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^-	$\{(y,x) \mid (x,y) \in P^{\mathcal{I}}\}$
$\neg C$	$\Delta^{\mathcal{I}} \backslash C^{\mathcal{I}}$
$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$\exists R$	$\{x \mid \{y \mid (x,y) \in R^{\mathcal{I}}\} \neq \emptyset\}$
$\exists R.C$	$\{x \mid \{y \mid (x,y) \in R^{\mathcal{I}}\} \cap C^{\mathcal{I}} \neq \emptyset\}$
$\forall R.C$	$\{x \mid$ if $(x,y) \in R^{\mathcal{I}}$ then $y \in C^{\mathcal{I}}\}$

Proposition 4.6. Let A be an atomic concept, let C and D be $\mathcal{ALCHOI}$ concepts, and let R be either an atomic role or an inverse atomic role. Then:

1. $A, \{o\}, \exists R, \exists R.C$ 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.

Proposition 4.7. Let A be an atomic concept, let C and D be $\mathcal{SHOQ}$ concepts, and let R be an 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.

Let $\mathcal{ALCHOI}^*$ and $\mathcal{SHOQ}^*$ be, respectively, fragments of $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$, where only satisfiable TBox assertions are allowed. Let us assume hereafter in this chapter that $\mathbb{C} \neq \emptyset$. Then by Theorem 4.3 and Theorem 4.4, since

$\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$ are subfragments of $\mathcal{SHOIQ}$, the following theorems take place:

Theorem 4.8. Domain independent fragment of $\mathcal{ALCHOI}^*$ is **equal** to safe-range fragment of $\mathcal{ALCHOI}^*$.

Theorem 4.9. Domain independent fragment of $\mathcal{SHOQ}^*$ is **equal** to safe-range fragment of $\mathcal{SHOQ}^*$.

Theorem 4.10 (**Version of Codd's theorem**)**.** Domain independent fragment of $\mathcal{ALCHOI}$ is equally expressive to safe-range fragment of $\mathcal{ALCHOI}$.

Theorem 4.11 (**Version of Codd's theorem**)**.** Domain independent fragment of $\mathcal{SHOQ}$ is equally expressive to safe-range fragment of $\mathcal{SHOQ}$.

4.2.1 Guarded negation fragment of $\mathcal{ALCHOI}$

Recall the definition of *guarded negation first-order logic (GNFO)* given in Subsection 3.2.2. Now we consider $\mathcal{ALCHOI}_{GN}$ – a guarded negation fragment of $\mathcal{ALCHOI}$ (i.e. an intersection of GNFO and $\mathcal{ALCHOI}$), which is defined as follows. We say that

– a concept C is an $\mathcal{ALCHOI}_{GN}$ concept if C is an $\mathcal{ALCHOI}$ concept and the corresponding first-order logic translation $C(x)$ is expressed in GNFO;

– a concept inclusion axiom $C \sqsubseteq D$ is an $\mathcal{ALCHOI}_{GN}$ concept inclusion axiom if C and D are $\mathcal{ALCHOI}$ concepts and 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 (here $x = x$ is not allowed to be a guard, because such formulas cannot be expressed in description logic);

– a role inclusion axiom $S \sqsubseteq R$ is an $\mathcal{ALCHOI}_{GN}$ 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.

A guarded negation fragment of $\mathcal{ALCHOI}$ is a set of $\mathcal{ALCHOI}_{GN}$ concepts and $\mathcal{ALCHOI}_{GN}$ TBox axioms.
From the definition of GNFO and $\mathcal{ALCHOI}$ it follows that the complex concept C of the logic $\mathcal{ALCHOI}_{GN}$ is recursively defined as follows:

$$B ::= \; A \mid \{o\} \mid \exists R$$

$$C ::= \; B \mid \exists R.C \mid \exists R.\neg C \mid B \sqcap \neg C \mid C \sqcap D \mid C \sqcup D \tag{4.1}$$

where A is an atomic concept, R is an atomic role or an inverse atomic role, and C and D are $\mathcal{ALCHOI}_{GN}$ concepts (possibly complex).
Strictly speaking, $\exists R \sqcap \neg C$ is not in GNFO. Indeed, the formula $(\exists y.\, R(x,y)) \wedge \neg C(x)$ 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: $\exists y.(R(x,y)) \wedge \neg C(x))$. So, we can assume that the formula $\exists R \sqcap \neg C$ is in $\mathcal{ALCHOI}_{GN}$.
Since GNFO has finite model property, $\mathcal{ALCHOI}_{GN}$ also has finite model property as a subfragment.

Proposition 4.12. All $\mathcal{ALCHOI}_{GN}$ concepts are safe-range.

Proposition 4.13. Any safe-range $\mathcal{ALCHOI}$ concept is equivalent to some $\mathcal{ALCHOI}_{GN}$ concept.

Proposition 4.14. All $\mathcal{ALCHOI}_{GN}$ TBox axioms are safe-range.

Proposition 4.15. Any safe-range $\mathcal{ALCHOI}$ TBox axiom is equivalent to some $\mathcal{ALCHOI}_{GN}$ TBox axiom.

Then the following connection between the safe-range fragment of $\mathcal{ALCHOI}$ and $\mathcal{ALCHOI}_{GN}$ takes place.

Theorem 4.16. Safe-range fragment of $\mathcal{ALCHOI}$ and $\mathcal{ALCHOI}_{GN}$ are equally expressive.

Theorem 4.10 and Theorem 4.16 imply the following theorem.

Theorem 4.17 (**Expressive power equivalence**). Domain independent fragment of $\mathcal{ALCHOI}$ and $\mathcal{ALCHOI}_{GN}$ are equally expressive.

4.2.2 Extended guarded negation fragment of $\mathcal{SHOQ}$

Here we introduce a new subfragment $\mathcal{SHOQ}_{GN^+}$ of $\mathcal{SHOQ}$.
Any concept C of $\mathcal{SHOQ}_{GN^+}$ is recursively defined as follows:

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

$$C ::= \; B \mid \geq nR.C \mid \geq nR.\neg C \mid B \sqcap \neg C \mid C \sqcap D \mid C \sqcup D \qquad (4.2)$$

where A is an atomic concept, R is an atomic role, and C and D are $\mathcal{SHOQ}_{GN^+}$ concepts (possibly complex). We say that

– a role inclusion axiom is in $\mathcal{SHOQ}_{GN^+}$ if it is a $\mathcal{SHOQ}$ role inclusion axiom;

– a transitivity axiom is in $\mathcal{SHOQ}_{GN^+}$ if it is a $\mathcal{SHOQ}$ transitivity inclusion axiom;

– A $\mathcal{SHOQ}$ concept inclusion axiom $C \sqsubseteq D$ is a $\mathcal{SHOQ}_{GN^+}$ concept inclusion axiom if C is a $\mathcal{SHOQ}_{GN^+}$ concept or $SimplePushNeg(\neg D)$ is a $\mathcal{SHOQ}_{GN^+}$ concept, where the function $SimplePushNeg$ recursively replaces $\neg\neg D$ by D. This $SimplePushNeg$ transformation is evidently equivalence-preserving.

The fragment $\mathcal{SHOQ}_{GN^+}$ is a set of $\mathcal{SHOQ}_{GN^+}$ concepts (defined in (4.2)) and $\mathcal{SHOQ}_{GN^+}$ TBox axioms (concept inclusion axioms, role inclusion axioms and transitivity axioms). This fragment is not in GNFO just because of the cardinality restrictions and transitivity axioms, which are not expressible in GNFO. That is the reason for "+" in its abbreviation.

Example 4.18. Let us check if the concept inclusion axiom $\neg A \sqcup \{o\} \sqsubseteq \neg \leq 6R.\neg(\neg C \sqcup D)$ is a $\mathcal{SHOQ}_{GN^+}$ concept inclusion axiom. $PusgNeg(\neg \leq 6R.\neg(\neg C \sqcup D)) = \geq 7R.(C \sqcap \neg D)$ is a $\mathcal{SHOQ}_{GN^+}$ concept by (4.2). Hence, by the definition, the concept inclusion axiom is a $\mathcal{SHOQ}_{GN^+}$ concept inclusion axiom. Note that in the left-hand side of the inclusion, the concept $\neg A \sqcup \{o\}$, is not in $\mathcal{SHOQ}_{GN^+}$.

Proposition 4.19. All $\mathcal{SHOQ}_{GN^+}$ concepts are safe-range.

Proposition 4.20. Any safe-range $\mathcal{SHOQ}$ concept is equivalent to some $\mathcal{SHOQ}_{GN^+}$ concept.

Proposition 4.21. All $\mathcal{SHOQ}_{GN^+}$ TBox axioms are safe-range.

Proposition 4.22. Any safe-range $\mathcal{SHOQ}$ TBox axiom is equivalent to some $\mathcal{SHOQ}_{GN^+}$ TBox axiom.

The four last propositions prove the following theorem.

Theorem 4.23. The safe-range fragment of $\mathcal{SHOQ}$ and $\mathcal{SHOQ}_{GN^+}$ are equally expressive.

Theorem 4.11 and Theorem 4.23 imply the following theorem.

Theorem 4.24 (Expressive power equivalence). The domain independent fragment of $\mathcal{SHOQ}$ and $\mathcal{SHOQ}_{GN^+}$ are equally expressive.

4.3 Proofs of Section 4.1

First we need to prove an auxiliary proposition.

Proposition 4.25. Let $\phi(x)$ be any formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ with one free variable.

1. If $\phi(x)$ is domain independent then $\neg\phi(x)$ is not domain independent.
2. If $\forall x.\phi(x)$ is domain independent and satisfiable then $\phi(x)$ is not domain independent.

Proof.

1. Let us consider two interpretations: $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, $\mathcal{J} = \langle \Delta^{\mathcal{I}} \cup \{a\}, \cdot^{\mathcal{J}} \rangle$, where $a \notin \Delta^{\mathcal{I}}$ and $\cdot^{\mathcal{I}} = \cdot^{\mathcal{J}}$. Since $\phi(x)$ is domain independent and *act-range*$(\{x \to a\}) = a \notin \Delta^{\mathcal{I}}$, we have: $\mathcal{J}, \{x \to a\} \not\models \phi(x)$. Hence, $\mathcal{J}, \{x \to a\} \models \neg\phi(x)$. Then $\neg\phi(x)$ is not domain independent by definition because *act-range*$(\{x \to a\}) = a \notin \Delta^{\mathcal{I}}$.
2. Since $\forall x.\phi(x)$ is satisfiable, there exists an interpretation $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ such that $\mathcal{I} \models \forall x.\phi(x)$. Let $\mathcal{J} = \langle \Delta^{\mathcal{I}} \cup \{a\}, \cdot^{\mathcal{J}} \rangle$, where $a \notin \Delta^{\mathcal{I}}$ and $\cdot^{\mathcal{I}} = \cdot^{\mathcal{J}}$. Then, since $\forall x.\phi(x)$ is domain independent and $\mathcal{I} \models \forall x.\phi(x)$, we have that $\mathcal{J} \models \forall x.\phi(x)$. Then $\mathcal{J}, \{x \to a\} \models \phi(x)$. *act-range*$(\{x \to a\}) = a \notin \Delta^{\mathcal{I}}$, hence, $\phi(x)$ is not domain independent by definition.

The proposition is proved. □

Note that the Proposition 4.25 can be extended to the case, when ϕ has more than one free variable. The requirements of ϕ to be open and $\forall x.\phi(x)$ to be satisfiable (item 2) are essential for the proposition.

Example 4.26. Let ϕ be a closed formula. $\phi = \forall x.\, A(x) \vee \neg A(x) \equiv \mathbf{true}$ – closed and domain independent. $\neg\phi = \neg\forall x.\, A(x) \vee \neg A(x) \equiv \mathbf{false}$ – also domain independent. This example shows that if ϕ is not an open formula, then the item 1 of the Proposition 4.25 does not hold.

Example 4.27. Let $\phi = A(x) \wedge \neg A(x)$ - domain independent. Then $\forall x.\phi(x) = \forall x.\, A(x) \wedge \neg A(x) \equiv \mathbf{false}$ is domain independent and unsatisfiable. This example shows that in the case when $\forall x.\phi(x)$ is unsatisfiable, the item 2 of the Proposition 4.25 does not hold.

4.3.1 Proposition 4.1

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 a $\mathcal{SHOIQ}$

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, that it is impossible by the Proposition 4.25. 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.

- $C(x) = (\leq nR.D)(x) = \forall x_0 \ldots x_n.\, R(x, x_0) \wedge \ldots \wedge R(x, x_n) \wedge D(x_0) \ldots \wedge D(x_n) \rightarrow (x_0 = x_1) \ldots (x_{n-1} = x_n) \equiv \neg\exists x_0, \ldots, x_n.\, R(x, x_0) \wedge \ldots \wedge R(x, x_n) \wedge D(x_0) \wedge \ldots \wedge D(x_n) \wedge (x_0 \neq x_1) \wedge \ldots \wedge (x_{n-1} \neq x_n)$ - not safe-range, where D is any (possibly complex) concept. Then $\neg C(x) = \exists x_0, \ldots, x_n.\, R(x, x_0) \wedge \ldots \wedge R(x, x_n) \wedge D(x_0) \wedge \ldots \wedge D(x_n) \wedge (x_0 \neq x_1) \wedge \ldots \wedge (x_{n-1} \neq x_n)$ is safe-range.

- 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.

The proposition is proved. □

4.3.2 Corollary 4.2

Proof. By definition axiom $C \sqsubseteq D$ is safe-range if and only if the first-order logic formula $\neg\exists x.\, C(x) \wedge \neg D(x)$ is safe-range. This formula is safe-range if and only if the formula $C(x) \wedge \neg D(x)$ is safe-range. This formula is safe-range if and only if the $\mathcal{SHOIQ}$ concept $C \sqcap \neg D$ is safe-range. By items 2 and 4 of Proposition 4.1 this concept is safe-range if and only if C is safe-range or D is not safe-range.

The corollary is proved. □

Proposition 4.28. All $\mathcal{SHOIQ}$ role inclusion and transitivity axioms are safe-range.

Proof.

– Let $S \sqsubseteq R$ be any role inclusion axiom in $\mathcal{SHOIQ}$. Then $S \sqsubseteq R \equiv \neg\exists x, y.\, S(x,y)^* \wedge \neg R(x,y)^*$ – safe-range.

– For any role R in $\mathcal{SHOIQ}$, $Trans(R) = \neg\exists x, y, z.\, R(x,y)^* \wedge R(y,z)^* \wedge \neg R(x,z)^*$ – safe-range.

Here $(x,y)^*$ stands for (x,y) if the preceding role is atomic and $(x,y)^*$ stands for (y,x) if the preceding role is inverse atomic.

The proposition is proved. □

4.3.3 Theorem 4.3

Proof. Since any safe-range formula is domain independent, any safe-range concept (TBox axiom) in $\mathcal{SHOIQ}^*$ is domain independent. Let us prove now, that all domain independent concepts and TBox axioms in $\mathcal{SHOIQ}^*$ are safe-range.

1. **Concepts**. Let C be a domain independent concept in $\mathcal{SHOIQ}^*$. Let us prove that C is safe-range by contradiction. Suppose that C is not safe-range. Then by the item 4 of the Proposition 4.1 $\neg C$ is safe-range. Hence, $\neg C$ is domain independent. But since C is domain independent, by the first item of the Proposition 4.25 $\neg C$ is not domain independent. Contradiction.

2. **Concept inclusion axioms**. Let $C \sqsubseteq D$ be a domain independent concept inclusion axiom in $\mathcal{SHOIQ}^*$. That is $C \sqsubseteq D$ is satisfiable. $\forall x.\, C(x) \rightarrow D(x)$ is a corresponding domain independent formula in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$, where $C(x)$ and $D(x)$ are formulas in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ corresponding to the concepts C and D respectively. $\forall x.\, C(x) \rightarrow D(x) \equiv \forall x.\, \neg(C(x) \wedge \neg D(x))$ - domain independent and satisfiable. Then by the second item of the Proposition 4.25 $\neg(C(x) \wedge \neg D(x))$ is not domain independent. Hence, $\neg(C(x) \wedge \neg D(x))$ is not safe-range. That is the concept $\neg(C \sqcap \neg D)$ is not safe-range. Then by the item 4 of the Proposition 4.1, $C \sqcap \neg D$ is safe-range. That is, $C(x) \wedge \neg D(x)$ is safe-range. Hence, $\neg\exists x.\, C(x) \wedge \neg D(x)$ is safe-range. Therefore $C \sqsubseteq D$ is safe-range.

3. **Transitivity axioms**. Any transitivity axiom in $\mathcal{SHOIQ}$ and, hence, in $\mathcal{SHOIQ}^*$ is safe-range and, hence, domain independent by Proposition 4.28.

4. **Role inclusion axioms**. Any role inclusion axiom in $\mathcal{SHOIQ}$ and, hence, in $\mathcal{SHOIQ}^*$ is safe-range and, hence, domain independent by Proposition 4.28.

