TECHNISCHE UNIVERSITÄT DRESDEN

Technische Universität Dresden
Institute for Theoretical Computer Science
Chair for Automata Theory

# LTCS–Report

## Query Rewriting for *DL-Lite* with $n$-ary Concrete Domains (Extended Version)

Franz Baader, Stefan Borgwardt, and Marcel Lippmann

LTCS-Report 17-04

*This is an extended version of a paper published in the proceedings of IJCAI 2017.*

Postal Address:
Lehrstuhl für Automatentheorie
Institut für Theoretische Informatik
TU Dresden
01062 Dresden

http://lat.inf.tu-dresden.de

Visiting Address:
Nöthnitzer Str. 46
Dresden

# Contents

**Abstract**

We investigate ontology-based query answering (OBQA) in a setting where both the ontology and the query can refer to concrete values such as numbers and strings. In contrast to previous work on this topic, the built-in predicates used to compare values are not restricted to being unary. We introduce restrictions on these predicates and on the ontology language that allow us to reduce OBQA to query answering in databases using the so-called combined rewriting approach. Though at first sight our restrictions are different from the ones used in previous work, we show that our results strictly subsume some of the existing first-order rewritability results for unary predicates.

# 1 Introduction

*Ontology-based query answering (OBQA)* (see, e.g., [32] for an overview) extends query answering in databases in two directions. On the one hand, in OBQA it is not assumed that the available data are complete, and thus facts that are not present are assumed to be unknown rather than false (*no* closed world assumption [CWA]). On the other hand, an ontology can be used to state background knowledge about the data and to translate between different vocabularies (e.g., user oriented versus system oriented). Nevertheless, if the query language and the ontology language are suitably restricted, then OBQA can be reduced to classical query answering in databases.

Regarding the query language, one usually considers (unions of) *conjunctive queries ((U)CQs)* (i.e., select-project-join queries) in this setting. If the ontology language belongs to the so-called *DL-Lite family* of Description Logics (DLs) [3, 10], then the ontology can often be compiled into the query, which can then be evaluated over the unchanged data using the CWA [10, 13]. If this approach is feasible, then one says that the query language is *first-order (FO) rewritable* w.r.t. the ontology language. FO rewritability implies that OBQA then has the same data complexity as query answering in databases, $AC^0$. For settings where the data complexity of OBQA is no longer in $AC^0$ (e.g., if the DL $\mathcal{EL}$ is used as ontology language), the *combined rewriting* approach, in which both the query and the data are changed, has turned out to be useful [24, 29]. In case the data can be rewritten in polynomial time, this yields polynomial data complexity.

Real-world datasets, however, frequently contain concrete data values (such as numbers and strings), and database queries use built-in predicates on these values to formulate restrictions on the tuples to be selected. When extending the use of concrete data values and built-in predicates to the OBQA setting, it makes sense to employ them not only in the query, but also in the ontology. In ontology languages based on DLs, one then talks about DLs with *concrete domains* [5, 27]. In addition to *concepts* and *roles* (i.e., unary and binary predicates on the abstract domain), such DLs employ *attributes* (i.e., binary relations between the abstract and the concrete domain) to assign concrete values to individuals, and *concrete predicates* (corresponding to built-in predicates in databases) to formulate constraints on these values.

Motivated by OBQA applications, several authors have introduced dialects of *DL-Lite* and CQs with concrete domains [4, 35, 38]. However, like the standard Web Ontology Language OWL 2,[1] these extensions of *DL-Lite* with concrete domains consider only *unary* predicates on data values, which can be used to constrain a single value, but cannot require relationships between different values. With unary predicates one can, for example, express that the systolic blood pressure of a patient is >120 and the diastolic blood pressure is >80, but setting the systolic blood pressure into a relationship with the diastolic one requires a binary predicate.

In this work, we lift this restriction, i.e., we define an extension of *DL-Lite* with concrete domains that may have predicates of arbitrary arity, and show that—for concrete domains satisfying certain properties—CQs with built-in predicates from the concrete domain allow for a combined rewriting w.r.t. ontologies formulated in this new language. For example, using an appropriate

[1] see https://www.w3.org/TR/owl2-overview/

binary predicate we can then express that the pulse pressure, i.e., the difference between the systolic and the diastolic blood pressure, is 50.

Note that, in general, we do not assume that attributes are functional, but our logic can express (local) functionality (e.g., ensuring that a patient can have only one systolic blood pressure). We also show that concrete domains satisfying our restrictions are closed under disjoint union and product. Using the product of two numerical domains (one for pressure values and one for time), we can compare measurements at different time points; e.g., we can ask for patients whose systolic blood pressure increased by 20 in 30 seconds.

In addition to introducing our combined rewriting approach and proving that it is correct, we also show that the FO rewritability results for the *DL-Lite* variant with concrete domains with unary predicates in [38] follow from our results. Basically, we show that (i) concrete domains with unary predicates satisfying the restrictions in [38] can be turned into ones satisfying our restrictions, and (ii) in the unary case our combined rewriting boils down to an FO rewriting. The results in [4] are orthogonal to ours since, on the one hand, they are restricted to the unary case, but on the other hand, they allow for more expressiveness on the DL side. In contrast to our work and [4, 38], in [35] queries do not contain built-in predicates. Finally, in [19] the authors also consider a setting with non-unary concrete domains, but where the data complexity is co-NP-hard in general. They then investigate for which kinds of queries this complexity goes down to P. In contrast, our goal is to find restrictions on non-unary concrete domains that ensure combined rewritability, and thus polynomial data complexity, for all queries.

## 2 Concrete Domains

Concrete domains are a well-known formalism for dealing with values in ontologies [26, 33, 38]. We first introduce the general notion of concrete domains, and then restrict it such that it fits our purpose.

**Definition 2.1** (Concrete domain)**.** A *concrete domain* $\mathcal{D}$ consists of

- a non-empty set $\Delta^{\mathcal{D}}$ of *values*,
- a collection of predicates $\Pi_i$ with associated arities $m_i$, containing the special unary predicate $\top_{\mathcal{D}}$, and
- interpretations $\Pi_i^{\mathcal{D}} \subseteq (\Delta^{\mathcal{D}})^{m_i}$ for all predicates, where $(\top_{\mathcal{D}})^{\mathcal{D}} = \Delta^{\mathcal{D}}$.

Concrete domains can be used to formulate constraints as follows. Let from now on $\mathsf{N_V}$ be a countable set of *variables*.

**Definition 2.2** (Syntax and semantics of $\mathcal{D}$-formulas)**.** A *$\mathcal{D}$-formula* $\phi$ is a Boolean combination of *$\mathcal{D}$-atoms* of the form $\Pi(v_1, \ldots, v_m)$, where $\Pi$ is an $m$-ary predicate and $v_1, \ldots, v_m \in \Delta^{\mathcal{D}} \cup \mathsf{N_V}$. The set of variables occurring in $\phi$ is denoted by $\mathsf{Var}(\phi)$. A *$\mathcal{D}$-conjunction* (*$\mathcal{D}$-disjunction*) is a conjunction (disjunction) of $\mathcal{D}$-atoms.

Given a finite set $V \subseteq \mathsf{N_V}$, a *variable assignment (for $V$)* is a mapping $f\colon V \to \Delta^{\mathcal{D}}$. For a $\mathcal{D}$-formula $\phi$ with $\mathsf{Var}(\phi) \subseteq V$, the set $\mathsf{sol}_V(\phi)$ contains all *solutions* for $\phi$, which are the variable assignments for $V$ under which $\phi$ is satisfied in $\mathcal{D}$ (using the standard notion of satisfaction in a relational structure). We simply write $\mathsf{sol}(\phi)$ if $V = \mathsf{Var}(\phi)$. A $\mathcal{D}$-formula is *satisfiable* if it has at least one solution. A set of $\mathcal{D}$-formulas $\Gamma$ *implies* a $\mathcal{D}$-formula $\psi$ if $\bigcap_{\phi\in\Gamma} \mathsf{sol}_V(\phi) \subseteq \mathsf{sol}_V(\psi)$, where $V \coloneqq \bigcup_{\phi\in\Gamma} \mathsf{Var}(\phi) \cup \mathsf{Var}(\psi)$. If $\Gamma$ is a singleton $\{\phi\}$, then we say that $\phi$ implies $\psi$.

In the DL literature, concrete domains are usually required to satisfy additional properties that are tailored to the reasoning problems under consideration. For example, in order to obtain

decidability of standard DL reasoning problems such as subsumption, Baader and Hanschke [1991] require the concrete domain to be *decidable*, which in our setting means that satisfiability of $\mathcal{D}$-conjunctions and implications between $\mathcal{D}$-conjunctions must be decidable. In the context of concrete domain extensions of $\mathcal{EL}$, this requirement is tightened by Baader et al. [2005] to decidability in *polynomial* time. However, to obtain polynomiality of subsumption, one additionally needs to require that the concrete domain is *convex*, i.e., whenever a $\mathcal{D}$-conjunction implies a (non-empty) $\mathcal{D}$-disjunction, then it should also imply one of its disjuncts. The papers [4, 38] among other things require $\mathcal{D}$ to be *unary*, which means that all its predicates must be unary.

Our combined rewritability results depend on the following properties.

**Definition 2.3** (cr-admissible)**.** The concrete domain $\mathcal{D}$ is *cr-admissible* if it is polynomial, convex, and satisfies the following additional properties:

- $\mathcal{D}$ *has equality*: it contains all unary predicates $=_d$ with $d \in \Delta^{\mathcal{D}}$, which are interpreted as $\{d\}$, as well as a binary predicate $=$, interpreted as $\{(d, d) \mid d \in \Delta^{\mathcal{D}}\}$.
- $\mathcal{D}$ is *functional*: for any $m$-ary predicate $\Pi$, $d \in \Delta^{\mathcal{D}}$, and $i$, $1 \leq i \leq m$, the formula $\Pi(v_1, \ldots, v_m) \wedge =_d(v_i)$ has at most one solution.
- $\mathcal{D}$ is *constructive*: for all $\mathcal{D}$-conjunctions $\phi$ and $\mathcal{D}$-disjunctions $\psi$ with $\mathsf{sol}_V(\phi) \setminus \mathsf{sol}_V(\psi) \neq \emptyset$, an element of this set can be computed in polynomial time.