The theorem is proved. □

4.3.4 Theorem 4.4

Proof. Since any safe-range formula is domain independent and the Theorem 4.3 holds, the only thing one left to prove is that for any unsatisfiable domain independent concept inclusion axiom in $\mathcal{SHOIQ}$ there exists a logically equivalent safe-range concept inclusion axiom in $\mathcal{SHOIQ}$.
Let $C \sqsubseteq D$ be any unsatisfiable domain independent concept inclusion axiom in $\mathcal{SHOIQ}$. Then, since the set of constants (individual names) is not empty, $C \sqsubseteq D$ is logically equivalent to the unsatisfiable $\mathcal{SHOIQ}$ concept inclusion axiom $\{o\} \sqsubseteq \neg\{o\}$, where o is a constant. This axiom is safe-range, because its first-order translation is a safe-range formula $\neg\exists x.\,(x = o) \wedge (x = o)$.

The theorem is proved. □

4.4 Proofs of Subsection 4.2.1

4.4.1 Proposition 4.12

Proof. Let us prove by induction on the structure of $\mathcal{ALCHOIQ}_{GN}$ concepts defined by (4.1).

1. $A, \{o\}, \exists R, \exists R.C, \exists R.\neg C$ (C is an $\mathcal{ALCHOIQ}_{GN}$ concept) are safe-range because of the item 1 of Proposition 4.6.

2. For any atomic concept A, any individual o, any role R and any concept C in $\mathcal{ALCHOIQ}_{GN}$ the concepts $A \sqcap \neg C$, $\{o\} \sqcap \neg C$ and $\exists R \sqcap \neg C$ are safe-range because of the item 2 of Proposition 4.6 and since A, $\{o\}$ and $\exists R$ 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 4.6 respectively.

The proposition is proved. □

Lemma 4.29. For any safe-range concept C in $\mathcal{ALCHOI}$ the following holds:

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

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;

– $\exists R$, where R is an atomic role or inverse atomic role.

Proof. Let us prove the proposition by induction for all safe-range concepts of $\mathcal{ALCHOI}$.
Base. $A, \{o\}, \exists R, \exists R.C$ are safe-range by Proposition 4.6. $A \sqsubseteq A$, $\{o\} \sqsubseteq \{o\}$, $\exists R \sqsubseteq \exists R$, $\exists R.C \sqsubseteq \exists R$.
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, $C_1 \sqsubseteq B_1 \sqcup \ldots \sqcup B_m$, where B_i is a concept of the aforementioned type. Then $C_1 \sqcap C_2 \sqsubseteq C_1 \sqsubseteq B_1 \sqcup \ldots \sqcup B_m$.

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

3. $C = \neg D$ is safe-range. By Proposition 4.6 it is possible if and only if D is not safe-range. That is one of the following cases takes place.

   – $D = \forall R.D_1 \equiv \neg\exists R.\neg D_1$. Then $\neg D \equiv \exists R.\neg D_1 \sqsubseteq \exists R$. $\exists R$ is safe-range the item 1 of Proposition 4.6.

   – $D = D_1 \sqcap D_2$. 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. □

Lemma 4.30. For any $\mathcal{ALCHOI}$ concept C there exists an $\mathcal{ALCHOI}_{GN}$ concept C' such that either $C \equiv C'$ or $C \equiv \neg C'$.

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

1. Base. $A, \{o\}, \exists R$ are $\mathcal{ALCHOI}_{GN}$ concepts by the definition of $\mathcal{ALCHOI}_{GN}$ concept (4.1).

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

3. $C = \forall R.D \equiv \neg\exists R.\neg D$ and C' is an $\mathcal{ALCHOI}_{GN}$ concept such that $\exists R.\neg D \equiv C'$ or $\exists R.\neg D \equiv \neg C'$. Then $C \equiv \neg\exists R.\neg D \equiv \neg C'$ or $C \equiv \neg\exists R.\neg D \equiv \neg\neg C' \equiv C'$. And the item is proved.

4. $C = \neg D$ and D' is an $\mathcal{ALCHOI}_{GN}$ 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.

5. $C = C_1 \sqcap C_2$ and C_1' is an $\mathcal{ALCHOI}_{GN}$ concept such that $C_1 \equiv C_1'$ or $C_1 \equiv \neg C_1'$, C_2' is an $\mathcal{ALCHOI}_{GN}$ 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 $\mathcal{ALCHOI}_{GN}$ concept (because C_1' and C_2' are $\mathcal{ALCHOI}_{GN}$ 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 $\mathcal{ALCHOI}_{GN}$ concept (because C_1' and C_2' are $\mathcal{ALCHOI}_{GN}$ 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 $\mathcal{ALCHOI}_{GN}$ concept by Proposition 4.12 it is safe-range and, hence, by Lemma 4.29 $C_1' \sqsubseteq B_1 \sqcup \ldots \sqcup B_n$, 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 $\mathcal{ALCHOI}_{GN}$ concept (because C_2 is $\mathcal{ALCHOI}_{GN}$ concept and by the definition (4.1) of $\mathcal{ALCHOI}_{GN}$ concepts). Then $C' = C_1' \sqcap (B_1 \sqcap \neg C_2' \sqcup \ldots \sqcup B_n \sqcap \neg C_2')$ is an $\mathcal{ALCHOI}_{GN}$ concept. $C \equiv C'$. The item is proved.

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

The lemma is proved. □

Corollary 4.31. For any $\mathcal{ALCHOI}$ concept C and any concept B, which is either an atom A or $\{o\}$ or $\exists R$, the concept $B \sqcap C$ is equivalent to some $\mathcal{ALCHOI}_{GN}$ concept.

Proof. By Lemma 4.30 there exists an $\mathcal{ALCHOI}_{GN}$ 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 $\mathcal{ALCHOI}_{GN}$ concepts (by the definition (4.1) of $\mathcal{ALCHOI}_{GN}$ concepts).

The corollary is proved. □

4.4.2 Proposition 4.13

Proof. Let C be any safe-range $\mathcal{ALCHOI}$ concept. By Lemma 4.29 $C \sqsubseteq B_1 \sqcup \ldots \sqcup B_n$, where each B_i is either an atom A or $\{o\}$ or $\exists R$. 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 4.31 for each disjunct $B_i \sqcap C$ there exists an $\mathcal{ALCHOI}_{GN}$ 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 $\mathcal{ALCHOI}_{GN}$ concept as a disjunction of $\mathcal{ALCHOI}_{GN}$ concepts.

The proposition is proved. □

Proposition 4.32.

– All $\mathcal{ALCHOI}_{GN}$ role inclusion axioms are safe-range.

– All safe-range role inclusion axioms in $\mathcal{ALCHOI}$ are in $\mathcal{ALCHOI}_{GN}$.

Proof. It is easy to see that any role inclusion axiom in $\mathcal{ALCHOI}$ is a role inclusion axiom in $\mathcal{ALCHOI}_{GN}$. Then the proof of the proposition follows from Proposition 4.28. □

4.4.3 Proposition 4.14

Proof.

– Any $\mathcal{ALCHOI}_{GN}$ role inclusion axiom is safe-range by the first item of Proposition 4.32.

– Let $C \sqsubseteq D$ be any concept inclusion axiom in $\mathcal{ALCHOI}_{GN}$. 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 $\mathcal{ALCHOI}_{GN}$. 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 $\mathcal{ALCHOI}_{GN}$ concept $C \sqcap \neg D$ is safe-range. But by Proposition 4.12 any $\mathcal{ALCHOI}_{GN}$ concept is safe-range.

The proposition is proved. □

Lemma 4.33. For any safe-range $\mathcal{ALCHOI}$ concept C and any $\mathcal{ALCHOI}$ concept D the concept $C \sqcap D$ is equivalent to some $\mathcal{ALCHOI}_{GN}$ concept $C' \sqcap D'$, where C' and D' are $\mathcal{ALCHOI}_{GN}$ concepts.

Proof. Since C is safe-range by Lemma 4.29 $C \sqsubseteq B_1 \sqcup \ldots \sqcup B_n$, 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 Corollary 4.31 for each disjunct $B_i \sqcap D$ there is an equivalent $\mathcal{ALCHOI}_{GN}$ concept D_i. Hence, $D' := D_1 \sqcup \ldots \sqcup D_n$ is an $\mathcal{ALCHOI}_{GN}$ concept, that is equivalent to $B_1 \sqcap D \sqcup \ldots \sqcup B_n \sqcap D$. Since C is s safe-range by Proposition 4.13 there exists an $\mathcal{ALCHOI}_{GN}$ concept C' such that $C \equiv C'$. Then $C \sqcap D \equiv C' \sqcap D'$, and $C' \sqcap D'$ is an $\mathcal{ALCHOI}_{GN}$ concept, where C' and D' are $\mathcal{ALCHOI}_{GN}$ concepts.

The lemma is proved. □

Proposition 4.34. Any safe-range $\mathcal{ALCHOI}$ concept inclusion axiom $C \sqsubseteq D$ can be transformed to a concept inclusion axiom $C' \sqsubseteq \neg D'$, where C' and D' are $\mathcal{ALCHOI}_{GN}$.

Proof. Let $C \sqsubseteq D$ be any safe-range $\mathcal{ALCHOI}$ 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 $\mathcal{ALCHOI}$ concept $C \sqcap \neg D$ is safe-range. By Proposition 4.6 we have that C is safe-range or $\neg D$ is safe-range.

– C is safe-range. Then by Lemma 4.33 there exist two $\mathcal{ALCHOI}_{GN}$ concepts C' and D' such that $C \sqcap \neg D$ is logically equivalent to the $\mathcal{ALCHOI}_{GN}$ 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 $\mathcal{ALCHOI}_{GN}$ concepts).

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

The proposition is proved. □

Proposition 4.35. For any two $\mathcal{ALCHOI}_{GN}$ concepts C and D the axiom $C \sqsubseteq \neg D$ is an $\mathcal{ALCHOI}_{GN}$ 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 $\mathcal{ALCHOI}_{GN}$ concept inclusion axiom the axiom $C \sqsubseteq \neg D$ is an $\mathcal{ALCHOI}_{GN}$ concept inclusion axiom.

The proposition is proved. □

4.4.4 Proposition 4.15

Proof. The proof is implied by the second item of Proposition 4.32, Proposition 4.34 and Proposition 4.35. □

4.4.5 Theorem 4.16

Proof. Just take into account Proposition 4.12, Proposition 4.13, Proposition 4.14, Proposition 4.15 and Proposition 4.32. □

4.5 Proofs of Subsection 4.2.2

4.5.1 Proposition 4.19

Proof. Let us prove by induction on the structure of $\mathcal{SHOQ}_{GN^+}$ concepts defined by (4.2).

1. $A, \{o\}, \geq nR, \geq nR.C, \geq nR.\neg C$ (C is an $\mathcal{SHOQ}_{GN^+}$ concept) are safe-range because of the item 1 of Proposition 4.7.

2. For any atomic concept A, any individual o any atomic 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 2 of Proposition 4.7 and since A, $\{o\}$ and $\geq nR$ are safe-range by the first item.

3. Suppose that $\mathcal{SHOQ}_{GN^+}$ 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 4.7 respectively.

The proposition is proved. □

Lemma 4.36. For any safe-range concept C in $\mathcal{SHOQ}$ the following holds:

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

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, n is a natural number.

Lemma 4.37. For any $\mathcal{SHOQ}$ concept C there exists a $\mathcal{SHOQ}_{GN^+}$ concept C' such that either $C \equiv C'$ or $C \equiv \neg C'$.

Corollary 4.38. For any $\mathcal{SHOQ}$ 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 $\mathcal{SHOQ}_{GN^+}$ concept.

Proofs of Lemma 4.36, Lemma 4.37, Corollary 4.38 and **Proposition 4.20** repeat the proofs of Lemma 4.29, Lemma 4.30, Corollary 4.31 and Proposition 4.13 respectively. One just needs to replace existential restrictions with cardinality restrictions, $\mathcal{ALCHOI}_{GN}$ with $\mathcal{SHOQ}_{GN^+}$ and auxiliary statements (lemmas, propositions and corollaries) for $\mathcal{ALCHOI}_{GN}$ with the corresponding auxiliary statements for $\mathcal{SHOQ}_{GN^+}$.

Lemma 4.39. $\mathcal{SHOQ}$ concept inclusion axiom $C \sqsubseteq D$ is safe-range if and only if either C is safe-range or $\neg D$ is safe-range.

Proof. Let $C \sqsubseteq D$ is any concept inclusion axiom in $\mathcal{SHOQ}$. It is safe-range if and only if the corresponding first-order formula $\neg\exists x.\, C(x) \wedge \neg D(x)$ is safe-range. It is safe-range if and only if the concept $C \sqcap \neg D$ is safe-range, which, by Proposition 4.7 is safe-range if and only if either the concept C is safe-range or the concept $\neg D$ is safe-range. Hence, we have that the concept inclusion $C \sqsubseteq D$ is safe-range if and only if either C is safe-range or $\neg D$ is safe-range.

The lemma is proved. □

4.5.2 Proposition 4.21

Proof.

– All $\mathcal{SHOQ}_{GN^+}$ role inclusion and transitivity axioms are safe-range because all $\mathcal{SHOQ}$ role inclusion and transitivity axioms are safe-range by Proposition 4.28.

– Let $C \sqsubseteq D$ be any concept inclusion axiom in $\mathcal{SHOQ}_{GN^+}$. Then by definition we have that either C is in $\mathcal{SHOQ}_{GN^+}$ and, hence, safe-range by Proposition 4.19 or $SimplePushNeg(\neg D)$ is in $\mathcal{SHOQ}_{GN^+}$ and, hence, safe-range by Proposition 4.19 and, hence, $\neg D$ is safe-range, because $SimplePushNeg$ is a necessary step in transformation of a formula to safe-range normal form. Then by Lemma 4.39 the concept inclusion $C \sqsubseteq D$ is safe-range.

The proposition is proved. □

4.5.3 Proposition 4.22

Proof.

– All role inclusion and transitivity axioms in $\mathcal{SHOQ}$ are safe-range by Proposition 4.28. And all role inclusion and transitivity axioms in $\mathcal{SHOQ}$ are in $\mathcal{SHOQ}_{GN^+}$. Hence, all safe-range role inclusion and transitivity axioms in $\mathcal{SHOQ}$ are in $\mathcal{SHOQ}_{GN^+}$.

– Let $C \sqsubseteq D$ be any safe-range concept inclusion axiom in $\mathcal{SHOQ}$. Hence, by Lemma 4.39, either C is safe-range or $\neg D$ is safe-range. By Proposition 4.20, either C is equivalent to some $\mathcal{SHOQ}_{GN^+}$ concept C' or $\neg D$ is equivalent to some $\mathcal{SHOQ}_{GN^+}$ concept D'.

  – Suppose that C is equivalent to some $\mathcal{SHOQ}_{GN^+}$ concept C'. Then $C \sqsubseteq D$ is equivalent to $C' \sqsubseteq D$. By definition $C' \sqsubseteq D$ is a $\mathcal{SHOQ}_{GN^+}$ concept inclusion axiom.

  – Suppose that $\neg D$ is equivalent to some $\mathcal{SHOQ}_{GN^+}$ concept D'.
  Then $C \sqsubseteq D$ is equivalent to $C \sqsubseteq \neg D'$. $SimplePushNeg(\neg\neg D') = SimplePushNeg(D') = D'$ – $\mathcal{SHOQ}_{GN^+}$ concept, because it is evident by the definition (4.2) of $\mathcal{SHOQ}_{GN^+}$ concepts that for any $\mathcal{SHOQ}_{GN^+}$ concept C, $SimplePushNeg(C) = C$.

The proposition is proved. □

5. Query Reformulation

This chapter describes our query reformulation framework that presents a constructive way (a tableau-based algorithm) to obtain a reformulation of a query under an ontology. We also show which conditions an ontology and a query should satisfy in order to obtain as an output of the algorithm an "SQL-executable" reformulation – namely a safe-range formula – that can be evaluated over a database (DBox) without taking into account the ontology. Finally, we present applications of the framework in expressive description logics $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$.

A *DBox* $\mathcal{DB}$ 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 $c_i \in \mathbb{C}$ $(1 \leq i \leq n)$. DBox can be seen as a variant of database representation. The set of all predicates appearing in a DBox $\mathcal{DB}$ 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*.

An interpretation $\mathcal{I}$ *embeds a DBox* $\mathcal{DB}$, if it holds that $a^{\mathcal{I}} = a$ for every DBox 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 DBox $\mathcal{DB}$ as $E(\mathcal{DB})$.

In other words, in every interpretation embedding $\mathcal{DB}$ the interpretation of any DBox predicate is always the same and it is given exactly by its content in the DBox; this is, in general, not the case for the interpretation of the non-DBox predicates. We say that all the DBox 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 DBox in an interpretation. In an open world, an interpretation $\mathcal{I}$ soundly embeds a DBox 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 DBox 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 $\mathbb{C}$, which we assume to be finite. 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 $\mathbb{C}$.

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 DBox $\mathcal{DB}$, and call it a *semantic active domain of a signature* σ' *in a DBox* $\mathcal{DB}$.

Intuitively, $adom(\sigma', \mathcal{DB})$ includes the constants from σ' and from $\mathcal{DB}$ appearing in the relations corresponding to the predicates from σ'. If $\sigma' = \sigma(\phi) \subseteq \mathcal{DB}$, where ϕ is a formula, we call $adom(\sigma(\phi), \mathcal{DB})$ a *semantic active domain of a formula ϕ in a DBox $\mathcal{DB}$*.
Let $X \subseteq \mathbb{X}$ be a set of variables and $\mathbb{S}$ a set; in this chapter we consider the restriction of a substitution to a set of variables from $\mathbb{X}$. That is, we consider a function $\Theta|_X$ assigning an element in $\mathbb{S}$ to each variable in X. We abuse the notation and call such restriction simply *substitution*. Thus, hereafter substitution is a function from a set of variables $X \subseteq \mathbb{X}$ to a set S: $\Theta : X \mapsto S$, including the empty substitution ϵ when $X = \emptyset$. 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 $\phi(\bar{x})$ is true in an interpretation $\mathcal{I}$ with its free variables substituted according to a substitution $\Theta : \bar{x} \mapsto \mathbb{C}'$ as $\mathcal{I} \models \phi(\bar{x}/\Theta)$.
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 $\phi(\bar{x})$ is true in $\mathcal{I}$ with its free variables interpreted according to a substitution $\Theta : \bar{x} \mapsto \Delta$ as $\mathcal{I}, \Theta \models \phi(\bar{x})$.
The *extension domain* of a formula $\phi(\bar{x})$ with respect to the interpretation $\mathcal{I}$ is defined as the set of domain elements $\bigcup\{rng(\Theta) \mid dom(\Theta) = \bar{x},\ rng(\Theta) \subseteq \Delta,\ \mathcal{I}, \Theta \models \phi(\bar{x})\}$.
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 (respectively $\mathcal{KB}$) is denoted as $M(\phi)$ (respectively $M(\mathcal{KB})$). A DBox $\mathcal{DB}$ is *legal for an ontology $\mathcal{KB}$* if there exists a model of $\mathcal{KB}$ embedding $\mathcal{DB}$.

5.1 Queries

A *query* is a (possibly closed) formula. Given a query $\mathcal{Q}(\bar{x})$. We define the *certain answer over a DBox $\mathcal{DB}$ and under an ontology $\mathcal{KB}$* as follows:

Definition 5.1 (Certain answer). The (certain) answer to a query $\mathcal{Q}(\bar{x})$ over a DBox $\mathcal{DB}$ under an ontology $\mathcal{KB}$ is the set of substitutions with constants:

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

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 DBox, and $\emptyset$ otherwise. In the following, we assume that the closed formula $\mathcal{Q}(\bar{x}/\Theta)$ is neither valid nor inconsistent under the ontology $\mathcal{KB}$, given a substitution $\Theta : \bar{x} \mapsto \mathbb{C}$

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.
One can see that if an ontology is inconsistent or a DBox is illegal for an ontology, then the certain answer to any query over the DBox under the ontology is a set of all possible substitutions. Also, if an ontology is a tautology, we actually have a simple case of query answering over a database (DBox) without an ontology. Thus, we can discard these cases and assume to have only consistent non-tautological ontologies and legal DBoxes.
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 the standard name assumption as $I(\mathit{SNA})$. We denote the set of all interpretations satisfying the 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 DBoxes. 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 5.2 (**SNA vs UNA**). For any query $\mathcal{Q}(\bar{x})$, ontology $\mathcal{KB}$ and DBox $\mathcal{DB}$,
$\{\Theta \mid \mathit{dom}(\Theta) = \bar{x},\ \mathit{rng}(\Theta) \subseteq \mathbb{C},\ \forall \mathcal{I} \in I(\mathit{SNA}) \cap M(\mathcal{KB}) \cap E(\mathcal{DB}) : \mathcal{I} \models \mathcal{Q}(\bar{x}/\Theta)\} =$
$\{\Theta \mid \mathit{dom}(\Theta) = \bar{x},\ \mathit{rng}(\Theta) \subseteq \mathbb{C},\ \forall \mathcal{I} \in I(\mathit{UNA}) \cap M(\mathcal{KB}) \cap E(\mathcal{DB}) : \mathcal{I} \models \mathcal{Q}(\bar{x}/\Theta)\}$.

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 $Q(x) = \neg Student(x)$ over the database (DBox) $\{Student(a), Student(b)\}$, with domain $\{a, b, c\}$ has the answer $\{x \to c\}$, while with domain $\{a, b, c, d\}$ has the answer $\{x \to c,\ x \to d\}$. 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 5.3 (**Domain independence**). A formula $\mathcal{Q}(\bar{x})$ is *domain independent with respect to an ontology* $\mathcal{KB}$ if and only if 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. are compatible), and for every substitution $\Theta : \bar{x} \mapsto \Delta^{\mathcal{I}} \cup \Delta^{\mathcal{J}}$ we have:

$$rng(\Theta) \subseteq \Delta^{\mathcal{I}} \text{ and } \mathcal{I}, \Theta \models \mathcal{Q}(\bar{x}) \quad \text{iff}$$
$$rng(\Theta) \subseteq \Delta^{\mathcal{J}} \text{ and } \mathcal{J}, \Theta \models \mathcal{Q}(\bar{x}).$$

The above definition reduces to the classical definition of domain independence whenever the ontology is empty. In the case when $\mathcal{KB} = \emptyset$ this definition coincide with Definition 3.3. For convenience in this section use this variant of definition of domain independence. A weaker version of domain independence – which is relevant for open formulas – is the following.

Definition 5.4 (**Ground domain independence**)**.** A formula $\mathcal{Q}(\bar{x})$ is *ground domain independent* if and only if $\mathcal{Q}(\bar{x}/\Theta)$ is domain independent for every substitution $\Theta : \bar{x} \mapsto \mathbb{C}$.

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 (that we mentioned already in Section 3.2); 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 (see Definition 3.12). For example, the formula $\neg A(x) \wedge B(x)$ is safe-range, while queries $\neg A(x)$ and $\forall x.\, A(x)$ are not. To check whether a formula is safe-range, the formula is transformed into a logically equivalent *safe-range normal form* and its *range restriction* is computed according to a set of syntax based rules; the range restriction of a formula is a subset of its free variables, and if 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).

5.2 Determinacy

Definition 5.5 (**Finite determinacy** or **implicit definability**)**.** A query $\mathcal{Q}(\bar{x})$ is (finitely) determined by (or *implicitly definable* from) the DBox predicates $\mathbb{P}_{\mathcal{DB}}$

under $\mathcal{KB}$ if and only if for any two models $\mathcal{I}$ and $\mathcal{J}$ of the ontology $\mathcal{KB}$ – both with a *finite* interpretation to the DBox 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 $\Theta : \bar{x} \mapsto \Delta^{\mathcal{I}}$ we have: $\mathcal{I}, \Theta \models \mathcal{Q}(\bar{x})$ if and only if $\mathcal{J}, \Theta \models \mathcal{Q}(\bar{x})$.

Intuitively, the answer to an implicitly definable query does not depend on the interpretation of non-DBox predicates. Once the DBox 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 DBox predicates and constants and on the domain (which are fixed). In practice, by focusing 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 DBox predicates (finite determinacy) and determinacy under models with an unrestricted interpretation of DBox predicates (unrestricted determinacy) coincide. We say that these fragments have *finitely controllable determinacy*. Sometimes it may be the case that we consider ontology in one fragment ($\mathcal{F}_1$) and query in another one ($\mathcal{F}_2$). Then we say that fragment $\mathcal{F}_1$ has *finitely controllable determinacy of queries from fragment* $\mathcal{F}_2$ if for every query expressed in $\mathcal{F}_2$ and for every ontology expressed in $\mathcal{F}_1$ finite determinacy of the query under the ontology coincides with unrestricted 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 thesis would fail if finite determinacy and unrestricted determinacy do not coincide: it can be shown (Gurevich, 1984) that Theorem 5.9 below fails if we consider only models with a finite interpretation of DBox predicates.

Example 5.6 (**Example from database theory**). Let $\mathbb{P} = \{P, R, A\}$, $\mathbb{P}_{\mathcal{DB}} = \{P, R\}$,

$$\begin{aligned}\mathcal{KB} = \{&\forall x, y, z.\, R(x, y) \wedge R(x, z) \rightarrow y = z,\\ &\forall x, y.\, R(x, y) \rightarrow \exists z.\, R(z, x),\\ &(\forall x, y.\, R(x, y) \rightarrow \exists z.\, R(y, z)) \rightarrow (\forall x.\, A(x) \leftrightarrow P(x))\}.\end{aligned}$$

$\mathcal{KB}$ is domain independent. The formula $\forall x, y.\, R(x, y) \rightarrow \exists z.\, R(y, z)$ is entailed from the first two formulas *only* over finite interpretations of R. The query $\mathcal{Q} = A(x)$ is domain independent and 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 DBox predicate under models with an unrestricted interpretation of R. The fragment in which $\mathcal{KB}$ and $\mathcal{Q}$ are expressed does not enjoy finitely controllable determinacy.

This theorem immediately follows from the example above.

Theorem 5.7. Domain independent fragment does not have 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 DBox predicates and constants.

Definition 5.8 (**Exact reformulation** or **explicit definability**)**.** A query $\mathcal{Q}(\bar{x})$ is *explicitly definable from the DBox predicates* $\mathbb{P}_{\mathcal{DB}}$ *under the ontology* $\mathcal{KB}$ if and only if there is some formula $\widehat{\mathcal{Q}}(\bar{x})$ in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$, such that $\mathcal{KB} \models \forall \bar{x}.\, \mathcal{Q}(\bar{x}) \leftrightarrow \widehat{\mathcal{Q}}(\bar{x})$ and $\sigma(\widehat{\mathcal{Q}}) \subseteq \mathbb{P}_{\mathcal{DB}}$. We call this formula $\widehat{\mathcal{Q}}(\bar{x})$ an *exact reformulation of* $\mathcal{Q}(\bar{x})$ *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 DBox predicates *if and only if* there exists a first-order exact reformulation.

Theorem 5.9 (**Projective Beth definability**, (Beth, 1953))**.** A query $\mathcal{Q}(\bar{x})$ is implicitly definable from the DBox predicates $\mathbb{P}_{\mathcal{DB}}$ under an ontology $\mathcal{KB}$, if and only if it is explicitly definable as a formula $\widehat{\mathcal{Q}}(\bar{x})$ in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ over $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$.

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

Theorem 5.10 (**Testing determinacy**, which can be easily shown from Beth (1953))**.** A query $\mathcal{Q}(\bar{x})$ is *implicitly definable from the DBox predicates* $\mathbb{P}_{\mathcal{DB}}$ *under the ontology* $\mathcal{KB}$ if and only if $\mathcal{KB} \cup \widetilde{\mathcal{KB}} \models \forall \bar{x}.\, \mathcal{Q}(\bar{x}) \leftrightarrow \widetilde{\mathcal{Q}}(\bar{x})$.

5.2.1 Finite controllability of determinacy for GNFO

Finite controllability of determinacy for GNFO is important of us because we consider GNFO subfragments of DLs (Section 5.6) for application of our query reformulation framework.
Suppose, we have an ontology $\mathcal{KB}$ and a query $\mathcal{Q}(\bar{x})$ both expressed in GNFO. $\mathbb{P}_{\mathcal{DB}}$ is a set of DBox predicates. Let τ be a conjunction of all sentences in $\mathcal{KB}$.

Then by the Theorem 5.9 we will have a finite controllability of determinacy if validity of the sentence $\tau \wedge \widetilde{\tau} \rightarrow (\forall \bar{x}. \mathcal{Q}(\bar{x}) \rightarrow \widetilde{\mathcal{Q}}(\bar{x}))$ over finite models implies the validity of the sentence over unrestricted models. That is, we have a finite controllability of determinacy if whenever sentence $\tau \wedge \widetilde{\tau} \wedge \exists \bar{x}. (\mathcal{Q}(\bar{x}) \wedge \neg \widetilde{\mathcal{Q}}(\bar{x}))$ has a finite model, it has an unrestricted one. If $\mathcal{Q}(\bar{x}) \wedge \neg \widetilde{\mathcal{Q}}(\bar{x})$ is in GNFO then the sentence $\tau \wedge \widetilde{\tau} \wedge \exists \bar{x}. (\mathcal{Q}(\bar{x}) \wedge \neg \widetilde{\mathcal{Q}}(\bar{x}))$ is in GNFO, and we gain finite controllability of determinacy simply by finite model property of GNFO. But even if $\mathcal{Q}(\bar{x})$ is a conjunctive query, which is a strict subfragment of GNFO, the formula $\mathcal{Q}(\bar{x}) \wedge \neg \widetilde{\mathcal{Q}}(\bar{x})$ may not be in GNFO.

Example 5.11. $Q(x, z) = \exists y.\, A(x, y) \wedge B(z, y)$. Then $Q(x, z) \wedge \neg \widetilde{Q}(x, z) = (\exists y.\, A(x, y) \wedge B(z, y)) \wedge \neg (\exists y.\, \widetilde{A}(x, y) \wedge \widetilde{B}(z, y)) = \exists y_1. (A(x, y_1) \wedge B(z, y_1) \wedge \neg \exists y_2.\, \widetilde{A}(x, y_2) \wedge \widetilde{B}(z, y_2)) \notin$GNFO.

We say that a first-order logic formula is *answer-guarded* if it has a form

$$Atom(\bar{x}) \wedge \varphi(\bar{x}),$$

where $Atom$ is a predicate which arity is equal to the number of free variables of the formula.

Theorem 5.12. GNFO has finitely controllable determinacy of

– answer-guarded GNFO queries;

– boolean GNFO queries;

– GNFO queries with one free variable.

The problem of checking whether a query is implicitly definable reduces to the problem of checking entailment in first-order logic.

5.3 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 DBox signature $\mathbb{P}_{\mathcal{DB}}$, an ontology $\mathcal{KB}$, and a query $\mathcal{Q}(\bar{x})$ expressed in some fragment of $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ and determined by the DBox predicates, our goal is to find a safe-range reformulation $\widehat{\mathcal{Q}}(\bar{x})$ of $\mathcal{Q}(\bar{x})$ in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$, that when evaluated as a relational algebra expression over a legal DBox, gives the same

answer as the certain answer to $\mathcal{Q}(\bar{x})$ over the DBox under $\mathcal{KB}$. This can be reformulated as the following problem:

Problem 5.13 (**Exact safe-range query reformulation**)**.** Find an exact reformulation $\widehat{\mathcal{Q}}(\bar{x})$ of $\mathcal{Q}(\bar{x})$ 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 to the original query and to the reformulated query are identical. More precisely, the following proposition holds.

Proposition 5.14. Given a DBox $\mathcal{DB}$, let $\mathcal{Q}(\bar{x})$ be implicitly definable from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$ and let $\widehat{\mathcal{Q}}(\bar{x})$ be an exact reformulation of $\mathcal{Q}(\bar{x})$ under $\mathcal{KB}$ over $\mathbb{P}_{\mathcal{DB}}$, then:

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

From the above equation it is clear that in order to answer to an exactly reformulated query, one may still need to consider all the models of the ontology embedding the DBox, i.e., we still have an entailment problem to solve. The following theorem states the condition to reduce the original query answering problem – based on entailment – to the problem of checking the validity of the exact reformulation over a *single* model: the condition is that the reformulation should be domain independent. Indeed there is only one interpretation (with a particular domain) embedding the DBox with the signature restricted to the DBox predicates.

Theorem 5.15 (**Adequacy of exact safe-range query reformulation**)**.** Let $\mathcal{DB}$ be a DBox which is legal for $\mathcal{KB}$, and let $\mathcal{Q}(\bar{x})$ be a query. If $\widehat{\mathcal{Q}}(\bar{x})$ is an exact domain independent (or safe-range) reformulation of $\mathcal{Q}(\bar{x})$ under $\mathcal{KB}$ over $\mathbb{P}_{\mathcal{DB}}$, then:

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

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 DBox, if the exact reformulation is not safe-range.

Example 5.16. Let $\mathbb{P} = \{P, A\}$, $\mathbb{P}_{\mathcal{DB}} = \{P\}$, $\mathbb{C} = \{a\}$,
$\mathcal{DB} = \{P(a, a)\}$, $\mathcal{KB} = \{\forall y.\, P(a, y) \vee A(y)\}$,
$\mathcal{Q}(\bar{x}) = \widehat{\mathcal{Q}}(\bar{x}) = \forall y.\, P(x, y)$ (i.e., $\bar{x} = \{x\}$).

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

As we have seen, answers to a query for which a reformulation exists will contain only constants from the active domain of the DBox and the query; therefore, ground statements in the ontology involving non-DBox 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 DBox.

5.4 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 5.17. Let $\mathbb{P} = \{A, B\}$, $\mathbb{P}_{\mathcal{DB}} = \{A\}$, $\mathcal{KB} = \{\forall x.B(x) \leftrightarrow A(x)\}$, $\mathcal{Q}(x) = \neg B(x)$. Then $\mathcal{Q}(x)$ is implicitly definable from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$, and every exact reformulation of $\mathcal{Q}(x)$ 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 5.18. Let $\mathbb{P}_{\mathcal{DB}} = \{A, C\}$, $\mathcal{KB} = \{\neg A(a), \forall x.\, A(x) \leftrightarrow B(x)\}$ and let a query $\mathcal{Q}(x) = \exists y\, \neg B(y) \wedge C(x)$. It is easy to see that $\mathcal{KB}$ is domain independent and $\mathcal{Q}(x)$ is not. $\mathcal{Q}(x)$ is implicitly definable from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$, and $\widehat{\mathcal{Q}}(x) = \neg A(a) \wedge C(x)$ is an exact domain independent reformulation of $\mathcal{Q}(x)$. □

If in the example above $\mathbb{P}_{\mathcal{DB}} = \{B, C\}$ and original query to be rewritten is $\widehat{\mathcal{Q}}(x)$, then $\mathcal{Q}(x)$ is an exact reformulation of $\widehat{\mathcal{Q}}(x)$ under $\mathcal{KB}$ over $\mathbb{P}_{\mathcal{DB}}$. That is, even domain independence of an ontology and an original query does not guarantee domain independence of a reformulation.
It is obvious that in spite of the fact that the query $\mathcal{Q}(x)$ form the example above 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 5.19 (**Semantic characterisation**)**.** Given a set of DBox predicates $\mathbb{P}_{\mathcal{DB}}$. A domain independent ontology $\mathcal{KB}$, and a query $\mathcal{Q}(\bar{x})$, a domain independent exact reformulation $\widehat{\mathcal{Q}}(\bar{x})$ of $\mathcal{Q}(\bar{x})$ over $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$ exists if and only if $\mathcal{Q}(\bar{x})$ 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 (a fragment with finite controllability of determinacy), and gives us a constructive way to compute it, if it exists.