**Example 2.4.** The following concrete domains are known to be *p-admissible* [6, 7], i.e., polynomial and convex:

- $\mathcal{D}_{\mathbb{Q}}$: The set $\mathbb{Q}$ of all rational numbers with the unary predicates $\top_{\mathcal{D}_{\mathbb{Q}}}$, $=_q$, and $>_q$ (with the interpretation $\{x \mid x > q\}$), and binary predicates $=$ and $+_q$ (with the interpretation $\{(x, y) \mid x = q + y\}$), for any $q \in \mathbb{Q}$.
- $\mathcal{D}_{\Sigma^*}$: The set $\Sigma^*$ of all words over a fixed alphabet $\Sigma$ with the unary predicates $\top_{\mathcal{D}_{\Sigma^*}}$ and $=_w$, and binary predicates $=$ and $\mathsf{conc}_w$ (with the interpretation $\{(x, y) \mid x = w \cdot y\}$), for any $w \in \Sigma^*$.

Interestingly, both $\mathcal{D}_{\mathbb{Q}}$ and $\mathcal{D}_{\Sigma^*}$ are also functional and constructive, and hence admissible. Although the properties p-admissibility, functionality, and constructivity are independent, the latter two are used extensively in the proofs of p-admissibility for $\mathcal{D}_{\mathbb{Q}}$ and $\mathcal{D}_{\Sigma^*}$ in [7].

## 2.1 Closure Properties

We consider only a single cr-admissible concrete domain $\mathcal{D}$ in this paper. However, one can combine several concrete domains $\mathcal{D}_1, \ldots, \mathcal{D}_n$ (e.g., the ones from Example 2.4) into a single one by constructing their disjoint union or their product, without affecting cr-admissibility.

### 2.1.1 Disjoint Union

The concrete domain $\mathcal{D}_1 \oplus \cdots \oplus \mathcal{D}_n$ has as its domain the disjoint union of $\Delta^{\mathcal{D}_1}$, …, $\Delta^{\mathcal{D}_n}$, and $\Omega$, where $\Omega$ is a set of countably infinitely many *untyped values* [38]. Its predicates are the individual predicates of $\mathcal{D}_1, \ldots, \mathcal{D}_n$, the unary predicates $\top_{\mathcal{D}_1 \oplus \cdots \oplus \mathcal{D}_n}$ and $=_\omega$, $\omega \in \Omega$, and the extended binary predicate $=$.

We have to add the set $\Omega$ as it was done in [38], since otherwise this construction would not preserve convexity: The atom $\top_{\mathcal{D}}(v)$ would imply $\top_{\mathcal{D}_1}(v) \vee \cdots \vee \top_{\mathcal{D}_n}(v)$, but not any of the disjuncts. The additional countably many predicates $=_\omega$ prevent this behavior. They

loosely correspond to *plain literals* in the RDF standard, which do not have an explicit type in OWL [14, 33]. The set $\Omega$ does not affect answers to queries since usually the type of all variables is known; i.e., we can restrict each variable $v$ using an atom of the form $\top_{\mathcal{D}_i}(v)$ to prevent solutions from using the set $\Omega$. In our ontology language, we can additionally restrict all attributes to one domain $\mathcal{D}_i$ using attribute range constraints (see Section 3), e.g., expressing that the age of a person is always a number and the name is always a string.

**Lemma 2.5.** *If $\mathcal{D}_1, \dots, \mathcal{D}_n$ are cr-admissible concrete domains, then $\mathcal{D}_1 \oplus \cdots \oplus \mathcal{D}_n$ is also cr-admissible.*

*Proof.* Let $\mathcal{D} \coloneqq \mathcal{D}_1 \oplus \cdots \oplus \mathcal{D}_n$. Functionality is preserved since it is a condition on single predicates, which is satisfied by the new predicates $\top_{\mathcal{D}}$, $=$, and $=_\omega$, $\omega \in \Omega$.

The satisfiability of $\mathcal{D}$-conjunctions $\phi$ can be decided by the following procedure. First, we can remove atoms involving predicate $\top_{\mathcal{D}}$ since they do not restrict the solutions of $\phi$. Now, we assign each variable $v$ occurring in $\phi$ to one of the component domains $\Delta^{\mathcal{D}_1}, \dots, \Delta^{\mathcal{D}_n}, \Omega$ as follows: If $v$ occurs in a predicate of one of these domains, then it is assigned to this domain. Moreover, if $v$ occurs in an atom $=(v, v')$ in $\phi$ and we already know that $v'$ is assigned to $\mathcal{D}_i$ or $\Omega$, then $v$ must also belong to that component. After exhaustively labeling variables in this way, the only remaining variables are those that occur only in equality atoms $=(v_1, v_2)$. These atoms are obviously satisfiable independently of the other atoms in $\phi$, and hence we can remove them. This means that each atom should now be uniquely associated to one of the component domains. If a variable occurs in predicates from several of the component domains, then $\phi$ is obviously unsatisfiable. Otherwise, $\phi$ can be split into independent components for each of the component domains, and satisfiability can be checked independently. A conjunction of atoms over $\Omega$ is satisfiable iff no two variables connected by a chain of equality predicates occur in two different $=_\omega$-atoms, which is decidable in polynomial time.

To show that $\mathcal{D}$ is constructive, consider a $\mathcal{D}$-disjunction $\psi$ that is not implied by a $\mathcal{D}$-conjunction $\phi$, and let $f \in \mathsf{sol}(\phi) \setminus \mathsf{sol}(\psi)$. We can partition $\phi$ according to the images of the variables under $f$ into independent conjunctions $\phi_i, \dots, \phi_n, \phi_\Omega$, which do not share variables and are solved by $f$ in one of the component domains $\Delta^{\mathcal{D}_1}, \dots, \Delta^{\mathcal{D}_n}, \Omega$, respectively. Since each component is constructive, we can compute partial solutions $f_i \in \mathsf{sol}(\phi_i) \setminus \mathsf{sol}(\psi)$, $i \in \{1, \dots, n, \Omega\}$ in polynomial time. In particular, for $\phi_\Omega$ we can easily find enough values in $\Omega$ that do not satisfy any atom of the form $=(v_i, v_j)$ or $=_{\omega_i}(v)$ in $\psi$. Since each $f_i$ maps the variables of $\phi_i$ into a disjoint component, we can combine them directly to obtain a solution of $\phi$ that does not satisfy any disjunct of $\psi$.

To show decidability of implications of $\mathcal{D}$-atoms $\alpha$ by $\mathcal{D}$-conjunctions $\phi$, we can likewise assume that $\top_{\mathcal{D}}$ does not occur in the input. Furthermore, we again assign all variables and atoms in $\phi$ to a unique component domain, where we associate all equality atoms $=(v_1, v_2)$ whose component is not clear with $\Omega$. If $\phi$ is unsatisfiable (for example if a variable occurs in predicates from different domains), then the implication holds. Otherwise, if a variable in $\alpha$ occurs in $\phi$ in a predicate belonging to a different domain than the one in $\alpha$, then the implication does not hold. Finally, if these type checks have succeeded, then the implication only depends on those conjuncts in $\phi$ that match the domain of the variables in $\alpha$. An implication between atoms over $\Omega$ can be decided by computing the equivalence closure of all involved equality atoms and assigning elements of $\Omega$ to some of the resulting equivalence classes, in polynomial time.

To prove convexity, consider a valid implication between a $\mathcal{D}$-conjunction $\phi$ and a $\mathcal{D}$-disjunction $\psi$. If $\phi$ is unsatisfiable, then it also implies all the disjuncts of $\psi$ individually, and hence we are done. If $\psi$ contains an atom of the form $\top_{\mathcal{D}}(v)$, then $\phi$ also implies this atom, and thus we assume in the following that $\top_{\mathcal{D}}$ does not occur in $\psi$. Further, if $\top_{\mathcal{D}}(v)$ occurs in $\phi$ and $v$ also occurs in at least one different atom in $\phi$, then we can remove the atom $\top_{\mathcal{D}}(v)$ from $\phi$ without affecting convexity. If $v$ occurs in $\phi$ only in an atom of the form $\top_{\mathcal{D}}(v)$, then the solutions of the finite disjunction $\psi$ must cover all possibilities of mapping $v$ to an untyped element $\omega \in \Omega$.

Since there are infinitely many such elements and they only occur in the interpretation of the predicates $=_\omega$, $\top_\mathcal{D}$, and $=$, it must be the case that $v$ occurs in $\psi$ only in equality atoms of the form $=(v, v')$. By fixing all other variables according to an arbitrary solution of $\phi$, we can show that $\top_\mathcal{D}(v)$ implies a disjunction of the form $=_{d_1}(v) \vee \cdots \vee =_{d_m}(v)$, which is impossible (cf. the proof of Lemma 2.7). Thus, we assume in the following that $\top_\mathcal{D}$ does not occur in $\phi$ or $\psi$.