Theorem 5.20 (**Constructive**)**.** If:

1. $\mathcal{KB} \cup \widetilde{\mathcal{KB}} \models \forall \bar{x}.\, \mathcal{Q}(\bar{x}) \leftrightarrow \widetilde{\mathcal{Q}}(\bar{x})$ (that is, $\mathcal{Q}(\bar{x})$ is implicitly definable),
2. $\mathcal{Q}(\bar{x})$ is safe-range (that is, $\mathcal{Q}(\bar{x})$ is domain independent),
3. $\mathcal{KB}$ is safe-range (that is, $\mathcal{KB}$ is domain independent),

then there exists an exact reformulation $\widehat{\mathcal{Q}}(\bar{x})$ of $\mathcal{Q}(\bar{x})$ 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 $\widehat{\mathcal{Q}}(\bar{x})$ from a validity proof of the implication $(\mathcal{KB} \wedge \mathcal{Q}(\bar{x})) \rightarrow (\widetilde{\mathcal{KB}} \rightarrow \widetilde{\mathcal{Q}}(\bar{x}))$. See Section 5.5 for full details.
Let us now consider a fully worked out example, adapted from the paper by Nash et al. (2010).

Example 5.21. Given: $\mathbb{P} = \{R, V_1, V_2, V_3\}$, $\mathbb{P}_{\mathcal{DB}} = \{V_1, V_2, V_3\}$,

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

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 $(\mathcal{KB} \wedge \mathcal{Q}(\bar{x})) \rightarrow (\widetilde{\mathcal{KB}} \rightarrow \widetilde{\mathcal{Q}}(\bar{x}))$ 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 to $\widehat{\mathcal{Q}}(x,y)$ is in the semantic active domain of the signature $\sigma(\widehat{\mathcal{Q}}) \subseteq \mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}_{\widehat{\mathcal{Q}}}$ in the DBox $\mathcal{DB}$ (see Lemma 5.37), all fee variables in $\widehat{\mathcal{Q}}(x,y)$ can be "guarded" by some DBox predicates or constants. Note that $\mathcal{Q}(x,y) = \exists z, v, u.\, R(z,x) \wedge R(z,v) \wedge R(v,u) \wedge R(u,y)\} \equiv^{\mathcal{KB}} \exists z, v.\, R(z,x) \wedge R(z,v) \wedge V_2(v,y) \equiv^{\mathcal{KB}} \mathcal{Q}(x,y) \wedge V_2(v,y)$ (where '$\equiv^{\mathcal{KB}}$' means "logically equivalent with respect to $\mathcal{KB}$"). Then $\mathcal{KB} \models \mathcal{Q}(x,y) \leftrightarrow \widehat{\mathcal{Q}}(x,y) \wedge \exists v.\, V_2(v,y)$. Therefore, $\widehat{\mathcal{Q}}(x,y) \wedge \exists v.\, V_2(v,y) = (\exists z.\, V_1(x,z) \wedge \forall v.\, (V_2(v,z) \rightarrow V_3(v,y))) \wedge \exists v.\, V_2(v,y)$ is an exact safe-range reformulation of $\mathcal{Q}(x,y)$ from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$.

5.5 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 5.20 are satisfied. The method is based on the notion of interpolant introduced by Craig (1957).

Definition 5.22 (Interpolant). The sentence χ is an *interpolant* for the sentence $\phi \rightarrow \psi$ 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 $\phi \rightarrow \chi$ and $\chi \rightarrow \psi$ are valid sentences in $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$.

Theorem 5.23 (**Craig's interpolation**). If $\phi \rightarrow \psi$ 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 5.9) 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 5.23).
An interpolant is used to find an exact reformulation of a given implicitly definable query as follows.

Theorem 5.24 (**Interpolant as definition**). Let $\mathcal{Q}(\bar{x})$ be a query with $n \geq 0$ free variables implicitly definable from the DBox predicates $\mathbb{P}_{\mathcal{DB}}$ under the ontology $\mathcal{KB}$. Then, the closed formula with $c_1, \ldots, c_n$ distinct constant symbols in $\mathbb{C}$ not appearing in $\mathcal{KB}$ or $\mathcal{Q}(\bar{x})$:

$$((\bigwedge \mathcal{KB}) \wedge \mathcal{Q}(\bar{x}/c_1, \ldots, c_n)) \rightarrow ((\bigwedge \widetilde{\mathcal{KB}}) \rightarrow \widetilde{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n)) \qquad (5.1)$$

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

Therefore, to find an exact reformulation of an implicitly definable query in terms of DBox predicates it is enough to find an interpolant of the implication (5.1) and then to substitute all the constants $c_1, \ldots, c_n$ back with the free variables $\bar{x}$ of the original query. An interpolant can be constructed from a validity proof of (5.1) 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).

5.5.1 Tableau-based method to compute an interpolant

In this section we recall in our context the tableau based method to compute an interpolant. This method was described and its correctness was proved in Fitting (1996).
Assume $\phi \rightarrow \psi$ 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 5.25 (**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: $\dfrac{X(\neg\neg\varphi)}{X(\varphi)}$ $\quad$ $\dfrac{X(\neg\top)}{X(\bot)}$ $\quad$ $\dfrac{X(\neg\bot)}{X(\top)}$

$\alpha-$rule: $\dfrac{X(\varphi_1 \wedge \varphi_2)}{\begin{array}{c} X(\varphi_1) \\ X(\varphi_2) \end{array}}$

$\beta-$rule: $\dfrac{X(\neg(\neg\varphi_1 \wedge \neg\varphi_2))}{X(\varphi_1) \quad | \quad X(\varphi_2)}$

First order rules

$\gamma-$rule: $\dfrac{X(\forall x.\varphi)}{X(\varphi(t))}$ for any $t \in C^{par}$

$\sigma-$rule: $\dfrac{X(\exists x.\varphi)}{X(\varphi(c))}$ for a new constant c

Equality rules

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

replacement rule: $\dfrac{X(t=u)Y(\varphi(t))}{Y(\varphi(u))}$

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 first-order logic 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: $S \cup \{L(\varphi), L(\neg\varphi)\} \xrightarrow{int} \bot$ $\quad$ r2: $S \cup \{R(\varphi), R(\neg\varphi)\} \xrightarrow{int} \top$

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

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

Rules for propositional cases

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

p3: $\dfrac{S \cup \{X(\bot)\} \xrightarrow{int} I}{S \cup \{X(\neg\top)\} \xrightarrow{int} I}$ $\quad$ 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 case

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

f2: $$\frac{S \cup \{L(\varphi(c))\} \xrightarrow{int} I}{S \cup \{L(\forall x.\varphi(x))\} \xrightarrow{int} I}$$ if c occurs in $\{\phi_1, ..., \phi_n\}$

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

f4: $$\frac{S \cup \{L(\varphi(c))\} \xrightarrow{int} I}{S \cup \{L(\forall x.\varphi(x))\} \xrightarrow{int} \forall x.I[c/x]}$$ if c does not occur in $\{\phi_1, ..., \phi_n\}$

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

Rules for equality cases

e1: $$\frac{S \cup \{X(\varphi(p)), X(t = t)\} \xrightarrow{int} I}{S \cup \{X(\varphi(p))\} \xrightarrow{int} I}$$

e2: $$\frac{S \cup \{X(\varphi(u)), X(t = u)\} \xrightarrow{int} I}{S \cup \{X(\varphi(t)), X(t = u)\} \xrightarrow{int} I}$$

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

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

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

e6: $$\frac{S \cup \{R(\varphi(u)), L(t = u)\} \xrightarrow{int} I}{S \cup \{R(\varphi(t)), L(t = u)\} \xrightarrow{int} I[u/t]}$$ 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 5.26. 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 $S \xrightarrow{int} I$.

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

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

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

and the interpolant of S_1 can be computed as in Figure 5.1 below. 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 $\mathcal{Q}(x)$.

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

Figure 5.1 – Computation of the interpolant of S_1

5.5.2 A safe-range reformulation

Now we want to show that the reformulation computed by the above tableau based method under the condition of Theorem 5.20 generates a ground safe-range query.

Theorem 5.27 (**Ground safe-range reformulation**). Let $\mathcal{KB}$ be an ontology, and let $\mathcal{Q}(\bar{x})$ be a query which is implicitly definable from $\mathbb{P}_{\mathcal{DB}}$. If $\mathcal{KB}$ and $\mathcal{Q}(\bar{x})$ are safe-range then a reformulation $\widehat{\mathcal{Q}}(\bar{x})$ obtained using the tableau method described in Section 5.5.1 is ground safe-range.

In other words, the conditions of Theorem 5.27 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 signature* σ' as the formula:

$$Adom_{\sigma'}(x) := \bigvee_{P \in \mathbb{P} \cap \sigma'} (\exists x_1, \ldots, x_{\mathrm{AR}(P)-1}.\, P(x, x_1, \ldots, x_{\mathrm{AR}(P)-1}) \vee \ldots \vee \vee P(x_1, \ldots, x_{\mathrm{AR}(P)-1}, x)) \vee \bigvee_{c \in \mathbb{C} \cap \sigma'} (x = c).$$

If $\sigma' = \sigma(\phi)$, where ϕ is a formula, then instead of $Adom_{\sigma(\phi)}$ we simply write $Adom_{\phi}$ and call it *active domain predicate of the formula* ϕ.

Theorem 5.28 (**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

$$\mathcal{KB} \models \forall x_1, \ldots, x_n.\, \mathcal{Q}(x_1, \ldots, x_n) \rightarrow Adom_{\mathcal{Q}}(x_1) \wedge \ldots \wedge Adom_{\mathcal{Q}}(x_n).$$

Given a safe-range ontology, a safe-range and implicitly definable query is obviously domain independent with respect to the ontology (by definition). In this case, Theorem 5.28 says that the answer to the reformulation can only include semantic active domain elements of the reformulation. Therefore, the active domain predicate of the reformulation can be used as a "guard" for free variables which are not bounded by any positive predicate.
Based on Theorem 5.27 and Theorem 5.28, we propose a complete procedure to construct a safe-range reformulation in Algorithm 3.

Algorithm 3 Safe-range reformulation

Input: a safe-range $\mathcal{KB}$, a safe-range and implicitly definable query $\mathcal{Q}(\bar{x})$.
Output: an exact safe-range reformulation $\widehat{\mathcal{Q}}(\bar{x})$.

1: Compute the interpolant $\widehat{\mathcal{Q}}(\bar{x})$ as in Theorem 5.24
2: For each free variable x which is not bounded by any positive predicate in $\widehat{\mathcal{Q}}(\bar{x})$ do
$\widehat{\mathcal{Q}}(\bar{x}) := \widehat{\mathcal{Q}}(\bar{x}) \wedge Adom_{\widehat{\mathcal{Q}}}(x)$
3: Return $\widehat{\mathcal{Q}}(\bar{x})$

5.6 The Guarded Negation Fragment of $\mathcal{ALCHOI}$ and Safe-Range Fragment of $\mathcal{SHOQ}$

In this section, we present an application of Theorem 5.20 in the $\mathcal{ALCHOI}_{GN}$ description logic, the *guarded negation* syntactic fragment of $\mathcal{ALCHOI}$ (Figure 5.2), and $\mathcal{SHOQ}_{GN^+}$, the *extended guarded negation* syntactic fragment of $\mathcal{SHOQ}$ (Figure 5.3) introduced and studied in details in Chapter 4. Each of these fragments happens to express *exactly* the domain independent concepts and TBoxes of the corresponding description logic (see Theorem 4.17 and Theorem 4.24). Recall that $\mathcal{ALCHOI}_{GN}$ and $\mathcal{SHOQ}_{GN^+}$ restrict $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$ respectively by just prescribing that negated concepts should be *guarded* by some generalised atom – an atomic concept, a nominal, an unqualified existential restriction (for $\mathcal{ALCHOI}$) or an unqualified *atleast* number restriction (for $\mathcal{SHOQ}$), i.e., absolute negation is forbidden. $\mathcal{ALCHOI}_{GN}$ is actually at the intersection of the GNFO fragment and $\mathcal{ALCHOI}$ (by definition).

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

Figure 5.2 – Syntax of $\mathcal{ALCHOI}_{GN}$ concepts and roles

$$\begin{array}{lll} B & ::= & A \mid \{o\} \mid \geq nP \\ C & ::= & B \mid \geq nP.C \mid \geq nP.\neg C \mid B \sqcap \neg C \mid C \sqcap D \mid C \sqcup D \end{array}$$

Figure 5.3 – Syntax of $\mathcal{SHOQ}_{GN^+}$ concepts

Each of these fragments has the very important property of *coinciding* with (being equally expressive to) the domain independent and safe-range fragments of the

corresponding description logic (Theorem 4.17, Theorem 4.16, Theorem 4.24 and Theorem 4.23), therefore providing an excellent candidate language for ontologies and queries satisfying the conditions of Theorem 5.20.
To be precise, Theorem 4.17 says that any domain independent TBox axiom and any domain independent concept query in $\mathcal{ALCHOI}$ is logically equivalent, respectively, to a TBox axiom and a concept query in $\mathcal{ALCHOI}_{GN}$, and vice-versa. And Theorem 4.24 says that any domain independent TBox axiom and any domain independent concept query in $\mathcal{SHOQ}$ is logically equivalent, respectively, to a TBox axiom and a concept query in $\mathcal{SHOQ}_{GN^+}$, and vice-versa. These theorems state in particular that a database query can be formulated in one language if and only if it can be expressed in the other.
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`" ($\neg\,\texttt{male} \sqsubseteq \texttt{female}$) – 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`" ($\texttt{person} \sqcap \neg\,\texttt{male} \sqsubseteq \texttt{female}$).
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 unqualified existential restriction (in the case of $\mathcal{ALCHOI}$) or unqualified *atleast* number restriction (in the case of $\mathcal{SHOQ}$). Therefore, the user is asked to make explicit the *type* of that concept, in a way to make it domain independent (i.e. belonging to $\mathcal{ALCHOI}_{GN}$ or $\mathcal{SHOQ}_{GN^+}$). Note that the type could be also a fresh new atomic concept. We believe that the fix we are proposing for $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$ is a reasonable one, and would make all $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$ ontologies eligible to be used with our framework.

5.6.1 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. We consider applications of the Constructive Theorem 5.20 in the fragments $\mathcal{ALCHOI}_{GN}$ and $\mathcal{SHOQ}_{GN^+}$. In this context, the database is a DBox, the ontology is an $\mathcal{ALCHOI}_{GN}$ ($\mathcal{SHOQ}_{GN^+}$) TBox, and the query is an $\mathcal{ALCHOI}_{GN}$ ($\mathcal{SHOQ}_{GN^+}$) *concept query*. A concept query is either an $\mathcal{ALCHOI}_{GN}$ ($\mathcal{SHOQ}_{GN^+}$) concept expression denoting an open formula with one free variable, or an $\mathcal{ALCHOI}_{GN}$ ($\mathcal{SHOQ}_{GN^+}$) ABox concept assertion de-

noting 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 the Theorem 4.17 and the Theorem 4.24 one can draw the following corollaries.

Corollary 5.29. $\mathcal{ALCHOI}_{GN}$ TBoxes and concept queries are domain independent.

Corollary 5.30. $\mathcal{SHOQ}_{GN^+}$ TBoxes and concept queries are domain independent.

We also proved the following theorems.

Theorem 5.31. $\mathcal{ALCHOI}_{GN}$ TBoxes have finitely controllable determinacy of concept queries.

Theorem 5.32. $\mathcal{SHOQ}_{GN^+}$ TBoxes have finitely controllable determinacy of concept queries.

Therefore, we satisfy the conditions of Theorem 5.20, with a language which is like the very expressive $\mathcal{ALCHOI}$ description logic, but with *guarded* negation. And we also satisfy the conditions of Theorem 5.20, with a language which is like the very expressive $\mathcal{SHOQ}$ description logic, but with *extended guarded* negation ("extended" here means that cardinality restrictions and transitivity axioms are allowed in $\mathcal{SHOQ}_{GN^+}$ in spite of the fact that they are not expressible in GNFO).

5.6.2 A complete procedure

$\mathcal{ALCHOI}_{GN}$ and $\mathcal{SHOQ}_{GN^+}$ are decidable logics (as a fragments of $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$ respectively) and they are feasible applications of our general framework. Given an $\mathcal{ALCHOI}_{GN}$ ($\mathcal{SHOQ}_{GN^+}$) ontology $\mathcal{KB}$ and a concept query $\mathcal{Q}$ in $\mathcal{ALCHOI}_{GN}$ ($\mathcal{SHOQ}_{GN^+}$), we can apply the procedure below to generate a safe-range reformulation over the DBox concepts and roles (based on the constructive theorem, all the conditions of which are satisfied), if it exists.
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, Franconi, & Seylan, 2011, 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.

Input: An $\mathcal{ALCHOI}_{GN}$ ($\mathcal{SHOQ}_{GN^+}$) TBox $\mathcal{KB}$, a concept query $\mathcal{Q}$ in $\mathcal{ALCHOI}_{GN}$ ($\mathcal{SHOQ}_{GN^+}$), and a DBox signature (DBox atomic concepts and roles).

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

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 ($\mathcal{ALCHOI}_{GN}$ and $\mathcal{SHOQ}_{GN^+}$ are sublanguages 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 5.5). This can be implemented as a simple extension of a standard description logic 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 et al. (2011).

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 DBox).

5.7 Proofs of Section 5.1

5.7.1 Proposition 5.2

Proof. Let

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

and

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

Since SNA is stricter than UNA, i.e. $I(\mathit{SNA}) \subseteq I(\mathit{UNA})$, 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 $\bar{\mathcal{I}} \not\models \mathcal{Q}_{[\bar{x}/\bar{\Theta}]}$. 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. $\bar{\mathcal{J}}$ embeds $\mathcal{DB}$ and $\bar{\mathcal{J}}$ also is a model of $\mathcal{KB}$. Since first-order logic sentences cannot distinguish two isomorphic structures, $\bar{\mathcal{J}} \not\models \mathcal{Q}_{[\bar{x}/\bar{\Theta}]}$, which contradicts with the assumption $\bar{\Theta} \in A_{sna}$. Therefore $\bar{\Theta} \in A_{una}$.

The proposition is proved. □

5.8 Proofs of Section 5.2

5.8.1 Theorem 5.12

Proof. We need to prove that for any GNFO ontology $\mathcal{KB}$, any GNFO query $\mathcal{Q}(\bar{x})$ that satisfies one of the mentioned conditions, and any set of DBox predicates $\mathbb{P}_{\mathcal{DB}}$, whenever the query is finitely determined by the DBox predicates under the ontology then it is also determined in unrestricted models.
Suppose that $\mathcal{Q}(\bar{x})$ is finitely determined by $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$. Then from Theorem 5.10 it follows that $\mathcal{KB} \cup \widetilde{\mathcal{KB}} \models_{fin\_\mathbb{P}_{\mathcal{DB}}} \forall \bar{x}.\, \mathcal{Q}(\bar{x}) \rightarrow \widetilde{\mathcal{Q}}(\bar{x})$, where $\models_{fin\_\mathbb{P}_{\mathcal{DB}}}$ means entailment over models with a finite interpretation of the DBox predicates. Hence, in particular $\mathcal{KB} \cup \widetilde{\mathcal{KB}} \models_{fin} \forall \bar{x}.\, \mathcal{Q}(\bar{x}) \rightarrow \widetilde{\mathcal{Q}}(\bar{x})$, where $\models_{fin}$ means entailment over finite models. Hereafter let τ be a conjunction of all sentences in $\mathcal{KB}$. Then from the aforementioned entailment we have:

$$\models_{fin} (\neg\tau \vee \neg\widetilde{\tau}) \vee (\neg\exists\bar{x}.\, \mathcal{Q}(\bar{x}) \wedge \neg\widetilde{\mathcal{Q}}(\bar{x})). \tag{5.2}$$

Let $\phi(\bar{x}) := \mathcal{Q}(\bar{x}) \wedge \neg\widetilde{\mathcal{Q}}(\bar{x})$. Let us prove that in any of the three mentioned cases $\phi(\bar{x})$ is in GNFO.

- Suppose that $\mathcal{Q}(\bar{x})$ is answer-guarded formula in GNFO. That is $\mathcal{Q}(\bar{x}) = Atom(\bar{x}) \wedge \varphi(\bar{x})$. Then $\varphi(\bar{x})$ or $\neg\varphi(\bar{x})$ is expressed in GNFO by definition of GNFO fragment. $\phi(\bar{x}) = \mathcal{Q}(\bar{x}) \wedge \neg\widetilde{\mathcal{Q}}(\bar{x}) = Atom(\bar{x}) \wedge \varphi(\bar{x}) \wedge \neg(\widetilde{Atom}(\bar{x}) \wedge$

$\widetilde{\varphi}(\bar{x})) = (Atom(\bar{x}) \wedge \varphi(\bar{x})) \wedge (Atom(\bar{x}) \wedge \neg(\widetilde{Atom}(\bar{x}) \wedge \widetilde{\varphi}(\bar{x}))) = \mathcal{Q}(\bar{x}) \wedge (Atom(\bar{x}) \wedge \neg\widetilde{\mathcal{Q}}(\bar{x})) \in$GNFO.

– Suppose that $\mathcal{Q}(\bar{x})$ is boolean query in GNFO. That is $\bar{x} = \emptyset$. Then $\phi(\bar{x})$ is in GNFO by definition of the fragment.

– Suppose that $\mathcal{Q}(\bar{x})$ is GNFO formula with one free variable ($\bar{x}$ consists of just one variable). Then $\phi(\bar{x})$ is in GNFO because any first-order formula with one free variable is always in GNFO (since $x = x$ can always be considered as a guard for negated formulas by definition of GNFO).

Since all the sentences in $\mathcal{KB}$ are in GNFO, 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 (5.2) is in GNFO. Then $\neg((\neg\tau \vee \neg\widetilde{\tau}) \vee (\neg\exists\bar{x}.\, \mathcal{Q}(\bar{x}) \wedge \neg\widetilde{\mathcal{Q}}(\bar{x})))$is also in GNFO and by the entailment (5.2) does not have a finite model. Then, since GNFO has the finite model property, $\neg((\neg\tau \vee \neg\widetilde{\tau}) \vee (\neg\exists\bar{x}.\, \mathcal{Q}(\bar{x}) \wedge \neg\widetilde{\mathcal{Q}}(\bar{x})))$ is unsatisfiable. Hence, we have:

$$\models (\neg\tau \vee \neg\widetilde{\tau}) \vee (\neg\exists\bar{x}.\, \mathcal{Q}(\bar{x}) \wedge \neg\widetilde{\mathcal{Q}}(\bar{x})). \tag{5.3}$$

Then $\mathcal{KB} \cup \widetilde{\mathcal{KB}} \models \forall\bar{x}.\, \mathcal{Q}(\bar{x}) \rightarrow \widetilde{\mathcal{Q}}(\bar{x})$. By Theorem 5.10 it means that the query $\mathcal{Q}(\bar{x})$ is determined in unrestricted models by the DBox predicates $\mathbb{P}_{\mathcal{DB}}$ under the ontology $\mathcal{KB}$.

The theorem is proved. □

5.9 Proofs of Section 5.3

5.9.1 Proposition 5.14

Proof. Since $\widehat{\mathcal{Q}}(\bar{x})$ is an exact reformulation of $\mathcal{Q}(\bar{x})$, $\mathcal{KB} \models \forall\bar{x}.\, \mathcal{Q}(\bar{x}) \leftrightarrow \widehat{\mathcal{Q}}(\bar{x})$. Then, for any model $\mathcal{I} = \langle\Delta, \cdot^{\mathcal{I}}\rangle \in M(\mathcal{KB})$ and for any substitution $\Theta : \bar{x} \mapsto \Delta^{\mathcal{I}}$ we have: $\mathcal{I}, \Theta \models \mathcal{Q}(\bar{x}) \leftrightarrow \widehat{\mathcal{Q}}(\bar{x})$, which is equivalent to $(\mathcal{I}, \Theta \models \mathcal{Q}(\bar{x}) \iff \mathcal{I}, \Theta \models \widehat{\mathcal{Q}}(\bar{x}))$.
Now, let $\bar{\Theta}$ be any substitution from $\{\Theta \mid \mathit{dom}(\Theta) = \bar{x},\ \mathit{rng}(\Theta) = \mathbb{C},\ \forall\mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB}) : \mathcal{I} \models \mathcal{Q}(\bar{x}/\Theta)\}$, 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. $\tilde{\Theta}(x) = a \in \Delta$ if and only if $\bar{\Theta}(x) = c \in \mathbb{C}$ and $c^{\mathcal{I}} = a$). Then $\mathcal{I}, \tilde{\Theta} \models \mathcal{Q}(\bar{x}) \iff \mathcal{I} \models \mathcal{Q}(\bar{x}/\bar{\Theta})$ and $\mathcal{I}, \tilde{\Theta} \models \widehat{\mathcal{Q}}(\bar{x}) \iff \mathcal{I} \models \widehat{\mathcal{Q}}(\bar{x}/\bar{\Theta})$. Summing up: $\mathcal{I} \models \mathcal{Q}(\bar{x}/\bar{\Theta}) \iff \mathcal{I} \models \widehat{\mathcal{Q}}(\bar{x}/\bar{\Theta})$.

Hence, $\bar{\Theta} \in \{\Theta \mid dom(\Theta) = \bar{x},\ rng(\Theta) = \mathbb{C},\ \forall \mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB}) : \mathcal{I} \models \widehat{\mathcal{Q}}(\bar{x}/\Theta)\}$. The inverse inclusion can be proved similarly.

The proposition is proved. □

5.9.2 Theorem 5.15

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

Proposition 5.33 (Domain independence). A query $\mathcal{Q}(\bar{x})$ is domain independent if and only if 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 $\Theta : \bar{x} \mapsto \Delta^{\mathcal{I}} \cup \Delta^{\mathcal{J}}$ we have:

$$rng(\Theta) \subseteq \Delta^{\mathcal{I}} \text{ and } \mathcal{I}, \Theta \models \mathcal{Q}(\bar{x}) \quad \text{iff}$$
$$rng(\Theta) \subseteq \Delta^{\mathcal{J}} \text{ and } \mathcal{J}, \Theta \models \mathcal{Q}(\bar{x}).$$

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 $\Theta : \bar{x} \mapsto \Delta^{\mathcal{I}} \cup \Delta^{\mathcal{J}}$ (if the query is closed, we just omit everything that concerns a substitution below in the proof) such that:

$$rng(\Theta) \subseteq \Delta^{\mathcal{I}} \text{ and } \mathcal{I}, \Theta \models \mathcal{Q}(\bar{x}). \tag{5.4}$$

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 $\mathcal{Q}(\bar{x})$ equally. Therefore, since $\mathcal{I}, \Theta \models \mathcal{Q}(\bar{x})$ (by (5.4)), $\mathcal{I}', \Theta \models \mathcal{Q}(\bar{x})$.
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

$$rng(\Theta) \subseteq \Delta^{\mathcal{I}} \text{ and } \mathcal{I}', \Theta \models \mathcal{Q}(\bar{x}), \tag{5.5}$$

we have that

$$rng(\Theta) \subseteq \Delta^{\mathcal{J}} \text{ and } \mathcal{J}', \Theta \models \mathcal{Q}(\bar{x}). \tag{5.6}$$

Then again interpretations $\mathcal{J}$ and $\mathcal{J}'$ have the same domain and interpret all the predicates and constants, occurring in $\mathcal{Q}(\bar{x})$ equally. Thus, because of (5.6),

$$rng(\Theta) \subseteq \Delta^{\mathcal{J}} \text{ and } \mathcal{J}, \Theta \models \mathcal{Q}(\bar{x}). \tag{5.7}$$

Therefore, (5.4) $\Longrightarrow$ (5.7). Similarly (5.7) $\Longrightarrow$ (5.4), and the proposition is proved. □

Proposition 5.34. If $\mathcal{Q}(\bar{x})$ is domain independent, then for any interpretation $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ and any substitution $\Theta : \bar{x} \mapsto \Delta$, such that $\mathcal{I}, \Theta \models \mathcal{Q}(\bar{x})$, the following holds:

$$rng(\Theta) \subseteq adom(\sigma(\mathcal{Q}(\bar{x})), \mathcal{I}).$$

Proof. Assume that $\bar{x} = \{x\}$, 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 $\mathcal{I}', \{x \to b\} \models \mathcal{Q}(\bar{x})$, 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 (are compatible). 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 $a \notin \Delta$.
The proposition is proved. □

Now we prove the theorem itself.

$$\begin{aligned} L := \{\Theta \mid\ & dom(\Theta) = \bar{x},\ rng(\Theta) \subseteq \mathbb{C},\ \forall \mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB}) : \\ & \mathcal{I} \models \mathcal{Q}(\bar{x}/\Theta)\}; \\ R := \{\Theta \mid\ & dom(\Theta) = \bar{x},\ rng(\Theta) \subseteq adom(\sigma(\widehat{\mathcal{Q}}), \mathcal{DB}), \\ & \forall \mathcal{I} = \langle \mathbb{C}, \cdot^{\mathcal{I}} \rangle \in E(\mathcal{DB}) : \mathcal{I}|_{\mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}} \models \widehat{\mathcal{Q}}(\bar{x}/\Theta)\} \end{aligned}$$

Let $\bar{\Theta} \in L$. Then for any $\mathcal{I} \in M(\mathcal{KB}) \cap E(\mathcal{DB})$ we have: $\mathcal{I} \models \mathcal{Q}(\bar{x}/\bar{\Theta})$ and $\mathcal{I} \models \widehat{\mathcal{Q}}(\bar{x}/\bar{\Theta})$, because of Proposition 5.14.
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 $\widehat{\mathcal{Q}}(\bar{x})$ is domain independent, by Proposition 5.33 we have: $\mathcal{J} \models \widehat{\mathcal{Q}}(\bar{x}/\bar{\Theta})$. Since $\sigma(\widehat{\mathcal{Q}}(\bar{x})) \subseteq \mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}$, $\mathcal{J}|_{P_{\mathcal{DB}} \cup \mathbb{C}} \models \widehat{\mathcal{Q}}(\bar{x}/\bar{\Theta})$. Since $\widehat{\mathcal{Q}}(\bar{x})$ is domain independent, by Proposition 5.34 we have:
$rng(\bar{\Theta}) \subseteq adom(\sigma(\widehat{\mathcal{Q}}(\bar{x})), \mathcal{J})$. $adom(\sigma(\widehat{\mathcal{Q}}(\bar{x})), \mathcal{J}) = adom(\sigma(\widehat{\mathcal{Q}}(\bar{x})), \mathcal{DB})$, because we assume SNA and $\sigma(\widehat{\mathcal{Q}}(\bar{x})) \subseteq \mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}$. Therefore,

$$rng(\bar{\Theta}) \subseteq adom(\sigma(\widehat{\mathcal{Q}}(\bar{x})), \mathcal{DB}).$$

Then $\bar{\Theta} \in R$ and, hence, $L \subseteq R$.
Let $\bar{\Theta} \in R$. That is, $rng(\bar{\Theta}) \subseteq adom(\sigma(\widehat{\mathcal{Q}}(\bar{x})), \mathcal{DB})$. Then for any $\mathcal{J} = \langle \mathbb{C}, \cdot^{\mathcal{I}} \rangle$ embedding $\mathcal{DB}$ we have: $\mathcal{J}|_{P_{\mathcal{DB}} \cup \mathbb{C}} \models \widehat{\mathcal{Q}}(\bar{x}/\bar{\Theta})$. Then $\mathcal{J} \models \widehat{\mathcal{Q}}(\bar{x}/\bar{\Theta})$. 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 $\sigma(\widehat{\mathcal{Q}}(\bar{x}/\bar{\Theta})) \subseteq \mathbb{P}_{\mathcal{DB}} \cup \mathbb{C}$ and $\widehat{\mathcal{Q}}(\bar{x})$ is domain independent, by Proposition 5.33 we have: $\mathcal{I} \models \widehat{\mathcal{Q}}(\bar{x}/\bar{\Theta})$. Since $\widehat{\mathcal{Q}}(\bar{x})$ is exact reformulation of $\mathcal{Q}(\bar{x})$ under $\mathcal{KB}$ over $\mathbb{P}_{\mathcal{DB}}$, by Proposition 5.14 we have: $\mathcal{I} \models \mathcal{Q}(\bar{x}/\bar{\Theta})$. Then $\bar{\Theta} \in L$ and, hence, $R \subseteq L$.
Note that if $\mathcal{DB}$ is not legal for $\mathcal{KB}$, that is $M(\mathcal{KB}) \cap E(\mathcal{DB} = \emptyset$, then $L = \{\Theta \mid dom(\Theta) = \bar{x}, rng(\Theta) \subseteq \mathbb{C}\} \neq R$, because for any substitution $\bar{\Theta}$ from R, $rng(\bar{\Theta}) \subseteq adom(\sigma(\widehat{\mathcal{Q}})$, which, in general, is not the case for substitutions from L.
The theorem is proved. □

5.10 Definitions and Proofs of Section 5.4

First we need to prove some auxiliary propositions and lemmas.

Proposition 5.35. 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 $\mathcal{KB}$.

The proposition is proved. □

Proposition 5.36. Let $\mathcal{KB}$ be an ontology, and let $\mathcal{Q}(\bar{x})$ be a query which is domain independent with respect to $\mathcal{KB}$. Any exact reformulation of $\mathcal{Q}(\bar{x})$ under $\mathcal{KB}$ (over any set of predicates) is also domain independent with respect to $\mathcal{KB}$.