Now, any disjunct $\Pi(v_1, \ldots, v_m)$ where one of the variables $v_1, \ldots, v_m$ occurs in $\phi$ in a predicate from a different component domain can be removed from $\psi$ without changing the validity of the implication. Afterwards, it is easy to see that the implication can again be split up according to the membership of the predicates to the same concrete domain (where all equality atoms in $\psi$ belong to all domains, and all equality atoms in $\phi$ that cannot be uniquely assigned to a domain are assigned to $\Omega$), and it must be the case that one of the resulting implications is valid in the corresponding component. If this component is one of $\mathcal{D}_1, \ldots, \mathcal{D}_n$, then it follows from their cr-admissibility and the fact that the conjunction on the left-hand side of a valid implication can be extended without affecting the validity of the implication that $\phi$ implies one of the disjuncts of $\psi$.

Otherwise, we know that a conjunction $\phi'$ using only the predicates $=_\omega$ and $=$ implies a disjunction $\psi'$ with the same property, where the former is a part of $\phi$ and the latter is a part of $\psi$. We can remove all equality atoms from $\phi'$ by identifying the variables occurring together in equality atoms (see the proof of Lemma 2.7). If afterwards a variable $v$ is not constrained by an atom $=_{\omega_i}(v)$, then this is equivalent to having an atom $\top_\mathcal{D}(v)$, which was shown to be impossible above. Hence, the remaining conjunction ensures that all solutions must map each variable $v$ occurring in $\psi'$ to a fixed element $\omega_v$ of $\Omega$. But then $\psi'$ must contain either an equality atom of the form $=(v, v)$, an atom of the form $=(v, v')$ with $\omega_v = \omega_{v'}$, or an atom of the form $=_{\omega_v}(v)$. In each case, this atom is obviously implied by $\phi'$, and hence by $\phi$. □

### 2.1.2 Product

The concrete domain $\mathcal{D}_1 \otimes \cdots \otimes \mathcal{D}_n$ has as its domain $\Delta^{\mathcal{D}_1} \times \cdots \times \Delta^{\mathcal{D}_n}$, and for each $n$-tuple of $m$-ary predicates $(\Pi_1, \ldots, \Pi_n)$, where each $\Pi_i$ belongs to $\Delta_i$, it contains the $m$-ary predicate $\Pi_1 \otimes \cdots \otimes \Pi_n$, whose interpretation is

$$\big\{\big((d_1^1, \ldots, d_n^1), \ldots, (d_1^m, \ldots, d_n^m)\big) \mid \forall i \in \{1 \ldots, n\} \colon (d_i^1, \ldots, d_i^m) \in \Pi_i^{\mathcal{D}_i}\big\}.$$

We can define $\top_{\mathcal{D}_1 \otimes \cdots \otimes \mathcal{D}_n} \coloneqq \top_{\mathcal{D}_1} \otimes \cdots \otimes \top_{\mathcal{D}_n}$ and $=_{(d_1, \ldots, d_n)} \coloneqq =_{d_1} \otimes \cdots \otimes =_{d_n}$ to obtain the predicates required by Definition 2.1 and cr-admissibility.

For example, the product $\mathcal{D}_\mathbb{Q} \times \mathcal{D}_\mathbb{Q}$ can be used to model measurements that are associated with time stamps, to express statements like an increase of blood pressure by 20 in 30 seconds.

**Lemma 2.6.** *If $\mathcal{D}_1, \ldots, \mathcal{D}_n$ are cr-admissible concrete domains, then $\mathcal{D}_1 \otimes \cdots \otimes \mathcal{D}_n$ is also cr-admissible.*

*Proof.* We show the claim for a binary product, from which it easily follows for products with more components. Let hence $\mathcal{D} \coloneqq \mathcal{D}_1 \otimes \mathcal{D}_2$. Since both $\mathcal{D}_1$ and $\mathcal{D}_2$ have equality, the binary equality predicate on $\mathcal{D}$ is obtained by the product of the individual equality predicates.

Consider now a satisfiable $\mathcal{D}$-conjunction $\Pi(v_1, \ldots, v_m) \wedge =_d(v_j)$, where $\Pi = \Pi_1 \otimes \Pi_2$ and $d = (d_1, d_2)$. This conjunction is equivalent to the two conjunctions $\Pi_i(v_1^i, \ldots, v_m^i) \wedge =_{d_i}(v_j^i)$, $i \in \{1, 2\}$. These conjunctions are still satisfiable in their respective domains, and hence we obtain unique solutions for the variables $v_\ell^i$. These can be composed into a solution for the original conjunction, which must be unique since any other solution would differ in at least one component, and hence yield an alternative solution for one of the component conjunctions. This shows functionality of $\mathcal{D}$. The same arguments allow us to decompose any $\mathcal{D}$-conjunction into two conjunctions over $\mathcal{D}_1$ and $\mathcal{D}_2$ that are both satisfiable iff the original conjunction was

satisfiable. Similarly, an implication of a $\mathcal{D}$-atom by a $\mathcal{D}$-conjunction is valid iff it is valid in all components. Hence, the assumption that $\mathcal{D}_1$ and $\mathcal{D}_2$ are polynomial yields that that $\mathcal{D}$ is also polynomial.

To show that $\mathcal{D}$ is constructive, consider a $\mathcal{D}$-disjunction $\psi$ that is not implied by a $\mathcal{D}$-conjunction $\phi$. By the above arguments, one component of $\phi$, say $\phi_1$, does not imply the corresponding component $\psi_1$ of $\psi$ in $\mathcal{D}_1$, and we know that the other component $\phi_2$ has at least one solution in $\mathcal{D}_2$. Since both $\mathcal{D}_1$ and $\mathcal{D}_2$ are constructive, we can compute solutions of $\mathsf{sol}(\phi_1) \setminus \mathsf{sol}(\psi_1)$ and $\mathsf{sol}(\phi_2) \setminus \mathsf{sol}(\bot)$ in polynomial time, where $\bot$ denotes the empty disjunction. The combination of these solutions naturally forms an element of $\mathsf{sol}(\phi) \setminus \mathsf{sol}(\psi)$.

For convexity, consider a satisfiable $\mathcal{D}$-conjunction $\phi$ that implies a $\mathcal{D}$-disjunction $\psi$, and denote by $\phi_i$ and $\psi_i$ their restrictions to the $i$-th component, $i \in \{1, 2\}$. We assume that $\phi$ and $\psi$ contain the variables $v_1, \ldots, v_m$, and these correspond to the variables $v_1^i, \ldots, v_m^i$ in $\phi_i$ and $\psi_i$. By assumption, each $\phi_i$ is satisfiable and implies $\psi_i$. We show that $\phi$ implies a disjunct of $\psi$ by induction on the number $k$ of disjuncts of $\psi$. If $k = 1$, then clearly the claim holds. If $k > 1$, we choose an arbitrary disjunct $\alpha$ of $\psi$ and assume that $\phi$ does not imply $\alpha$. We will show that then $\phi$ must imply the remaining disjunction $\psi^\alpha$. Since $\psi^\alpha$ has $k - 1$ disjuncts, the induction hypothesis then yields that $\phi$ implies a disjunct of $\psi^\alpha$, which is also a disjunct of $\psi$.

Since $\phi$ does not imply $\alpha$, there is a solution $f \in \mathsf{sol}(\phi)$ that does not satisfy $\alpha$. By the semantics of the product predicates, $f$ can be split into solutions of $\phi_i$, $i \in \{1, 2\}$, by setting $f_i(v_j^i) \coloneqq d_j^i$, where $f(v_j) = (d_j^1, d_j^2)$. However, there must be one component $i \in \{1, 2\}$ such that $f_i$ does not satisfy $\alpha_i$; we assume without loss of generality that $i = 1$. Hence, $f_1$ is a counterexample for the implication of $\alpha_1$ by $\phi_1$. By the convexity of $\mathcal{D}_1$, we know that $\phi_1$ implies $\psi_1^\alpha$. Hence, it remains to show that also $\phi_2$ implies $\psi_2^\alpha$, since then the product semantics yields that $\phi$ implies $\psi^\alpha$. Assume to the contrary that $\phi_2$ does not imply $\psi_2^\alpha$, i.e., there is a solution $f_2'$ of $\phi_2$ that does not satisfy $\psi_2^\alpha$. We obtain a new variable assignment $f'$ by combining $f_1$ and $f_2'$ as follows: $f'(v_j) \coloneqq (f_1(v_j^1), f_2'(v_j^2))$. This assignment satisfies $\phi_1$ in the first component and $\phi_2$ in the second component, and thus is a solution of $\phi$. However, $f'$ does not satisfy $\alpha$ in the first component, and it does not satisfy $\psi^\alpha$ in the second component. Hence, it is a counterexample to the implication between $\phi$ and $\psi = \alpha \vee \psi^\alpha$, which contradicts our assumption. □

## 2.2 Unary Concrete Domains

The paper [38] about query answering in *DL-Lite* with unary concrete domains $\mathcal{D}$ imposes the following restriction:

***(infinitediff)*** For any $\mathcal{D}$-conjunction $\phi$ and $\mathcal{D}$-disjunction $\psi$, whenever $|\mathsf{sol}_V(\phi)| > 1$ and $\mathsf{sol}_V(\phi) \not\subseteq \mathsf{sol}_V(\psi')$ for every $\mathcal{D}$-atom $\psi'$ in $\psi$ (where $V \coloneqq \mathsf{Var}(\phi) \cup \mathsf{Var}(\psi)$), then the cardinality of $\mathsf{sol}_V(\phi) \setminus \mathsf{sol}_V(\psi)$ is infinite.

The original definition actually does not include the condition $|\mathsf{sol}_V(\phi)| > 1$. However, it is easily checked that the constructions and results of [38] remain valid under our weaker version of *(infinitediff)*. In our setting, this modification is useful to accommodate the predicates $=_d$, whose presence would otherwise contradict *(infinitediff)*. The authors of [37, 38] also consider two other restrictions, called *(infinite)* and *(opendomain)*, which, after closer inspection, turn out to be simply the special cases of *(infinitediff)* with $\psi = \mathsf{false}$ and $\phi = \mathsf{true}$, respectively.

In [4], a condition similar to *(infinitediff)* can be found, which was adapted from [37]. However, the paper [4] also considers only unary predicates and ignores the easy case of singleton solution sets: it requires that the difference between an arbitrary union and an arbitrary intersection of (interpretations of) predicates must be either empty or infinite. Additionally, a convexity condition is imposed on the concrete domain by requiring that the inclusion relationships between all predicates can be axiomatized by Horn rules.

The paper [38] considers only unary concrete domains that are decidable and satisfy *(infinitediff)*. To show that our results also apply to this setting, we first prove that we can add equality predicates to $\mathcal{D}$ without destroying *(infinitediff)*.

**Lemma 2.7.** *Let $\mathcal{D}$ be a* unary *concrete domain satisfying (infinitediff). Then the concrete domain $\mathcal{D}'$ that is obtained from $\mathcal{D}$ by adding the predicates $=$ and $=_d$ ($d \in \Delta^{\mathcal{D}}$) still satisfies (infinitediff).*

*Proof.* We first show that adding the unary predicates $=_d$, $d \in \Delta^{\mathcal{D}}$, does not affect *(infinitediff)*. To show this, we can without loss of generality restrict ourselves to formulas using only one variable $v$ since different variables cannot interact using only unary predicates. Assume that $\phi$ is a conjunction of unary atoms using only $v$ with $|\mathsf{sol}(\phi)| > 1$, and $\mathsf{sol}(\phi) \not\subseteq \mathsf{sol}(\psi')$ holds for all atoms $\psi'$ of a $\mathcal{D}$-disjunction $\psi$ using only $v$. Since $|\mathsf{sol}(\phi)| > 1$, we already know that $\phi$ cannot contain any of the new predicates $=_d$. Consider now the $\mathcal{D}$-disjunction $\psi^-$ obtained from $\psi$ by removing all atoms of the form $=_d(v)$. Hence, by *(infinitediff)* we know that $\mathsf{sol}(\phi)$ contains infinitely many solutions that are not elements of $\mathsf{sol}(\psi^-)$. However, since $\mathsf{sol}(\psi^-)$ and $\mathsf{sol}(\psi)$ differ only in finitely many elements, we must also have $|\mathsf{sol}(\phi) \setminus \mathsf{sol}(\psi)| = \infty$, as required. Hence, in the following we can assume that $\mathcal{D}$ already contains all the unary predicates $=_d$, $d \in \Delta^{\mathcal{D}}$.

We show below, in Lemma 2.8, that these unary predicates cause *(infinitediff)* to become equivalent to convexity of $\mathcal{D}$. Hence, we know that $\mathcal{D}$ is convex, and need to show that $\mathcal{D}'$ is also convex. For this purpose, consider a satisfiable $\mathcal{D}'$-conjunction $\phi$ that implies a $\mathcal{D}'$-disjunction $\psi$. We can assume without loss of generality that these formulas do not contain constants since atoms containing only constants must be valid in $\mathcal{D}$ and can hence be removed without affecting the solutions of $\phi$ and $\psi$, and atoms of the form $=(d, v)$ or $=(v, d)$ with $d \in \Delta^{\mathcal{D}}$ can be equivalently written as $=_d(v)$.

We will further assume that $\phi$ does not contain any equality atoms, which is without loss of generality due to the following arguments. Consider the equivalence relation $=$ on terms generated by the equalities occurring in $\phi$. We obtain the $\mathcal{D}$-conjunction $\phi/_=$ from $\phi$ by replacing all variables of an $=$-equivalence class $E$ by a single fresh variable $v_E$ and removing all equality atoms. Every solution $f \in \mathsf{sol}(\phi)$ yields a solution $f/_=$ of $\phi/_=$ by setting $f/_=(v_E) \coloneqq f(v)$ for any $v \in E$. Vice versa, from $f/_= \in \mathsf{sol}(\phi/_=)$ we obtain $f \in \mathsf{sol}(\phi)$ via $f(v) \coloneqq f/_=(v_E)$ for all $v \in E$. These two mappings describe a bijection between $\mathsf{sol}(\phi)$ and $\mathsf{sol}(\phi/_=)$. We apply the same variable replacement to $\psi$ to obtain a $\mathcal{D}'$-disjunction $\psi'$. It is easy to see that $\phi/_=$ still implies $\psi'$, and that $\phi$ implies a disjunct of $\psi$ iff $\phi/_=$ implies a disjunct of $\psi'$.

We can hence assume that $\phi$ is of the form $\phi_1(v_1) \wedge \cdots \wedge \phi_n(v_n)$, where the $\phi_i(v_i)$ are independent conjunctions of unary atoms over different variables $v_i$, $1 \leq i \leq n$. Likewise, $\psi$ can be written as $\psi_1(v_1) \vee \cdots \vee \psi_n(v_n) \vee \psi^=$, where each $\psi_i(v_i)$, $1 \leq i \leq n$, is a disjunction of unary atoms over $v_i$, and $\psi^=$ contains all binary equality atoms.

If $\phi$ implies one of the disjunctions $\psi_i(v_i)$, then convexity of $\mathcal{D}$ yields the claim. Otherwise, we know that $S_i \coloneqq \mathsf{sol}(\phi_i(v_i)) \setminus \mathsf{sol}(\psi_i(v_i))$ is non-empty, for all $i \in \{1, \ldots, n\}$. In all cases where $S_i = \{d_1, \ldots, d_k\}$ is finite, we obtain that $\phi_i(v_i)$ implies $\psi_i(v_i) \vee =_{d_1}(v_i) \vee \cdots \vee =_{d_k}(v_i)$. Since we assumed that $\phi_i(v_i)$ does not imply $\psi_i(v_i)$, by the convexity of $\mathcal{D}$ it must imply one of the atoms $=_{d_j}(v_i)$, which means that $\mathsf{sol}(\phi_i(v_i)) = \{d_j\}$.

Consider the (partial) variable assignment $f$ that assigns all such variables $v_i$ their uniquely determined solutions. If $f$ satisfies one of the equality atoms in $\psi^=$, this already shows that $\phi$ implies this atom. Otherwise, we proceed to extend $f$ to the remaining variables. For any variable $v_i$ that does not yet have a value under $f$, we know that $\mathsf{sol}(\phi_i(v_i)) \setminus \mathsf{sol}(\psi_i(v_i))$ is infinite. Hence, it is possible to find a value from $\mathsf{sol}(\phi_i(v_i))$ that is different from all values of $f$ that have already been fixed. After we have done this for all the remaining variables, we have found a solution $f$ of $\phi$ that does not satisfy any of the equality atoms in $\psi^=$, and by assumption also none of the remaining atoms of $\psi$. This contradicts the assumption that $\phi$ implies $\psi$.

In summary, we have shown that either $\phi$ must already imply one of the disjunctions $\psi_i(v_i)$, or

the singleton solution sets among $\mathsf{sol}(\phi_i(v_i))$ imply an equality atom in $\psi^{=}$, which concludes the proof of convexity. □

The properties convexity [6] and *(infinitediff)* [37, 38] were introduced independently to solve different problems in the presence of concrete domains. The former was used together with polynomiality of $\mathcal{D}$ to show that reasoning in an extension of the description logic $\mathcal{EL}$ remains polynomial. Closer to this paper, *(infinitediff)* was introduced for unary concrete domains, in order to obtain an FO rewriting for CQs in a dialect of *DL-Lite*. Surprisingly, we can show that in our setting these two seemingly unrelated properties are actually equivalent. In general, convexity is a weaker restriction since it does not force non-singleton predicates to be infinite. But in the presence of $=_d$ we can show equivalence. In fact, if $\mathsf{sol}_V(\phi) \setminus \mathsf{sol}_V(\psi)$ is finite, then one can use the predicates $=_d$ to construct a counterexample to convexity. In contrast to the previous lemma, this result is not restricted to unary concrete domains.

**Lemma 2.8.** *A concrete domain $\mathcal{D}$ containing the predicates $=_d$ ($d \in \Delta^{\mathcal{D}}$) is convex iff it satisfies (infinitediff).*

*Proof.* Consider a $\mathcal{D}$-conjunction $\phi$ and a $\mathcal{D}$-disjunction $\psi$, and let $V \coloneqq \mathsf{Var}(\phi) \cup \mathsf{Var}(\psi)$.

($\Rightarrow$) Suppose that $|\mathsf{sol}_V(\phi)| > 1$, $\mathsf{sol}_V(\phi) \not\subseteq \mathsf{sol}_V(\psi')$ for every atom $\psi'$ in $\psi$, and that $\mathsf{sol}_V(\phi) \setminus \mathsf{sol}_V(\psi)$ is finite. Let $S \coloneqq \{f_1, \dots, f_m\} \coloneqq \mathsf{sol}_V(\phi) \setminus \mathsf{sol}_V(\psi)$ be the finitely many solutions of $\phi$ that do not satisfy $\psi$. Then we have

$$\begin{aligned}
\mathsf{sol}_V(\phi) &\subseteq \mathsf{sol}_V(\psi) \cup S \\
&= \mathsf{sol}_V(\psi) \cup \bigcup_{i=1}^{m} \mathsf{sol}_V\Big( \bigwedge_{v \in V} {=_{f_i(v)}}(v) \Big) \\
&= \mathsf{sol}_V\Big( \psi \vee \bigvee_{i=1}^{m} \bigwedge_{v \in V} {=_{f_i(v)}}(v) \Big) \\
&= \mathsf{sol}_V\Big( \psi \vee \bigwedge_{(v_1,\dots,v_m) \in V^m} \bigvee_{i=1}^{m} {=_{f_i(v_i)}}(v_i) \Big) \\
&= \bigcap_{(v_1,\dots,v_m) \in V^m} \mathsf{sol}_V\Big( \psi \vee \bigvee_{i=1}^{m} {=_{f_i(v_i)}}(v_i) \Big).
\end{aligned}$$

Hence, $\phi$ implies $\psi \vee \bigvee_{i=1}^{m} {=_{f_i(v_i)}}(v_i)$, for all $(v_1, \dots, v_m) \in V^m$. Hence, by convexity of $\mathcal{D}$ and our assumption that $\phi$ does not imply any single atom of $\psi$, for every $(v_1, \dots, v_m) \in V^m$ there must be an index $i$, $1 \leq i \leq m$, such that $\mathsf{sol}_V(\phi) \subseteq \mathsf{sol}_V({=_{f_i(v_i)}}(v_i))$, i.e., all solutions of $\phi$ map $v_i$ to $f_i(v_i)$. Consider now the constant tuples of the form $(v, \dots, v) \in V^m$. By the above property, we know that, for all $v \in V$, there is an index $i_v$ such that all solutions of $\phi$ map $v$ to $f_{i_v}(v)$. Hence, either $\mathsf{sol}_V(\phi)$ is empty or it is a singleton set containing only the solution that maps all $v \in V$ to $f_{i_v}(v)$. This contradicts our assumption that $|\mathsf{sol}_V(\phi)| > 1$.