Proof. Let $\widehat{\mathcal{Q}}(\bar{x})$ be any exact reformulation of $\mathcal{Q}(\bar{x})$ 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 $\Theta : \bar{x} \mapsto \Delta^{\mathcal{I}} \cup \Delta^{\mathcal{J}}$ be any substitution such that $rng(\Theta) \subseteq \Delta^{\mathcal{I}}$ and $\mathcal{I}, \Theta \models \widehat{\mathcal{Q}}(\bar{x})$. Then, since $\widehat{\mathcal{Q}}(\bar{x})$ is exact reformulation of $\mathcal{Q}(\bar{x})$, we have: $\mathcal{I}, \Theta \models \mathcal{Q}(\bar{x})$. Then, since $\mathcal{Q}(\bar{x})$ is domain independent with respect to $\mathcal{KB}$, we have: $rng(\Theta) \subseteq \Delta^{\mathcal{J}}$ and $\mathcal{J}, \Theta \models \mathcal{Q}(\bar{x})$. And again, since $\widehat{\mathcal{Q}}(\bar{x})$ is exact reformulation of $\mathcal{Q}(\bar{x})$, we have: $\mathcal{J}, \Theta \models \widehat{\mathcal{Q}}(\bar{x})$. Thus, $\widehat{\mathcal{Q}}(\bar{x})$ is domain independent with respect to $\mathcal{KB}$ by definition.

The proposition is proved. □

Lemma 5.37. Let $\mathcal{KB}$ be a domain independent ontology, and let $\mathcal{Q}(\bar{x})$ 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 $\Theta : \bar{x} \mapsto \Delta$ such that $\mathcal{I}, \Theta \models \mathcal{Q}(\bar{x})$ the following holds:

$$rng(\Theta) \subseteq adom(\sigma(\mathcal{Q}(\bar{x})), \mathcal{I}).$$

Proof. Without loss of generality assume that $\bar{x} = \{x\}$, 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 $b \in \Delta \setminus adom(\sigma(\mathcal{Q}(\bar{x})), \mathcal{I})$. 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 5.35. 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. □

Recall the definition of the active domain predicate of a formula (Subsection 5.5.2).

Observation 5.38.

1. For any query $\mathcal{Q}(\bar{x})$ and any interpretation $\mathcal{I} = \langle \Delta, \cdot^{\mathcal{I}} \rangle$ the following holds:

$$Adom^{\mathcal{I}}_{\mathcal{Q}} = adom(\sigma(\mathcal{Q}(\bar{x})), \mathcal{I}).$$

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

Let us denote for the sake of readability: $Adom_{\mathcal{KB}\cup\mathcal{Q}} := Adom_{\sigma(\mathcal{KB})\cup\sigma(\mathcal{Q})}$.

Lemma 5.39. Let $\mathcal{KB}$ be a domain independent ontology, and let $\mathcal{Q}(\bar{x})$ be a query which is domain independent with respect to $\mathcal{KB}$. Then the following holds:

$$\mathcal{KB} \models \forall \bar{x}.\, \mathcal{Q}(\bar{x}) \leftrightarrow \mathcal{Q}(\bar{x})|_{Adom_{\mathcal{KB}\cup\mathcal{Q}}},$$

where $\mathcal{Q}(\bar{x})|_{Adom_{\mathcal{KB}\cup\mathcal{Q}}}$ is a relativisation of the query $\mathcal{Q}(\bar{x})$ to $Adom_{\mathcal{KB}\cup\mathcal{Q}}$.

Proof. The lemma is proved analogously to Theorem 3.25. □

5.10.1 Theorem 5.19

Proof. The theorem can be proved after Theorem 5.20.

– The "if" direction.
Based on Lemma 5.39, one can see that exact reformulations of $\mathcal{Q}(\bar{x})$ are also exact reformulations of $\mathcal{Q}(\bar{x})|_{Adom_{\mathcal{KB}\cup\mathcal{Q}}}$. Since by Theorem 3.23 $\mathcal{Q}(\bar{x})|_{Adom_{\mathcal{KB}\cup\mathcal{Q}}}$ 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 $\widehat{\mathcal{Q}}(\bar{x})$ found in Theorem 5.20 which takes $\mathcal{KB}'$ and $\mathcal{Q}(\bar{x})|_{Adom_{\mathcal{KB}\cup\mathcal{Q}}}$ as its input is the exact domain independent reformulation of $\mathcal{Q}(\bar{x})$.

– The "only if" direction.
Suppose that there exists an exact domain independent reformulation $\widehat{\mathcal{Q}}(\bar{x})$ of $\mathcal{Q}(\bar{x})$ over $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$. Then it is domain independent with respect to $\mathcal{KB}$. Hence, by Proposition 5.36, $\mathcal{Q}(\bar{x})$ is domain independent with respect to $\mathcal{KB}$. Since there exists an exact reformulation of $\mathcal{Q}(\bar{x})$, $\mathcal{Q}(\bar{x})$ is implicitly definable from $\mathbb{P}_{\mathcal{DB}}$ under $\mathcal{KB}$ by the Theorem 5.9.

The theorem is proved. □

5.10.2 Theorem 5.20

Proof. The theorem can be proved after Theorem 5.27 and Theorem 5.28. We will use the following lemma in the proof.

Lemma 5.40. If $\mathcal{KB}$ is an ontology, $\mathcal{Q}(\bar{x})$ ($\bar{x} = \{x_1, \ldots, x_n\}$) is ground safe-range query and

$$\mathcal{KB} \models \forall \mathbb{X}.\, \mathcal{Q}(\bar{x}) \rightarrow \psi_1(x_1) \wedge \ldots \wedge \psi_n(x_n), \tag{5.8}$$

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

Proof. Let $\mathcal{Q}'(\bar{x})$ be a safe-range normal form of the query $\mathcal{Q}(\bar{x})$, i.e. $\mathcal{Q}'(\bar{x}) := \textsc{srnf}(\mathcal{Q}(\bar{x})) = \exists \bar{y}.\, \phi(\bar{x} \cup \bar{y})$, where $\phi(\bar{x} \cup \bar{y})$ is in conjunctive normal form (the safe-range normal form of the query is in prenex normal form). Then $\mathcal{Q}'(\bar{x})$ is ground safe-range, and $\mathcal{KB} \models \mathcal{Q}'(\bar{x}) \leftrightarrow \mathcal{Q}(\bar{x})$. Hence, $\mathcal{KB} \models \forall \bar{x}.\, \mathcal{Q}'(\bar{x}) \rightarrow \psi_1(x_1) \wedge \ldots \wedge \psi_n(x_n)$. Let $\mathcal{Q}''(\bar{x}) := \mathcal{Q}'(\bar{x}) \wedge \psi'_1(x_1) \wedge \ldots \wedge \psi'_n(x_n)$, where each $\psi'_i(x_i) = \textsc{srnf}(\psi_i(x_i))$. Then by 5.8, $\mathcal{KB} \models \mathcal{Q}'(\bar{x}) \leftrightarrow \mathcal{Q}''(\bar{x})$. On the other hand $\mathcal{KB} \models \forall \bar{x}.\, \widehat{\mathcal{Q}}(\bar{x}) \leftrightarrow \mathcal{Q}''(\bar{x})$ by construction. Summing up everything, we have: $\mathcal{KB} \models \widehat{\mathcal{Q}}(\bar{x}) \leftrightarrow \mathcal{Q}(\bar{x})$ and the only thing we need to prove is that $\widehat{\mathcal{Q}}(\bar{x})$ is safe-range.
One can see that $\mathcal{Q}''(\bar{x}) \equiv \exists \bar{y}.\, (\phi(\bar{x} \cup \bar{y}) \wedge \psi'_1(x_1) \wedge \ldots \wedge \psi'_n(x_n))$ which is a safe-range normal form of $\widehat{\mathcal{Q}}(\bar{x})$. Since $\mathcal{Q}'(\bar{x}) = \exists \bar{y}.\, \phi(\bar{x} \cup \bar{y})$ is ground safe-range, then $rr(\phi(\bar{x} \cup \bar{y})) \setminus \bar{x} = \bar{y}' \subseteq \bar{y}$, where for any $y \in \bar{y} \setminus \bar{y}'$ there exists a conjunct $x = y$ in $\phi(\bar{x} \cup \bar{y})$, for some $x \in \bar{x}$. Then, since each $\psi'_i(x_i)$ is safe-range, by definition of range restriction $rr(\phi(\bar{x} \cup \bar{y}) \wedge \psi'_1(x_1) \wedge \ldots \wedge \psi'_n(x_n)) = \bar{x} \cup \bar{y}$, and then $rr(\exists \bar{y}.\, (\phi(\bar{x} \cup \bar{y}) \wedge \psi'_1(x_1) \wedge \ldots \wedge \psi'_n(x_n))) = \bar{x} = \textsc{free}(\mathcal{Q}''(\bar{x}))$. Therefore, $\exists \bar{y}.\, (\phi(\bar{x} \cup \bar{y}) \wedge \psi'_1(x_1) \wedge \ldots \wedge \psi'_n(x_n))$ is safe-range by definition, and hence $\widehat{\mathcal{Q}}(\bar{x})$ is safe-range.

The lemma is proved. □

Let us continue to prove the theorem.
If $\bar{x} = \emptyset$, ($\mathcal{Q}$ is closed) then we build an exact safe-range reformulation $\widehat{\mathcal{Q}}$ by using Theorem 5.27.
Suppose now, $\bar{x} = \{x_1, \ldots, x_n\}$. Since $\mathcal{Q}(\bar{x})$ is safe-range and implicitly definable from $\mathbb{P}_{\mathcal{DB}}$, we apply Theorem 5.27 for $\mathcal{Q}(\bar{x})$ and construct a ground safe-range rewriting $\mathcal{Q}'(\bar{x})$ expressed over $\mathbb{P}_{\mathcal{DB}}$ such that $\mathcal{KB} \models \forall \bar{x}.\, \mathcal{Q}(\bar{x}) \leftrightarrow \mathcal{Q}'(\bar{x})$. Since $\mathcal{Q}(\bar{x})$ is domain independent (since it is safe-range), it is also domain independent with respect to $\mathcal{KB}$. Hence, by Proposition 5.36, $\mathcal{Q}'(\bar{x})$ is also domain

independent with respect to $\mathcal{KB}$. Moreover, $\mathcal{KB}$ is safe-range and, hence, domain independent. Then by Theorem 5.28:

$$\mathcal{KB} \models \forall \bar{x}.\, \mathcal{Q}'(\bar{x}) \rightarrow Adom_{\mathcal{Q}}(x_1) \wedge \ldots \wedge Adom_{\mathcal{Q}}(x_n).$$

By the second item of Observation 5.38 $Adom_{\mathcal{Q}}(x)$ is a safe-range formula. Then by Lemma 5.40 the query $\widehat{\mathcal{Q}}(\bar{x}) := \mathcal{Q}'(\bar{x}) \wedge Adom_{\mathcal{Q}'}(x_1) \wedge \ldots \wedge Adom_{\mathcal{Q}'}(x_n)$ is safe-range and $\mathcal{KB} \models \forall \bar{x}.\, \mathcal{Q}'(\bar{x}) \leftrightarrow \widehat{\mathcal{Q}}(\bar{x})$. Since $\mathcal{KB} \models \forall \bar{x}.\, \mathcal{Q}(\bar{x}) \leftrightarrow \mathcal{Q}'(\bar{x})$, we have: $\mathcal{KB} \models \forall \bar{x}.\, \mathcal{Q}(\bar{x}) \leftrightarrow \widehat{\mathcal{Q}}(\bar{x})$. Therefore, the constructed query $\widehat{\mathcal{Q}}(\bar{x})$ is the one we were looking for.

The theorem is proved. □

5.11 Proofs of Section 5.5

5.11.1 Theorem 5.24

Proof. First we will prove that if $\mathcal{Q}(\bar{x})$ is implicitly definable then the following formula is valid:

$$((\bigwedge \mathcal{KB}) \wedge \mathcal{Q}(\bar{x}/c_1, \ldots, c_n)) \rightarrow ((\bigwedge \widetilde{\mathcal{KB}}) \rightarrow \widetilde{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n)) \qquad (5.9)$$

Applying syntactic definition of implicit definability (Theorem 5.10), we have: $\mathcal{KB} \cup \widetilde{\mathcal{KB}} \models \forall \bar{x}.\mathcal{Q}(\bar{x}) \leftrightarrow \widetilde{\mathcal{Q}}(\bar{x})$. Therefore, when we replace $\bar{x}$ by a set of constants $c_1, \ldots, c_n$, the following formula is valid:

$$(\bigwedge \mathcal{KB} \wedge \bigwedge \widetilde{\mathcal{KB}}) \rightarrow (\mathcal{Q}(\bar{x}/c_1, \ldots, c_n) \rightarrow \widetilde{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n)).$$

That is, $\neg(\bigwedge \mathcal{KB}) \vee \neg(\bigwedge \widetilde{\mathcal{KB}}) \vee \neg\mathcal{Q}(\bar{x}/c_1, \ldots, c_n) \vee \widetilde{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n)$ is valid. As a consequence, (5.9) is valid.
Next, we have to prove $\mathcal{KB} \models \mathcal{Q}(c_1, \ldots, c_n/\bar{x}) \leftrightarrow \widehat{\mathcal{Q}}(c_1, \ldots, c_n/\bar{x})$ where $\widehat{\mathcal{Q}}(c_1, \ldots, c_n/\bar{x})$ is a Craig interpolant of (5.9). Since $\widehat{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n)$ is an interpolant:

1. $((\bigwedge \mathcal{KB}) \wedge \mathcal{Q}(\bar{x}/c_1, \ldots, c_n)) \rightarrow \widehat{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n)$.
   Then : $\mathcal{KB} \models \mathcal{Q}(\bar{x}/c_1, \ldots, c_n) \rightarrow \widehat{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n)$.

2. $\widehat{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n) \rightarrow ((\bigwedge \widetilde{\mathcal{KB}}) \rightarrow \widetilde{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n))$.
   Then : $\widetilde{\mathcal{KB}} \models \widehat{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n) \rightarrow \widetilde{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n)$.
   Since $\sigma(\widehat{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n)) \subseteq \mathbb{P}_{\mathcal{DB}}$, the entailment
   $\mathcal{KB} \models (\widehat{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n) \rightarrow \mathcal{Q}(\bar{x}/c_1, \ldots, c_n))$ holds as well by construction of $\widetilde{\mathcal{KB}}$ and $\widetilde{\mathcal{Q}}$.

From the items 1 and 2 we have the expected statement.
Last but not least, since $\sigma(\widehat{\mathcal{Q}}(\bar{x}/c_1, \ldots, c_n)) \subseteq \mathbb{P}_{\mathcal{DB}}$ then $\sigma(\widehat{\mathcal{Q}}(c_1, \ldots, c_n/\bar{x})) \subseteq \mathbb{P}_{\mathcal{DB}}$.
With above statements, $\widehat{\mathcal{Q}}_{[c_1,\ldots,c_n/\bar{x}]}$ is really an exact reformulation of $\mathcal{Q}(\bar{x})$ under $\mathcal{KB}$ over $\mathbb{P}_{\mathcal{DB}}$.

The theorem is proved. □

5.11.2 Theorem 5.27

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

Proposition 5.41. $\varphi_1 \wedge \varphi_2$ is safe-range and closed iff φ_1 and φ_2 are safe-range and closed.

Proof. We have:

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

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

The proposition is proved. □

Proposition 5.42. $\varphi_1 \vee \varphi_2$ is safe-range and closed iff φ_1 and φ_2 are safe-range and closed.

Proof. We have:

- $rr(\varphi_1 \vee \varphi_2) = rr(\varphi_1) \cap rr(\varphi_2)$
- $\text{FREE}(\varphi_1 \vee \varphi_2) = \text{FREE}(\varphi_1) \cup \text{FREE}(\varphi_2)$
- $rr(\varphi_1) = \bot$ 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.

The proposition is proved. □

Proposition 5.43. $\forall \bar{x} \varphi(\bar{x})$ is closed and safe-range then $\varphi(\bar{t})$ is closed and safe-range where $\bar{t}$ are constants.

Proof. Obviously, if $\forall \bar{x} \varphi(\bar{x})$ is closed then $\varphi(\bar{t})$ is closed. Let us prove by contradiction. Assume that $\varphi(\bar{t})$ is not safe-range. Since it is closed, $rr(\text{SRNF}(\varphi(\bar{t}))) = \bot$. Then $\text{SRNF}(\varphi(\bar{t}))$ must contain a subformula which is in the form $\exists \bar{z} \varphi'(\bar{t}, \bar{z})$, where $\bar{z} \not\subseteq rr(\text{SRNF}(\varphi'(\bar{t}, \bar{z})))$. Hence, $\text{SRNF}(\varphi(\bar{x}))$ must contain a subformula which is in the form $\exists \bar{z} \varphi'(\bar{x}, \bar{z})$, where $\bar{z} \not\subseteq rr(\text{SRNF}(\varphi'(\bar{x}, \bar{z})))$. Then $\text{SRNF}(\neg\varphi(\bar{x}))$ must contain a subformula which is in the form $\exists \bar{z} \varphi'(\bar{x}, \bar{z})$, where $\bar{z} \not\subseteq rr(\text{SRNF}(\varphi'(\bar{x}, \bar{z})))$ because pushing negation does not effect the formula under $\exists$. Hence, $rr(\text{SRNF}(\neg\varphi(\bar{x}))) = \bot$. Hence, $rr(\text{SRNF}(\neg\exists \bar{x} \neg\varphi(\bar{x}))) = \bot$. Hence, $rr(\text{SRNF}(\forall \bar{x} \varphi(\bar{x}))) = \bot$. And, hence, $\forall \bar{x} \varphi(\bar{x})$ is not safe-range. Contradiction.