($\Leftarrow$) Assume that $\mathsf{sol}_V(\phi) \subseteq \mathsf{sol}_V(\psi)$ holds and $\psi$ contains at least one atom. If $\mathsf{sol}_V(\phi)$ is empty, then clearly $\mathsf{sol}_V(\phi) \subseteq \mathsf{sol}_V(\psi')$ holds for all atoms $\psi'$ in $\psi$, and hence the claim is trivial. If $|\mathsf{sol}_V(\phi)| = 1$, then the unique solution $f$ of $\phi$ must satisfy an atom $\psi'$ in $\psi$, and thus $\mathsf{sol}_V(\phi) = \{f\} \subseteq \mathsf{sol}_V(\psi')$. Finally, consider the case that $|\mathsf{sol}_V(\phi)| > 1$. Since $\mathsf{sol}_V(\phi) \subseteq \mathsf{sol}_V(\psi)$, we have $|\mathsf{sol}_V(\phi) \setminus \mathsf{sol}_V(\psi)| = 0$, which is finite, and hence *(infinitediff)* implies the existence of an atom $\psi'$ of $\psi$ such that $\mathsf{sol}_V(\phi) \subseteq \mathsf{sol}_V(\psi')$. □

Note that both convexity and *(infinitediff)* only talk about *finite* disjunctions. Hence, it is still possible for (conjunctions of) predicates to have infinite coverings. For example, $\top_{\mathbb{Q}}$ is covered by the union of all predicates $>_q$ with $q \in \mathbb{Q}$.

As a further step towards showing that our results imply the ones in [38], observe that every unary concrete domain $\mathcal{D}$ is trivially functional. We will finish this discussion in Section 6.6, where we also show that polynomiality (and decidability of implications using the predicate $=$) and constructivity are not needed for our results if $\mathcal{D}$ is unary. In contrast, in the presence of predicates of higher arity, the predicates $=_d$, polynomiality, and constructivity are essential for our combined rewriting approach (see Section 6).

Convexity is also necessary for our combined rewritability results since they imply polynomial data complexity. In fact, if the concrete domain is not convex, then answering conjunctive queries that can refer to concrete domain predicates is co-NP-hard in data complexity (and hence neither FO or combined rewritable, unless P = NP), even in the unary case [4, 37, 38]. For the case of predicates of larger arity, convexity alone is not sufficient: we also need functionality and constructivity of $\mathcal{D}$.

# 3 The Ontology Language

For any cr-admissible concrete domain $\mathcal{D}$, we introduce the logic *DL-Lite*$_{core}^{(\mathcal{HF})}(\mathcal{D})$, a common extension of *DL-Lite*$_{core}^{(\mathcal{HF})}$ and *DL-Lite*$_{\mathcal{A}}$ [3, 35].

## 3.1 Syntax

Let $\mathsf{N_C}$, $\mathsf{N_R}$, $\mathsf{N_A}$, and $\mathsf{N_I}$ be pairwise disjoint sets of *concept*, *role*, *attribute*, and *individual names*, respectively. A *role* is either a role name or an *inverse role* of the form $P^-$, where $P \in \mathsf{N_R}$. A *(basic) concept* is either a concept name, the special concept *top* ($\top$), an *existential restriction* of the form $\exists R$, where $R$ is a role, or an *attribute restriction* of the form $\exists U_1, \ldots, U_m.\Pi$, where $U_1, \ldots, U_m \in \mathsf{N_A}$ and $\Pi$ is an $m$-ary predicate of $\mathcal{D}$.

A *TBox* (or *ontology*) is a finite set of

- *inclusions* $X_1 \sqsubseteq X_2$,
- *disjointness constraints* $\mathsf{disj}(X_1, X_2)$,
- *functionality constraints* $\mathsf{funct}(R)$, and
- *attribute range constraints* $B \sqsubseteq \forall U_1, \ldots, U_m.\Pi$

where $X_1$ and $X_2$ are both either basic concepts, roles, or attribute names, $R$ is a role, $B$ is a concept, $\Pi$ is an $m$-ary predicate of $\mathcal{D}$, and $U_1, \ldots, U_m \in \mathsf{N_A}$. As usual, role names occurring in a functionality constraint are not allowed to occur on the right-hand side of inclusions. Without this restriction, CQs could not be FO-rewritable over *DL-Lite* logics [3].

In contrast to *DL-Lite*$_{core}^{(\mathcal{HF})}$, we do no explicitly have role (a)symmetry or (ir)reflexivity axioms here; they can, however, be simulated as described in [3, Lemma 5.17]. In particular, the (instantiated) canonical model that we construct in Section 5 constitutes a so-called *untangled* (tree-like) model.

An *ABox* is a finite set of

- *concept assertions* $A(a)$,
- *role assertions* $P(a, b)$, and
- *attribute assertions* $U(a, d)$,

where $A \in \mathsf{N_C}$, $P \in \mathsf{N_R}$, $U \in \mathsf{N_A}$, $a, b \in \mathsf{N_I}$, and $d \in \Delta^{\mathcal{D}}$.

Inclusions, constraints, and assertions are collectively called *axioms*. A *knowledge base (KB)* is a pair $\mathcal{K} \coloneqq \langle \mathcal{A}, \mathcal{T} \rangle$ consisting of a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$ that uses only the concept, role, and attribute names occurring in $\mathcal{T}$. We denote by $\Delta^{\mathcal{D}}(\mathcal{T})$ the set of all concrete domain values occurring in $\mathcal{T}$, and similarly define $\Delta^{\mathcal{D}}(\mathcal{A})$ and $\Delta^{\mathcal{D}}(\mathcal{K})$. We use $P^-(a, b)$ as an abbreviation for $P(b, a)$, and define $(P^-)^- \coloneqq P$.

## 3.2 Semantics

An *interpretation* $\mathcal{I}$ consists of a non-empty *domain* $\Delta^{\mathcal{I}}$ (sometimes called *object domain*) and an *interpretation function* $\cdot^{\mathcal{I}}$ that assigns

- to each individual name $a \in \mathsf{N_I}$ an object $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$ such that for all $a, b \in \mathsf{N_I}$ with $a \neq b$ we have $a^{\mathcal{I}} \neq b^{\mathcal{I}}$ *(unique name assumption (UNA))*,
- to each concept name $A \in \mathsf{N_C}$ a set $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$ of objects,
- to each role name $P \in \mathsf{N_R}$ a binary relation $P^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$ between objects,
- to each attribute name $U \in \mathsf{N_A}$ a binary relation $U^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{D}}$ between objects and concrete values.

This function is extended to roles and concepts as follows:

- $(P^-)^{\mathcal{I}} \coloneqq \{(e_1, e_2) \mid (e_2, e_1) \in P^{\mathcal{I}}\}$;
- $\top^{\mathcal{I}} \coloneqq \Delta^{\mathcal{I}}$;
- $(\exists R)^{\mathcal{I}} \coloneqq \{e_1 \mid \exists e_2 \colon (e_1, e_2) \in R^{\mathcal{I}}\}$;
- $e \in (\exists U_1, \ldots, U_m.\Pi)^{\mathcal{I}}$ iff there are $(d_1, \ldots, d_m) \in \Pi^{\mathcal{D}}$ such that $(e, d_i) \in U_i^{\mathcal{I}}$, $1 \leq i \leq m$; and
- $e \in (\forall U_1, \ldots, U_m.\Pi)^{\mathcal{I}}$ iff for all $d_1, \ldots, d_m \in \Delta^{\mathcal{D}}$ with $(e, d_i) \in U_i^{\mathcal{I}}$, $1 \leq i \leq m$, we have $(d_1, \ldots, d_m) \in \Pi^{\mathcal{D}}$;

An interpretation $\mathcal{I}$ *satisfies* (or is a *model* of)

- an inclusion or attribute range constraint $X_1 \sqsubseteq X_2$ if $X_1^{\mathcal{I}} \subseteq X_2^{\mathcal{I}}$;
- a disjointness constraint $\mathsf{disj}(X_1, X_2)$ if $X_1^{\mathcal{I}} \cap X_2^{\mathcal{I}} = \emptyset$;
- a functionality constraint $\mathsf{funct}(R)$ if $(e, e_1), (e, e_2) \in R^{\mathcal{I}}$ implies that $e_1 = e_2$;
- a concept assertion $A(a)$ if $a^{\mathcal{I}} \in A^{\mathcal{I}}$;
- a role assertion $P(a, b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in P^{\mathcal{I}}$;
- an attribute assertion $U(a, d)$ if $(a^{\mathcal{I}}, d) \in U^{\mathcal{I}}$;
- a TBox, ABox, or knowledge base $\mathcal{X}$ if it satisfies all its axioms (written $\mathcal{I} \models \mathcal{X}$).

A KB is *consistent* if it has a model.

### 3.3 Comparison to Other *DL-Lite* Logics

Our logic extends those from [35, 38]. In fact, the missing functionality restrictions on attributes can be expressed using attribute range constraints of the form $\top \sqsubseteq \forall U, U.=$. On top of that, we even allow functional attributes to occur on the right-hand sides of inclusions. In contrast to [4], we do not support number restriction on roles or attributes. But we can at least simulate *conjunctions* in inclusions via the concrete domain. For example, since $=_d(v_1) \wedge =(v_1, v_2)$ implies $=_d(v_2)$, the inclusion $B_1 \sqcap B_2 \sqsubseteq B_3$ can be simulated by the axioms

$$\top \sqsubseteq \exists U_1.\top_{\mathcal{D}},\ \top \sqsubseteq \exists U_2.\top_{\mathcal{D}},\ B_1 \sqsubseteq \forall U_1.=_d,\ B_2 \sqsubseteq \forall U_1, U_2.=,\ \exists U_2.=_d \sqsubseteq B_3,$$

where $U_1$, $U_2$ are fresh attribute names, and $d$ is a fresh constant.

**Example 3.1.** We consider $\mathcal{D}_{\mathbb{Q}}$ from Example 2.4. The TBox

$$\begin{aligned}\mathcal{T} \coloneqq \{&\mathsf{Patient} \sqsubseteq \exists \mathsf{maxHR}.\top_{\mathcal{D}_{\mathbb{Q}}},\ \exists \mathsf{age}.=_{60} \sqsubseteq \forall \mathsf{maxHR}.=_{160},\\ &\mathsf{Patient} \sqsubseteq \forall \mathsf{hr}, \mathsf{hr}.=,\ \exists \mathsf{maxHR}, \mathsf{hr}.+_5 \sqsubseteq \mathsf{Alert}\}\end{aligned}$$

says that every patient has a maximum heart rate, which is 160 for persons of age 60, there is always only one heart rate measurement, and an alert is raised when the measured heart rate rises to only 5 below the maximum heart rate. A corresponding ABox contains actual data such as

$$\begin{aligned}\mathcal{A} \coloneqq \{&\mathsf{Patient}(p_1),\ \mathsf{age}(p_1, 60),\ \mathsf{hr}(p_1, 155),\\ &\mathsf{Patient}(p_2),\ \mathsf{hr}(p_2, 155),\ \mathsf{maxHR}(p_2, 180)\},\end{aligned}$$

which implies the assertion $\mathsf{Alert}(p_1)$, but not $\mathsf{Alert}(p_2)$.

This example illustrates the most prominent advantage of our logic, namely attribute restrictions using predicates of arity greater than 1. Here, they allow us to express the concept of an *alert* by comparing the current measurement with a maximum value. Using unary predicates, one could express hard-coded limits like $\exists \mathsf{hr}.>_{180} \sqsubseteq \mathsf{Alert}$, but not comparisons with an (age-dependent) maximum rate, unless one writes a huge (finite) case distinction.

Usually, existential and attribute restrictions in *DL-Lite* are *unqualified*, i.e., of the form $\exists R$ or $\exists U$ without further restriction on the type of the $R$- or $U$-filler. However, *qualified* existential restrictions (over non-functional roles) on the right-hand side of concept inclusions can be simulated as usual [3, 35]: If $R$ is not functional, i.e., the TBox does not contain the constraint $\mathsf{funct}(R)$, then $A \sqsubseteq \exists R.B$ can be expressed by $A \sqsubseteq \exists P_B$, $P_B \sqsubseteq R$, and $\exists P_B^- \sqsubseteq B$, where $P_B$ is a fresh role name. Similarly, we could simulate attribute restrictions on the right-hand side of inclusions using only unqualified attribute restrictions in conjunction with attribute range restrictions and attribute inclusions. For example, to express $B \sqsubseteq \exists U, V.\Pi$, we could write $B \sqsubseteq \exists U_\Pi$, $B \sqsubseteq \exists V_\Pi$, $U_\Pi \sqsubseteq U$, $V_\Pi \sqsubseteq V$, and $B \sqsubseteq \forall U_\Pi, V_\Pi.\Pi$, where $U_\Pi, V_\Pi$ are fresh attribute names.

As in OWL 2 QL, but in contrast to many *DL-Lite* dialects, we allow qualified attribute restrictions on the *left-hand side* of inclusions. Since such restrictions only affect each abstract domain element individually, these concepts are harmless for our purposes. In contrast, if we would additionally allow *feature chains*, which are often used in more expressive description logics to compare attributes of different domain elements, we could easily prove undecidability of CQ answering by a reduction from the Post Correspondence Problem [5, 26], e.g., using the concatenation and equality predicates provided by $\mathcal{D}_{\Sigma^*}$.

## 4 Conjunctive Queries with Built-ins

We consider a set of variables $\mathsf{N_V}$ that is partitioned into the two sets $\mathsf{N_{OV}}$ (*object variables*) and $\mathsf{N_{CV}}$ (*concrete domain variables*). Elements of $\mathsf{N_I} \cup \mathsf{N_{OV}}$ are called *object terms*, and those

of $\Delta^{\mathcal{D}} \cup \mathsf{N_{CV}}$ are *value terms*.

**Definition 4.1** (CQs)**.** A *conjunctive query* $\phi$ is of the form $(\vec{x}, \vec{v}) \leftarrow \psi(\vec{y}, \vec{w})$, where

- $\vec{x}, \vec{y}$ are vectors over $\mathsf{N_{OV}}$;
- $\vec{v}, \vec{w}$ are vectors over $\mathsf{N_{CV}}$;
- all variables occurring in $(\vec{x}, \vec{v})$ also occur in $(\vec{y}, \vec{w})$; and
- $\psi$ is a conjunction of *atoms* of the following forms, using exactly the variables in $(\vec{y}, \vec{w})$:
  - $A(x)$ (concept atom),
  - $P(x, y)$ (role atom),
  - $U(x, v)$ (attribute atom),
  - $x = y$ (object equality atom), or
  - $\Pi(v_1, \ldots, v_m)$ (value comparison atom),

  where $A \in \mathsf{N_C}$, $P \in \mathsf{N_R}$, $U \in \mathsf{N_A}$, $x, y$ are object terms, $v, v_1, \ldots, v_m$ are value terms, and $\Pi$ is an $m$-ary predicate of $\mathcal{D}$.

The set of *answer variables* (or *distinguished variables*) of $\phi$, denoted by $\mathsf{FVar}(\phi)$, consists of the variables occurring in $(\vec{x}, \vec{v})$. The remaining variables in $(\vec{y}, \vec{w})$ are called *existentially quantified* (or *nondistinguished*). As for assertions, we may write $P^-(x, y)$ instead of $P(y, x)$. A CQ is called *Boolean* if it does not have any answer variables. We write $\alpha \in \phi$ to denote that $\alpha$ is an atom occurring in the CQ $\phi$. The set $\mathsf{terms}(\phi)$ contains all elements of $\mathsf{N_I}$, $\Delta^{\mathcal{D}}$, and $\mathsf{N_V}$ that occur in $\phi$, $\Delta^{\mathcal{D}}(\phi)$ denotes the set of all concrete domain values that occur in $\phi$, and we similarly define $\mathsf{N_I}(\phi)$, $\mathsf{N_{CV}}(\phi)$, et cetera.

An interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *satisfies* (or is a *model* of) a Boolean CQ $\phi$ (written $\mathcal{I} \models \phi$) if there is a *homomorphism* of $\phi$ into $\mathcal{I}$, which is a mapping $\pi \colon \mathsf{terms}(\phi) \to \Delta^{\mathcal{I}} \cup \Delta^{\mathcal{D}}$ such that

- $\pi$ maps $\mathsf{N_{OV}}(\phi)$ into $\Delta^{\mathcal{I}}$, and $\mathsf{N_{CV}}(\phi)$ into $\Delta^{\mathcal{D}}$;
- $\pi(a) = a^{\mathcal{I}}$ for all $a \in \mathsf{N_I}(\phi)$;
- $\pi(d) = d$ for all $d \in \Delta^{\mathcal{D}}(\phi)$;
- $\pi(x) \in A^{\mathcal{I}}$ for all concept atoms $A(x) \in \phi$;
- $(\pi(x), \pi(y)) \in P^{\mathcal{I}}$ for all role atoms $P(x, y) \in \phi$;
- $(\pi(x), \pi(v)) \in U^{\mathcal{I}}$ for attribute atoms $U(x, v) \in \phi$;
- $\pi(x) = \pi(y)$ for all object equality atoms $x = y \in \phi$; and
- $(\pi(v_1), \ldots, \pi(v_m)) \in \Pi^{\mathcal{D}}$ for all value comparison atoms $\Pi(v_1, \ldots, v_m) \in \phi$.

A KB $\mathcal{K}$ *entails* a Boolean CQ $\phi$ (written $\mathcal{K} \models \phi$) if every model $\mathcal{I}$ of $\mathcal{K}$ is also a model of $\phi$.