The proposition is proved. □

Proposition 5.44. $\exists\bar{x}\varphi(\bar{x})$ is closed and safe-range then $\varphi(\bar{t})$ is closed and safe-range where $\bar{t}$ are constants.

Proof. Undoubtedly, if $\exists\bar{x}\varphi(\bar{x})$ is closed then $\varphi(\bar{t})$ is closed. Let us prove by contradiction. Assume that $\varphi(\bar{t})$ is not safe-range. Since it is closed, $rr(\text{SRNF}(\varphi(\bar{t}))) = \bot$. Then $\text{SRNF}(\varphi(\bar{t}))$ must contain a subformula which is in the form $\exists\bar{z}\varphi'(\bar{t},\bar{z})$, where $\bar{z} \not\subseteq rr(\text{SRNF}(\varphi'(\bar{t},\bar{z})))$. Hence, $\text{SRNF}(\varphi(\bar{x}))$ must contain a subformula which is in the form $\exists\bar{z}\varphi'(\bar{x},\bar{z})$, where $\bar{z} \not\subseteq rr(\text{SRNF}(\varphi'(\bar{x},\bar{z})))$. Then $rr(\text{SRNF}(\varphi(\bar{x}))) = \bot$. Hence, $rr(\text{SRNF}(\exists\bar{x}\varphi(\bar{x}))) = \bot$. Hence, $\exists\bar{x}\varphi(\bar{x})$ is not safe-range. Contradiction.

The proposition is proved. □

Based on these propositions, we prove Theorem 5.27 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 5.5.

- 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 5.42, (**) holds.
  - For the rule (p6), apply the Proposition 5.41,(**) 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 $S \cup \{R(\varphi(c))\} \overset{int}{\rightarrow} I = \neg\varphi(c)$. 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 $\mathcal{Q}(\bar{c})$, $\mathcal{KB}$, $\mathcal{KB}'$,$\neg\mathcal{Q}'(\bar{c})$ are closed and safe-range then so is the interpolant $\widehat{\mathcal{Q}}(\bar{c})$ of $\mathcal{KB} \wedge \mathcal{Q}(\bar{c})$ and $\mathcal{KB}' \rightarrow \mathcal{Q}'(\bar{c})$.

The theorem is proved. □

5.11.3 Theorem 5.28

Proof. As a consequence of Lemma 5.37 and the first item of Observation 5.38, Theorem 5.28 holds. □

5.12 Definitions and Proofs of Section 5.6

5.12.1 Theorem 5.31

Proof. We need to prove that for any $\mathcal{ALCHOI}_{GN}$ TBox $\mathcal{T}$ (ontology), any concept query $\mathcal{Q}$ in $\mathcal{ALCHOI}_{GN}$ and any set of DBox predicates $\mathbb{P}_{\mathcal{DB}}$, whenever the query is finitely determined by the DBox predicates under the ontology then it is also determined in unrestricted models. We can consider $\mathcal{T}$ as a set of sentences in GNFO and $\mathcal{Q}$ as a GNFO formula with one free variable. Then the theorem immediately follows from the third item of Theorem 5.12. □

5.12.2 Theorem 5.32

Proof. We need to prove that for any $\mathcal{SHOQ}$ TBox $\mathcal{T}$ (ontology), any concept query $\mathcal{Q}$ in $\mathcal{SHOQ}$ and any set of DBox predicates $\mathbb{P}_{\mathcal{DB}}$, whenever the query is finitely determined by the DBox predicates under this 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 5.10 it follows that $\mathcal{T} \cup \widetilde{\mathcal{T}} \models_{fin\_\mathbb{P}_{\mathcal{DB}}} \mathcal{Q} \sqsubseteq \widetilde{\mathcal{Q}}$, where $\models_{fin\_\mathbb{P}_{\mathcal{DB}}}$ means entailment over models with a finite interpretation of the DBox predicates. Hence, in particular $\mathcal{T} \cup \widetilde{\mathcal{T}} \models_{fin} \mathcal{Q} \sqsubseteq \widetilde{\mathcal{Q}}$, where $\models_{fin}$ means entailment over finite models. This entailment can be reduced in $\mathcal{SHOQ}$ to a concept unsatisfiability problem over finite models for an empty TBox. Then, since $\mathcal{SHOQ}$ has the finite model

property (Lutz, Areces, Horrocks, & Sattler, 2005), this problem over finite models is equivalent to the same problem over unrestricted models and, hence, the entailment $\mathcal{T} \cup \widetilde{\mathcal{T}} \models \mathcal{Q} \sqsubseteq \widetilde{\mathcal{Q}}$ holds (over unrestricted models). By Theorem 5.10 it means, that the query $\mathcal{Q}$ is determined in unrestricted models by the DBox predicates $\mathbb{P}_{\mathcal{DB}}$ under the ontology $\mathcal{T}$.

The theorem is proved. □

5.13 Related Work

Nash et al. (2010) addressed the question as to whether a query can be answered using a set of exact view definitions by means of a rewriting over the views. The authors defined and investigated the notions of determinacy of a query by a set of views and its connection to (exact) rewriting. Nash et al. (2010) also studied several combinations of query and view languages trying to understand the expressivity of the language required to express the exact rewriting, and, thus, they obtained results in completeness of rewriting languages. They investigated languages ranging from full first-order logic to conjunctive queries. The setting Nash et al. (2010) considered is weaker than ours, since they only use a set of view definitions as an ontology, while we allow any theory with a signature containing view predicates – what we call DBox predicates. Another important difference is that their definition of determinacy and rewriting is based on different semantics for the underlying logical framework. In our definitions we use classical first-order logic interpretations (with arbitrary non-empty domains), while their definitions are based on what they call database instances (Nash et al., 2010), where a database instance is a (possibly finite) relational structure (interpretation) whose universe (domain) is the active domain of the database instance itself (the set of all elements occurring in the database instance plus constant values appearing in the query). This is called active domain semantics in the standard database theory literature (Abiteboul et al., 1995). The most interesting theorem for us is Theorem 3.1 (Nash et al., 2010), saying that first-order logic is complete for FO-to-FO rewritings under exact views. The proof is based on the Relativised Craig's Interpolation theorem proved by Otto (2000). As we mentioned already in Section 3.1 (the last paragraph), considering formulas under active domain semantics is equivalent to considering a domain independent fragment under classical semantics. So, it is possible to show that the theorem 3.1 in Nash et al. (2010) is a special case of our theorem 5.20 for the restricted case of exact view definitions (taking into account that the safe-range and domain independent fragments are equally expressive).

Bárány, Benedikt, and ten Cate (2013) show that Craig Interpolation and Projective

Beth Definability fail for the Guarded and the Packed fragments, contradicting earlier results in Marx (2007). By adapting the ideas of Marx (2007), they also proved that Craig Interpolation and Projective Beth Definability hold in GNFO. These results, which are useful and interesting by themselves, are used later in the paper by the authors in order to obtain results for determinacy and rewriting in the GNFO fragment. This part of the paper is the most relevant for our work, and we paid the most attention to it (especially to Theorem 9). Here for determinacy and rewriting, the authors assumed a setting similar to the one in Nash et al. (2010). In order to understand the difference between our setting and the setting of Bárány et al. (2013), let us summarise their setting using our notation. They assume having a finite set $\mathbb{V}$ of relation names (view predicates) $\mathbb{V} = \{V_i, \ldots, V_n\}$ and a set of view definitions (a restricted form of ontology) $\mathcal{T} = \{\forall \bar{x}_1. V_1(\bar{x}_1) \leftrightarrow \phi_1(\bar{x}_1), \ldots, \forall \bar{x}_n. V_n(\bar{x}_n) \leftrightarrow \phi_n(\bar{x}_n)\}$, where each ϕ_i is a first-order formula over a signature σ that is disjoint from $\mathbb{V}$. A query $\mathcal{Q}$ is a first-order formula over σ. They say that the views $\mathbb{V}$ determine the query $\mathcal{Q}$ if for any two models of $\mathcal{T}$, $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$ and $\mathcal{J} = \langle \Delta^{\mathcal{J}}, \cdot^{\mathcal{J}} \rangle$, such that $V_i^{\mathcal{I}} = V_i^{\mathcal{J}}$ for all i, $\mathcal{Q}(\mathcal{I}) = \mathcal{Q}(\mathcal{J})$. In our case the definition of determinacy simply coincides with the definition of implicit definability. The main difference between these definitions is that Bárány et al. (2013) do not assume $\Delta^{\mathcal{I}} = \Delta^{\mathcal{J}}$. In terms of Bárány et al. (2013), a rewriting of $\mathcal{Q}$ over $\mathbb{V}$ is a formula ρ expressed over the signature $\mathbb{V}$ if for any model $\mathcal{I}$ of $\mathcal{T}$, $\mathcal{I} = \langle \Delta^{\mathcal{I}}, \cdot^{\mathcal{I}} \rangle$, the following holds: $\mathcal{Q}(\mathcal{I}) = \rho(\mathcal{I}_{\mathbb{V}})$, where $\mathcal{I}_{\mathbb{V}}$ is an interpretation whose domain is $\bigcup_{i=1}^{n} V_i^{\mathcal{I}}$ and $V_i^{\mathcal{I}_{\mathbb{V}}} = V_i^{\mathcal{I}}$ for all i. We consider a rewriting to be simply a formula that is logically equivalent to the original query with respect to the ontology. We call such rewritings exact reformulations. The main difference here is that rewritings in terms of Bárány et al. (2013) require the answer to the query in any model of $\mathcal{T}$ to be equal not to the answer to the rewriting in the model itself (as it is for exact reformulations in our setting), but to the answer to the rewriting in a "relativisation" of the model over the union of extensions of all view predicates. If there is a rewriting, then there is a domain independent rewriting (one can obtain it by some kind of relativisation). We can observe that if a rewriting is domain independent, then these two definitions coincide.

Example 5.45. Consider a very simple example. Let $\mathcal{T} = \{\forall x. V(x) \leftrightarrow A(x)\}$ be view definitions and $\mathcal{Q} = \forall x. A(x)$ be a query. Note that the query is not domain independent. Let us check if the query is determined in the sense of Bárány et al. (2013) by the set of views. Let $\mathcal{I} = \langle \{a, b\}, \cdot^{\mathcal{I}} \rangle$ and $\mathcal{J} = \langle \{a\}, \cdot^{\mathcal{J}} \rangle$ be two interpretations such that $A^{\mathcal{I}} = A^{\mathcal{J}} = V^{\mathcal{I}} = V^{\mathcal{J}} = \{a\}$. Then $\mathcal{I}$ and $\mathcal{J}$ are models of $\mathcal{T}$. But $\mathcal{Q}(\mathcal{I}) = \forall x. A(x)(\mathcal{I}) = \mathbf{false}$ while $\mathcal{Q}(\mathcal{J}) = \forall x. A(x)(\mathcal{J}) =$

$\mathbf{true}$. So, the views do not determine the query and there is not a rewriting of the query over the views; on the other hand, the query is determined in our sense, and there exists a formula expressed over view predicates that is logically equivalent to $\mathcal{Q}$ with respect to $\mathcal{T}$ — an exact reformulation of $\mathcal{Q}$: $\forall x.\, V(x)$.

Example 5.46. Consider another simple example. Let $\mathcal{T} = \{\forall x.\, V(x) \leftrightarrow \neg A(x)\}$ be view definitions and $\mathcal{Q}(x) = A(x)$ be a query. Note that the only view, expressed by the formula $\neg A(x)$, is not domain independent. Let us check whether the query is determined in the sense of Bárány et al. (2013) by the set of views. Let $\mathcal{I} = \langle \{a, b, c\}, \cdot^{\mathcal{I}} \rangle$ and $\mathcal{J} = \langle \{a, b\}, \cdot^{\mathcal{J}} \rangle$ be two interpretations such that $A^{\mathcal{I}} = \{b, c\}$, $A^{\mathcal{J}} = \{b\}$, $V^{\mathcal{I}} = V^{\mathcal{J}} = \{a\}$. Then $\mathcal{I}$ and $\mathcal{J}$ are models of $\mathcal{T}$. But $\mathcal{Q}(x)(\mathcal{I}) = A(x)(\mathcal{I}) = \{b, c\}$ while $\mathcal{Q}(x)(\mathcal{J}) = A(x)(\mathcal{J}) = \{b\}$. So, the views do not determine the query and there is not a rewriting of the query over the views; on the other hand, the query is determined in our sense and there exists a formula expressed over view predicates that is logically equivalent to $\mathcal{Q}$ with respect to $\mathcal{T}$ — an exact reformulation of $\mathcal{Q}$: $\neg V(x)$.

The examples above show that the definition of determinacy given in Bárány et al. (2013) for GNFO and our definition of determinacy coincide if we restrict ourself to a domain independent fragment.

Bárány et al. (2013) also proved that under the additional assumption that the free variables in all the view definitions ϕ_i and in the query are guarded by some atom — namely whenever the views and the query are *answer-guarded* — if the views determine the query then it is possible to find a rewriting of the query over a set of views expressed in GNFO. Note also that, in addition to view definitions, the ontology may include arbitrary GNFO sentences. Determinacy and rewritings then are considered with respect to the whole ontology: the models considered are models of both the view definitions and of the sentences in the ontology. This makes their framework closer to ours. Using the finite model property of GNFO the authors show that the result in GNFO holds also if the query is determined by the views over finite models only. In this case, determinacy over finite and unrestricted models coincide, and this is similar to our notion of finite controllability of determinacy.

The paper by Benedikt, ten Cate, and Tsamoura (2014) is the most recent and relevant paper for us, appearing after we published our main results in Franconi et al. (2013). Benedikt et al. (2014) work on generating plans to answer queries completely in the presence of arbitrary first-order integrity constraints. Their technique is based on interpolation – they proposed a new interpolation theorem for their setting. Relations in a schema may have limited access (called access restrictions or binding patterns). The authors are also interested in finding the lowest cost plans, but this is not relevant for us. Our setting can be considered as

a special case of their setting. Indeed, our DBox predicates are relations in the schema that are fully accessible, while all the other predicates are not accessible at all. The important difference is again, that similarly to Nash et al. (2010), they assume the active domain semantics for relational structures (interpretations). They extend the work by Nash et al. (2010) by considering arbitrary first-order constraints. Benedikt et al. (2014) proved theorems that guarantee the existence of a plan (reformulation for us) for a query, if the query has a certain preservation property (a kind of determinacy). In particular, the most important and relevant results for us are Theorem 2 and Claim 1. In our setting the theorem together with the claim may be summarised in the following statement: a query has a reformulation over DBox predicates under constraints if and only if it is determined by DBox predicates under constraints. Taking into account that the authors use active domain semantics one can see that this result is equivalent to our constructive theorem.

The paper also gives conditions to obtain positive existential and existential rewritings respectively. Each of these conditions represent also a variant of determinacy.

The main theorem in Benedikt et al. (2014) (Theorem 2) in general does not work for the finite case. That is, it may be the case that there is a plan (reformulation) that works over finite structures only, but there is no plan that works over unrestricted structures. In order to regain completeness, Benedikt et al. (2014) proposed considering "finitely controllable" fragments, similar to our idea. As an example, one can consider in our setting the GNFO fragment that is finitely controllable for queries (in our definitions) from the answer-guarded subfragment of GNFO (see Subsection 5.2.1).

Toman and Weddell have long advocated the use of exact reformulations based on interpolation for automatic generation of plans that implement user queries, which they call query compilation (2011). Their published work focuses on additional extra-logical considerations, such as satisfaction of binding patterns, considerations of inherent ordering of data and the influence of plans on such orderings, and the estimated cost of alternative query plans.

In our analysis of the related literature concerning query rewritability, we noticed that the domain independence of a query and of the constraints is assumed in many cases "implicitly", i.e. by imposing the active domain semantics (Nash et al., 2010; Benedikt et al., 2014) or "encoding" it in the definitions of determinacy (Bárány et al., 2013). In our approach, we require it explicitly, revealing the influence and importance of this property and keeping a classical semantics for the language. This is important in order to be able to relate our results with more classical branches of knowledge representation – in this work we have considered description logics.