A *potential answer* to a CQ $\phi$ w.r.t. $\mathcal{K}$ is a mapping $\mathfrak{a} \colon \mathsf{FVar}(\phi) \to \mathsf{N_I}(\mathcal{K}) \cup \Delta^{\mathcal{D}}$ that maps all distinguished object variables into $\mathsf{N_I}(\mathcal{K})$ and all distinguished concrete domain variables into $\Delta^{\mathcal{D}}$. A *certain answer* to $\phi \colon (\vec{x}, \vec{v}) \leftarrow \psi(\vec{y}, \vec{w})$ w.r.t. $\mathcal{K}$ is an answer tuple of the form $\mathfrak{a}(\vec{x}, \vec{v})$, where $\mathfrak{a}$ is a potential answer for which $\mathcal{K}$ entails the Boolean CQ $\mathfrak{a}(\phi) \colon () \leftarrow \psi(\mathfrak{a}(\vec{y}, \vec{w}))$. The set of all certain answers to $\phi$ w.r.t. $\mathcal{K}$ is denoted by $\mathsf{cert}(\phi, \mathcal{K})$. Similarly, for an interpretation $\mathcal{I}$, we denote by $\mathsf{ans}_{\mathcal{K}}(\phi, \mathcal{I})$ the set of all tuples $\mathfrak{a}(\vec{x}, \vec{v})$, where $\mathfrak{a}$ is a potential answer to $\phi$ w.r.t. $\mathcal{K}$ such that $\mathcal{I} \models \mathfrak{a}(\phi)$. We usually omit the subscript $\mathcal{K}$ since it is clear from the context.

## 4.1 Rewritability

Given any query language and ontology language, first-order (FO) rewritability of a query $\phi$ w.r.t. a TBox $\mathcal{T}$ is the property that one can find a first-order query expression that, when evaluated over any ABox $\mathcal{A}$ viewed as a (closed-world) database, yields the same answers as $\phi$ on the KB $\langle \mathcal{A}, \mathcal{T} \rangle$. It was shown in [8] that this is actually equivalent to rewritability into a union (disjunction) of CQs. *Combined* rewritability generalizes this approach by allowing the ABox to be rewritten as well, by incorporating some information from the TBox. These techniques allow us to employ existing relational database systems for query answering over ontologies [13, 25].

**Definition 4.2.** A CQ $\phi$ is *FO-rewritable* w.r.t. a TBox $\mathcal{T}$ if there is a finite set $\Phi_{\mathcal{T}}$ of CQs such that for every consistent KB $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ we have

$$\mathsf{cert}(\phi, \mathcal{K}) = \bigcup_{\phi' \in \Phi_{\mathcal{T}}} \mathsf{ans}(\phi', \mathcal{I}(\mathcal{A})),$$

where $\mathcal{I}(\mathcal{A}) \coloneqq \mathcal{I}_{\mathcal{K}}^{(0)}$ is the finite interpretation that will be formally defined in Section 5.2, which satisfies exactly the assertions in $\mathcal{A}$. The set $\Phi_{\mathcal{T}}$ is called a *rewriting* of $\phi$ w.r.t. $\mathcal{T}$.

The CQ $\phi$ is *combined rewritable* w.r.t. $\mathcal{T}$ if there is a finite set $\Phi_{\mathcal{T}}$ of CQs such that for every consistent KB $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ we have

$$\mathsf{cert}(\phi, \mathcal{K}) = \bigcup_{\phi' \in \Phi_{\mathcal{T}}} \mathsf{ans}(\phi', \mathcal{I}(\mathcal{K})),$$

where $\mathcal{I}(\mathcal{K})$ is a finite interpretation that is constructed from $\mathcal{A}$ and $\mathcal{T}$ in polynomial time.

It is important that $\Phi_{\mathcal{T}}$ does not depend on the ABox, and $\mathcal{I}(\mathcal{K})$ does not depend on the query. This allows to draw close connections between these rewritability properties and the complexity of CQ entailment. FO rewritability implies that entailment of (Boolean) CQs is in the circuit complexity class $\mathrm{AC}^0$ w.r.t. *data complexity*, where the query and TBox are viewed as fixed, and only the ABox is counted as part of the input. Hence, the size of the rewriting $\Phi_{\mathcal{T}}$ does not affect the data complexity, as long as it is finite. For combined rewritability, since $\mathcal{I}(\mathcal{K})$ is of size polynomial in the size of $\mathcal{A}$, CQ entailment is in P. The interpretation $\mathcal{I}(\mathcal{K})$ is required to be polynomial since it would be impractical to increase an already large dataset by a superpolynomial amount [17, 24, 29].

Often, it is also desirable that the size of $\Phi_{\mathcal{T}}$ is at most polynomial in the size of $\phi$ and $\mathcal{T}$. However, for *DL-Lite*$_{core}^{\mathcal{H}}$ ontologies in general, it is impossible to find an FO rewriting that is a polynomial-sized union of CQs [21]. If one allows to rewrite into arbitrary FO formulas, the existence of polynomial rewritings is an open problem, although there are strong indications that the answer to this question is negative [21].

## 4.2 Safety

We follow the approach commonly used for databases and assume that all concrete domain predicates are *built-in* predicates of the database system, i.e., their full (possibly infinite) extensions are known [2, 9, 22, 38]. Although strictly speaking this means that the interpretation $\mathcal{I}(\mathcal{K})$ is not finite anymore, i.e., not a database in the classical sense, for so-called *domain-independent* queries it suffices to be able to check satisfiability of $\mathcal{D}$-conjunctions, which is usually implemented in database systems by using a dedicated solver, e.g., for integer arithmetic. Domain-independence is the crucial requirement that the set of answers should not depend on the chosen domain $\Delta^{\mathcal{D}}$ of available values, but only on the values in $\Delta^{\mathcal{D}}(\mathcal{K})$ [1]. To ensure this condition in our setting, we adopt the following syntactic restriction from [2, 38].

**Definition 4.3** (Safety)**.** Given a CQ $\phi$, a variable $v \in \mathsf{N_{CV}}(\phi)$ is *safe* (in $\phi$) if it occurs in $\phi$ in an atom of the form

a) $U(x, v)$ for some $U \in \mathsf{N_A}$ and $x \in \mathsf{N_V}(\phi)$, or

b) $=_d(v)$ for some $d \in \Delta^{\mathcal{D}}$.

The CQ $\phi$ is *safe* if all its concrete domain variables are safe. A variable $v$ that occurs in an atom $U(x, v)$ in $\phi$ is *bound* to $x$ (in $\phi$). All other variables of $\mathsf{N_{CV}}(\phi)$ are called *unbound*.

A concrete domain variable may be bound to several object variables, and unbound variables are always restricted to a constant value by condition b). Condition b) is not essential for our results, since unbound variables can always be replaced by constants without changing the semantics of the query (if we allow constants in answer tuples). However, this definition of safety is more convenient to formulate the rewriting in Section 6.

While safety is not necessary to obtain rewritability for unary concrete domains [4], the next lemma shows that for concrete domain predicates of higher arity non-safety may actually lead to non-rewritability. Unless P = NP, there cannot even exist a combined rewriting.

**Lemma 4.4.** *In DL-Lite$_{core}^{(\mathcal{HF})}(\mathcal{D}_{\{a\}^*})$, entailment of (non-safe) Boolean CQs is* co-NP*-hard in data complexity.*

*Proof.* The proof is based on the observation that unsafe variables can be used to express projections of $n$-ary predicates, which may not be convex anymore. In particular, the projection of the binary predicates $\mathsf{conc}_w$ of $\mathcal{D}^*_\Sigma$ (see Example 2.4) to the first component yields the unary predicates $\mathsf{pref}_w$ that match all words with prefix $w$, which were shown to be non-convex in [7]. Hence, we can adapt the co-NP-hardness proofs for non-convex concrete domains from [4, 37] for our purposes. These proofs are based on a reduction of the satisfiability problem for propositional 2+2-CNF formulas, which is NP-hard and is often used for showing hardness of reasoning problems in description logics [12, 15]. Here, we consider only a singleton alphabet, but the proof can easily be adapted to any finite alphabet (using a union of CQs).

Let $f = c_1 \wedge \cdots \wedge c_n$ be a propositional 2+2-CNF formula over the variables $p_1, \ldots, p_m$, where each clause $c_i$ is of the form $p_1^{(i)} \vee p_2^{(i)} \vee \neg p_3^{(i)} \vee \neg p_4^{(i)}$ for $p_j^{(i)} \in \{p_1, \ldots, p_m, \mathsf{true}, \mathsf{false}\}$. For the reduction, we abuse the notation and treat $c_1, \ldots, c_n$ and $p_1, \ldots, p_m, \mathsf{true}, \mathsf{false}$ as individual names. We further use a concept name $A$, role names $P_1, P_2, N_1, N_2$, and an attribute name $U$. We consider the TBox $\mathcal{T} \coloneqq \{A \sqsubseteq \exists U.\top_{\{a\}^*}\}$ and the ABox

$$\begin{aligned}\mathcal{A}_f \coloneqq \{&A(p_1), \ldots, A(p_m), U(\mathsf{true}, a), U(\mathsf{false}, \varepsilon)\} \cup \\ &\{P_1(c_i, p_1^{(i)}), P_2(c_i, p_2^{(i)}), N_1(c_i, p_3^{(i)}), N_2(c_i, p_4^{(i)}) \mid 1 \leq i \leq n\},\end{aligned}$$

which encodes the formula $f$. Intuitively, the truth value assignments will be given by the values of the attribute $U$ at $p_1, \ldots, p_m$, where $\varepsilon$ represents $\mathsf{false}$, and every other word $a^n$, $n \geq 1$, indicates $\mathsf{true}$.

The following Boolean CQ checks whether there exists an unsatisfied clause:

$$\begin{aligned}\phi \coloneqq () \leftarrow\ & P_1(x_c, x_1) \wedge P_2(x_c, x_2) \wedge N_1(x_c, x_3) \wedge N_2(x_c, x_4) \wedge \\ & U(x_1, v_1) \wedge U(x_2, v_2) \wedge U(x_3, v_3) \wedge U(x_4, v_4) \wedge \\ & =_\varepsilon(v_1) \wedge =_\varepsilon(v_2) \wedge \mathsf{conc}_a(v_3, v_3') \wedge \mathsf{conc}_a(v_4, v_4').\end{aligned}$$

Note that this query is not safe since $v_3'$ and $v_4'$ are not safe.