Regarding the complexity of reasoning with DBoxes in description logics, it has been shown that it is not the same as with ABoxes. Complexity of conjunctive query answering with DBoxes in description logics was studied in Franconi et al. (2011). Queries are evaluated over a DBox with respect to an ontology represented by a TBox. It was proved that conjunctive query answering in the quite expressive description logic $\mathcal{ALCFI}$ extended with DBoxes is polynomially reducible to the same problem in $\mathcal{ALCFIO}$ and vice versa. As a consequence of these reductions, conjunctive query answering for $\mathcal{ALCFI}$ with DBoxes (combined complexity) is strictly harder (coN2ExpTime-hard (Glimm, Kazakov, & Lutz, 2011)) than conjunctive query answering with ABoxes (2-ExpTime-complete (Glimm, Lutz, Horrocks, & Sattler, 2008)). Regarding data complexity, the lightweight description logic DL-Lite$_{\mathcal{F}}$ (Calvanese, Giacomo, Lembo, Lenzerini, & Rosati, 2007) with DBoxes that is closely related to $\mathcal{ALCFIO}$ has been considered. Data complexity of conjunctive query answering in DL-Lite$_{\mathcal{F}}$ with ABoxes is tractable (AC^0) since it can be reduced to query answering in relational databases. But this problem becomes coNP-complete with DBoxes. The exact combined complexity of conjunctive query answering in DL-Lite$_{\mathcal{F}}$ with DBoxes remains open. Query answering with DBoxes for another inexpressive logic $\mathcal{EL}$ (Baader, Brandt, & Lutz, 2005) which enjoys polynomial time data reasoning (with ABoxes) (Krisnadhi & Lutz, 2007) was also studied. The following results have been obtained: conjunctive query answering in $\mathcal{EL}$ with DBoxes is coNP-hard for data complexity (Lutz, Seylan, & Wolter, 2012) and ExpTime-hard for combined complexity.

We can conclude that query answering with DBoxes in general is really hard.

Both definability for $\mathcal{ALC}$ and its extensions with transitive roles, inverse roles, and/or functionality restrictions was studied in Seylan et al. (2009) and ten Cate et al. (2011). It can be considered as an instantiation of our general framework when the ontology is a description logic TBox and query is a concept query. In $\mathcal{ALC}$ and its extensions with constructors from $\{\mathcal{S}, \mathcal{I}, \mathcal{F}\}$, checking implicit definability is ExpTime-complete. The algorithm that computes explicit definitions (concepts) runs in 2-ExpTime and computes in the worst case an explicit definition of double exponential size, which is also a lower bound. The case when explicit definitions are allowed to be expressed in first-order logic was also considered. In this case, the algorithm computes a first-order explicit definition of an implicitly defined concept in single ExpTime and its size is proved to be at most single exponential.

6. Conclusions and Future Work

In the first part of the thesis we proposed a method that allows to reduce the problem of checking domain independence of a formula to checking standard first-order logic entailment: a formula $\phi(\bar{x})$ from $\mathcal{FOL}(\mathbb{C}, \mathbb{P})$ is domain independent whenever the entailment (3.2) holds. We considered applications of the method in a two-variable fragment of first-order logic and in prefix-vocabulary fragments $[\exists^*\forall^*, all, (0)]^{\mathbb{C}}_{=}$ and $[all, (w), (0)]^{\mathbb{C}, free}_{=}$.
We studied domain independent fragments of expressive description logics $\mathcal{SHOIQ}$, $\mathcal{SHOQ}$ and $\mathcal{ALCHOI}$ and proved variants of Codd's theorem for them. We also give recursive definitions of domain independent concepts of $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$ and thus provide convenient syntactic characterizations of domain independent fragments of these logics.
In the second part of the thesis we have introduced a framework to compute the exact reformulation of first-order queries to a database (DBox) under constraints. 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 DBox to give the same answer as the original query under the constraints. Non-trivial case studies have been presented in the field of description logics, with the $\mathcal{ALCHOI}$ and $\mathcal{SHOQ}$ languages.
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 reformulations preserving the certain answers. In our scenario with description logics ontologies, reformulations 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.
Next, 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 reformulations, the numbers of used predicates, the priority of predicates, the number of relational operators, and clever usage of duplicates. In this context one may also want to control the process of formula proving to make it produce an optimal reformulation. For instance, using the tableau method, one may prefer one order of expansion rules application to another and, hence, build another interpolant.
Concurrently, our research group is exploring the problem of fixing real ontologies in order to enforce definability when it is known it should be the case (Franconi,

Ngo, & Sherkhonov, 2012). This happens when it is intuitively obvious that the answer to 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. Our group introduced the novel problem of definability abduction and solved it completely in the data exchange scenario.

There is also another interesting open problem about checking that a given DBox is legal with respect to a given ontology. Remember that a DBox $\mathcal{DB}$ 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 DBox 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.

In the case of description logics, we would like to work on extending the theoretical framework with conjunctive queries: we need finitely controllable determinacy with conjunctive queries, which for some description logics seems to follow from the works by Bárány, Gottlob, and Otto (2010) and Rosati (2011).

References

Abiteboul, S., Hull, R., & Vianu, V. (1995). *Foundations of databases*. Boston, MA: Addison-Wesley.

Artale, A., Calvanese, D., Kontchakov, R., & Zakharyaschev, M. (2009*)*. The DL-Lite family and relations. *Journal of Artificial Intelligence Research, 36*, 1–69.

Avron, A. (2008, April). Constructibility and decidability versus domain independence and absoluteness. *Theoretical Computer Science, 394*(3), 144–158. https://doi.org/10.1016/j.tcs.2007.12.008

Baader, F., Brandt, S., & Lutz, C. (2005). Pushing the $\mathcal{EL}$ envelope. In L. P. Kaelbling & A. Saffiotti (Eds.), *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence, IJCAI05,* (pp. 364–369). Edinburgh, UK: Morgan Kaufman.

Bárány, V., Benedikt, M., & ten Cate, B. (2013). Rewriting guarded negation queries. In K. Chatterjee & J. Sgall (Eds.), *Lecture Notes in Computer Science: Vol. 8087. Mathematical Foundations of Computer Science, MFCS 2013 (*pp. 98–110). Retrieved from https://link.springer.com/content/pdf/10.1007%2F978-3-642-40313-2_11.pdf

Bárány, V., Gottlob, G., & Otto, M. (2010). Querying the guarded fragment. In F. V. Fomin, R. Freivalds, M. Kwiatkowska, & D. Peleg (Eds.), Symposium on *Logics in Computer Science, LICS2010,* (pp. 1–10). Edinburgh, UK: IEEE Computer Society Press.

Bárány, V., ten Cate, B., & Otto, M. (2012). Queries with guarded negation. *Proceedings of the VLDB Endowment, 5*(11), 1328–1339.

Bárány, V., ten Cate, B., & Segoufin, L. (2011). Guarded negation. In L. Aceto, M. Henzinger, & J. Sgall (Eds.), *Lecture Notes in Computer Science, 6756. International Colloquium on Automata, Languages, and Programming, ICALP 2011, Automata, Languages and Programming* (pp. 356–367). https://doi.org/10.1007/978-3-642-22012-8_28

Benedikt, M., ten Cate, B., & Tsamoura, E. (2014). Generating low-cost plans from proofs. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems PODS14* (pp. 200–211). http://doi.acm.org/10.1145/2594538.2594550

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

Börger, E., Grädel, E., & Gurevich, Y. (1997). *The classical decision problem.* Berlin: Springer.

Calvanese, D., Giacomo, G. D., Lembo, D., Lenzerini, M., & Rosati, R. (2007). Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *Journal of Automated Reasoning, 39*(3), 385–429.

Codd, E. F. (1972). Relational completeness of data base sublanguages. In R. Rustin (Ed.), *Data base systems* (pp. 65–98). Englewood Cliffs, NJ: Prentice-Hall.

Craig, W. (1957). Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory. *The Journal of Symbolic Logic, 22*(3), 269–285.

Demolombe, R. (1992, January). Syntactical characterization of a subset of domain-independent formulas. *Journal of the ACM, 39*(1), 71–94. doi:10.1145/147508.147520

Di Paola, R. A. (1969, April). The recursive unsolvability of the decision problem for the class of definite formulas. *Journal of the ACM, 16*(2), 324–327. Retrieved from http://dl.acm.org/citation.cfm?id=321524

Etzioni, O., Golden, K., & Weld, D. (1997). Sound and efficient closed-world reasoning for planning. *Artificial Intelligence, 89*, 113–148. https://doi.org/10.1016/S0004-3702(96)00026-4

Fan, W., Geerts, F., & Zheng, L. (2012). View determinacy for preserving selected information in data transformations. *Information Systems, 37*, 1–12. http://dx.doi.org/10.1016/j.is.2011.09.001

Feinerer, I., Franconi, E., & Guagliardo, P. (2015). Lossless selection views under conditional domain constraints. *IEEE Transactions on Knowledge and Data Engineering, 27*(2), 504–517. doi:10.1109/TKDE.2014.2334327

Feinerer, I., Guagliardo, P., & Franconi, E. (2014). Lossless selection views under constraints. In G. Gottlob & J. Pérez (Eds.), *Proceedings of the 8th Alberto Mendelzon International Workshop on Foundations of Data Management, AMW 2014. CEUR Workshop Proceedings, 1189.* Retrieved from http://ceur-ws.org/Vol-1189/paper_1.pdf

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

Franconi, E., & Guagliardo, P. (2012). On the translatability of view updates. In J. Freire & D. Suciu (Eds.), *Proceedings of the 6th Alberto Mendelzon International Workshop on Foundations of Data Management, AMW 2012. CEUR Workshop Proceedings, 866.* Retrieved from http://ceur-ws.org/Vol-866/paper11.pdf

Franconi, E., & Guagliardo, P. (2013). Effectively updatable conjunctive views.In L. Bravo & M. Lenzerini (Eds.), *Proceedings of the 7th Alberto Mendelzon International Workshop on Foundations of* Data *Management,*

AMW 2013. CEUR Workshop Proceedings, 1087. Retrieved from http://ceur-ws.org/Vol-1087/paper7.pdf

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

Franconi, E., Kerhet, V., & Ngo, N. (2012). Exact query reformulation over $\mathcal{SHOQ}$ DBoxes. In Y. Kazakov, D. Lembo, & F. Wolter (Eds.), *Proceedings of the 2012 International Workshop on Description Logics, DL2012. CEUR Workshop Proceedings, 846.* Retrieved from http://ceur-ws.org/Vol-846/paper_64.pdf

Franconi, E., Kerhet, V., & Ngo, N. (2013). Exact query reformulation over databases with first-order and description logics ontologies. *Journal of Artificial Intelligence Research (JAIR), 48*, 885-922. doi:10.1613/jair.4058

Franconi, E., Ngo, N., & Sherkhonov, E. (2012). The definability abduction problem for data exchange. In M. Krötzsch & U. Straccia (Eds.), *Lecture Notes in Computer Science: Vol. 7497. Proceedings of the 6th international conference on Web Reasoning and Rule Systems, RR2012* (pp. 217–220). https://doi.org/10.1007/978-3-642-33203-6_18

Franconi, E., Kerhet, V., & Ngo, N. (2012). Exact query reformulation with first-order ontologies and databases. In L. Fariñas del Cerro, A. Herzig, & J. Mengin (Eds.), *Lecture Notes in Computer Science: Vol. 7519. Logics in artificial intelligence* (pp. 202–214). https://doi.org/10.1007/978-3-642-33353-8_16

Gelder, A. V., & Topor, W. (1991). Safety and translation of relational calculus queries. *ACM Transactions on Database Systems, TODS, 16*(2), 235–278. doi:10.1145/114325.103712

Glimm, B., Kazakov, Y., & Lutz, C. (2011). Status $\mathcal{QIO}$: An update. In R. Rosati, S. Rudolph, & M. Zakharyaschev (Eds.), *Proceedings of the 2011 International Workshop on Description Logics, DL2011. CEUR Workshop Proceedings, 745.* Retrieved from http://ceur-ws.org/Vol-745/paper_44.pdf

Glimm, B., Lutz, C., Horrocks, I., & Sattler, U. (2008). Conjunctive query answering for the description logic $\mathcal{SHIQ}$. *Journal of Artificial Intelligence Research, 31*, 157–204. http://dx.doi.org/10.1613/jair.2372

Grädel, E., Kolaitis, P. G., Libkin, L., Marx, M., Spencer, J., Vardi, M. Y., … Weinstein, S. (2005). *Finite model theory and its applications*. doi: 10.1007/3-540-68804-8

Grädel, E., Kolaitis, P. G., & Vardi, M. Y. (1997). On the decision problem for two-variable first-order logic. *Bulletin of Symbolic Logic, 3*(1), 53–69. doi:10.2307/421196

Gurevich, Y. (1984). Toward logic tailored for computational complexity. In E. Börger, W. Oberschelp, M.M. Richter, B. Schinzel, W. Thomas (Eds.) *Lecture Notes in Mathematics: Vol. 1104. Computation and Proof Theory* (pp. 175-216). https://doi.org/10.1007/BFb0099486

Gyssens, M. (2009). Database dependencies. In L. Liu & M. Özsu (Eds.), *Encyclopedia of database systems* (pp. 704–708). doi:10.1007/978-0-387-39940-9

Halevy, A. Y. (2001). Answering queries using views: A survey. *The VLDB Journal – The International Journal on Very Large Data Bases, 10*(4), 270–294. http://dx.doi.org/10.1007/s007780100054

Horrocks, I., & Sattler, U. (2001). Ontology reasoning in the $\mathcal{SHOQ}$(D) description logic. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence, IJCAI01,* (Vol. 1, pp. 199–204). San Francisco,CA: Morgan Kaufmann.

Kerhet, V., & Franconi, E. (2012). On checking domain independence. In F. A. Lisi (Eds.): *Proceedings of the 9th Italian Convention on Computational Logic, CILC2012. CEUR Workshop Proceedings, 857.* Retrieved from http://ceur-ws.org/Vol-857/paper_s03.pdf

Kleene, S. C. (2002). *Mathematical logic*. New York: Dover.

Krisnadhi, A., & Lutz, C. (2007). Data complexity in the $\mathcal{EL}$ family of description logics. In N. Dershowitz & A. Voronkov (Eds.), *Lecture Notes in Artificial Intelligence: Vol. 4790 Proceedings of the 14th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning, LPAR2007* (pp. 333–347). Retrieved from https://iccl.inf.tu-dresden.de/w/images/b/b7/Krisnadhi-Lutz-LPAR07.pdf

Lutz, C., Areces, C., Horrocks, I., & Sattler, U. (2005). Keys, nominals,and concrete domains. *Journal of Artificial Intelligence Research, 23*, 667–726. Retrieved from https://www.jair.org/media/1542/live-1542-2375-jair.pdf

Lutz, C., Seylan, I., & Wolter, F. (2012). Mixing open and closed world assumption in ontology-based data access: Non-uniform data complexity. In Y. Kazakov, D. Lembo, & F. Wolter (Eds.), *Proceedings of the 2012 International Workshop on Description Logics, DL2012. CEUR Workshop Proceedings, 846.* Retrieved from http://ceur-ws.org/Vol-846/paper_17.pdf

Marx, M. (2007). Queries determined by views: pack your views. In *PODS'07 Proceedings of the 26rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems PODS07* (pp. 23–30). http://doi.acm.org/10.1145/1265530.1265534

Mortimer, M. (1975). On language with two variables. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik, 21*, 135–140.

Nash, A., Segoufin, L., & Vianu, V. (2010). Views and queries: Determinacy and rewriting. *ACM Transactions on Database Systems (TODS), 35*(3), 21:1–21:41. http://doi.acm.org/10.1145/1806907.1806913

Nicolas, J.-M. (1982). Logic for improving integrity checking in relational data bases. *Acta Informatica, 18*(3), 227–253. https://doi.org/10.1007/BF00263192

Otto, M. (2000). An interpolation theorem. *Bulletin of Symbolic Logic, 6*(4), 447–462. doi:10.2307/420966

Rosati, R. (2011). On the finite controllability of conjunctive query answering in databases under open-world assumption. *Journal of Computer and System Sciences, 77*(3), 572–594. https://doi.org/10.1016/j.jcss.2010.04.011

Scott, D. (1962). A decision method for validity of sentences in two variables. *Journal of Symbolic Logic, 27*, 377.

Seylan, I., Franconi, E., & de Bruijn, J. (2009). Effective query rewriting with ontologies over DBoxes. In C. Boutilier (Ed.), *Proceedings of the 21st International Joint Conference on Artifical intelligence, IJCAI09* (pp. 923-925). San Francisco, CA: Morgan Kaufmann.

ten Cate, B., Franconi, E., & Seylan, I. (2013). Beth definability in expressive description logics. *Journal of Artificial Intelligence Research, 48*, 347–414. http://dx.doi.org/10.1613/jair.4057

ten Cate, B., Franconi, E., & Seylan, I.. (2011). Beth definability in expressive description logics. In *Proceedings of the 22nd International Joint Conference on Artifical intelligence, IJCAI11* (p. 1099-1106). doi:10.5591/978-1-57735-516-8/IJCAI11-188

Toman, D., & Weddell, G. (2011). *Fundamentals of physical design and query compilation*. https://doi.org/10.2200/S00363ED1V01Y201105DTM018

Topor, R.W. (1987). Domain-independent formulas and databases. *Theoretical Computer Science, 52*(3), 281–306. https://doi.org/10.1016/0304-3975(87)90113-7

Van Heigenoort, J. (1977). *From Frege to Gödel: A source book in mathematical logic, 1879–1931*. Cambridge, MA: Harvard University Press.

The Author

Volha Kerhet received her MSc in Mathematics from the Belarusian State University in 2007. As a junior researcher she has been involved into the YourWay! project at the Fondazione Bruno Kessler in Trento, Italy and the ONTORULE project at the Free University of Bozen-Bolzano, Italy. In 2015 she received her PhD in Computer Science from the Free University of Bozen-Bolzano. Volha Kerhet is currently doing research at the KRDB Research Centre at the Faculty of Computer Science of the Free University of Bozen-Bolzano focusing on ontology-based data access (OBDA) and bag semantics.