If $f$ is satisfiable by a variable assignment $\eta$, then we can define an interpretation $\mathcal{I}$ by $\Delta^{\mathcal{I}} \coloneqq \{c_1, \ldots, c_n, p_1, \ldots, p_m, \mathsf{true}, \mathsf{false}\}$ and $U^{\mathcal{I}} \coloneqq \{(p, a) \mid \eta(p) = \mathsf{true}\} \cup \{(p, \varepsilon) \mid \eta(p) = \mathsf{false}\}$,

which can easily be extended by interpreting the other symbols in such a way that we obtain a model of $\langle \mathcal{A}_f, \mathcal{T} \rangle$. However this interpretation does not satisfy $\phi$.

Conversely, assume that $f$ is unsatisfiable, and let $\mathcal{I}$ be any model of $\mathcal{A}_f$ and $\mathcal{T}$. We consider the variable assignment $\eta$ defined by $\eta(p) \coloneqq \mathsf{false}$ if $(p, \varepsilon) \in U^{\mathcal{I}}$, and $\eta(p) \coloneqq \mathsf{true}$, otherwise. Then there must exist a clause $c_i$ such that $(p_1^{(i)}, \varepsilon), (p_2^{(i)}, \varepsilon), (p_3^{(i)}, a^{n_3}), (p_4^{(i)}, a^{n_4}) \in U^{\mathcal{I}}$ holds for some natural numbers $n_3 \geq 1$ and $n_4 \geq 1$. But then $\phi$ can be satisfied in $\mathcal{I}$ by mapping $v_3'$ to $a^{n_3 - 1}$ and $v_4'$ to $a^{n_4 - 1}$.

Since satisfiability of 2+2-CNF formulas is NP-hard [15] and neither TBox nor query depend on the input formula $f$, the CQ entailment problem is co-NP-hard in data complexity. □

# 5 Canonical Models

The usual way to prove rewritability results is via so-called *canonical models* of the input knowledge base. Given a KB $\mathcal{K}$, a canonical model $\mathcal{I}_{\mathcal{K}}$ is a model of $\mathcal{K}$ with the property that $\mathsf{ans}(\phi, \mathcal{I}_{\mathcal{K}}) = \mathsf{cert}(\phi, \mathcal{K})$ holds for all CQs $\phi$. Intuitively, the canonical model must be minimal in the sense that it does not contain any structures that are not enforced by the KB. In logics of the *DL-Lite* family, such models can usually be constructed by iterative application of the inclusions in the TBox to the facts of the ABox. For example, if $A(a) \in \mathcal{A}$, then the canonical model satisfies $a^{\mathcal{I}_{\mathcal{K}}} \in A^{\mathcal{I}_{\mathcal{K}}}$, and the inclusion $A \sqsubseteq \exists R$ would then cause a new domain element $e$ to be created and $(a^{\mathcal{I}_{\mathcal{K}}}, e)$ to be added to $R^{\mathcal{I}_{\mathcal{K}}}$, and so on.

Although the canonical model is usually not finite, it represents a first step in proving rewritability, by reducing query answering under the open-world assumption (i.e., certain answer semantics) to the closed-world assumption (i.e., answers over a single interpretation). Unfortunately, such canonical models need not exist in our setting.

**Example 5.1.** Consider the simple example of the KB $\mathcal{K} = \langle \{A(a)\}, \{A \sqsubseteq \exists U.{>_0}\} \rangle$ over the concrete domain $\mathcal{D}_{\mathbb{Q}}$ from Example 2.4. A canonical model $\mathcal{I}_{\mathcal{K}}$ of $\mathcal{K}$ must satisfy $(a^{\mathcal{I}_{\mathcal{K}}}, q) \in U^{\mathcal{I}_{\mathcal{K}}}$ for some value $q > 0$. However, it is clearly not possible to answer *all* CQs correctly using a single such model: For a fixed $q$, the safe Boolean CQ $\phi_q \colon () \leftarrow \exists v.U(a, v) \wedge {>_{q/2}}(v)$ is satisfied in $\mathcal{I}_{\mathcal{K}}$, but not entailed by $\mathcal{K}$.

In [38], the authors try to solve this problem by selecting $q$ as the "most general" value, which does not satisfy any $\mathcal{D}$-atoms except those implied by the constraint ${>_0}(q)$. More precisely, they propose to choose $q > 0$ such that "for any $m$ predicates $\Pi_1, \ldots, \Pi_m$ in $\mathcal{D}_{\mathbb{Q}}$ such that $\left(\bigcup_{i=1}^m \Pi_i^{\mathcal{D}_{\mathbb{Q}}}\right) \subsetneq (>_0)^{\mathcal{D}_{\mathbb{Q}}}$ it holds that $q \notin \left(\bigcup_{i=1}^m \Pi_i^{\mathcal{D}_{\mathbb{Q}}}\right)$" [38, page 725].[2] For a given choice of $\Pi_1, \ldots, \Pi_m$, such a value $q$ must exist due to *(infinitediff)*. However, it is clear that one cannot find a single value $q$ such that all (infinitely many) predicates $>_{q'}$ in $\mathcal{D}_{\mathbb{Q}}$ are avoided; regardless of the value of $q$, the CQ $\phi_q$ remains a counterexample. This shows that this construction in the presence of $>_q$, $q \in \mathbb{Q}$, contrary to the claim in [38, Example 2].

To overcome this problem, we weaken the requirements on the canonical model by considering only those CQs that use concrete domain predicates from a fixed, finite set of predicates. This solves the issue in Example 5.1 since there are infinitely many predicates $>_{q/2}$, $q \in \mathbb{Q}$, and thus not all CQs $\phi_q$ follow this restriction. For ease of presentation, we assume in the following that all CQs use only the concrete domain predicates from $\mathcal{T}$. We call such CQs *$\mathcal{T}$-restricted*. This assumption does not affect our results regarding the data complexity of CQ entailment since one can add the (constantly many) predicates $\Pi$ occurring in $\phi$ to $\mathcal{T}$ using trivial axioms such as $\top \sqsubseteq \forall U, \ldots, U.\Pi$, where $U$ is a fresh attribute name. Similarly, we can assume as usual that all other symbols occurring in $\phi$ also occur in $\mathcal{T}$. However, in practice this restriction affects the kind of queries a user can ask over a given KB, which is usually fixed in advance.

[2] We translated the syntax of [38] and applied the construction to Example 5.1.

## 5.1 Abstract Interpretations and Their Solutions

Another difference of our approach to the construction of the canonical model in [38] is that, due to the interaction between values enabled by $n$-ary predicates, the value of a concrete domain value cannot be fixed immediately after its introduction. For example, consider the axioms $\top \sqsubseteq \exists U.{>_0}$, $\top \sqsubseteq \exists V.\top_{\mathbb{Q}}$, and $\top \sqsubseteq \forall U, V.{+_4}$. Applying the first inclusion to a domain element $e \in \Delta^{\mathcal{I}_\mathcal{K}}$ creates an attribute filler: $(e, q) \in U^{\mathcal{I}_\mathcal{K}}$. However, at this point is is not yet clear which value $q$ we should take. Only after all inclusions have been applied to this domain element can we try to find a joint solution of all obtained constraints ${>_0}(q)$, $\top_{\mathbb{Q}}(q')$, and ${+_4}(q, q')$. For this reason, we modify the usual canonical model construction [34, 35] to use *abstract* interpretations that allow us to treat $q, q'$ as variables instead of fixed values.

**Definition 5.2** (Abstract interpretation)**.** An *abstract interpretation* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I}, (\Gamma_e)_{e \in \Delta^\mathcal{I}})$ with the *constraint sets* $\Gamma_e$, $e \in \Delta^\mathcal{I}$, is defined like an ordinary interpretation, with the exceptions that

- each attribute name $U$ is interpreted by a binary relation $U^\mathcal{I} \subseteq \Delta^\mathcal{I} \times (\Delta^\mathcal{D} \cup \mathsf{N_{CV}})$, and
- each $\Gamma_e$, $e \in \Delta^\mathcal{I}$, is a set of $\mathcal{D}$-atoms of the form $\Pi(v_1, \dots, v_m)$, where for each variable among the terms $v_o$, $1 \leq o \leq m$, there exists an attribute name $U$ such that $(e, v_o) \in U^\mathcal{I}$.

We denote by $\mathsf{terms}(\mathcal{I})$ the set of all variables and constants occurring in the constraint sets of $\mathcal{I}$, by $\mathsf{Var}(\mathcal{I})$ the set $\mathsf{terms}(\mathcal{I}) \cap \mathsf{N_{CV}}$, and similarly for individual constraint sets. We say that $\mathcal{I}$ *implies* a $\mathcal{D}$-formula $\psi$ if $\psi$ is implied by the union of all constraint sets $\bigcup_{e \in \Delta^\mathcal{I}} \Gamma_e$. To check such an implication, we can restrict ourselves to those variables that are connected to $\mathsf{Var}(\psi)$ via some atoms in the constraint sets. The (abstract) canonical model we construct below will have the additional property that no constraint sets share variables, and hence it suffices to consider a *finite* union of constraint sets.

To work with abstract interpretations, we need to modify those definitions from Sections 3 and 4 that are concerned with $\mathcal{D}$:

- The interpretation of attribute restrictions and the satisfaction of attribute range constraints at a domain element $e$ of $\mathcal{I}$ are lifted to variables by replacing the expression "$(d_1, \dots, d_m) \in \Pi^\mathcal{D}$" by "$\mathcal{I}$ implies $\Pi(d_1, \dots, d_m)$".
- $\mathcal{I}$ satisfies a disjointness constraint $\mathsf{disj}(U_1, U_2)$ for two attribute names $U_1, U_2$ if there are no pairs $(e, v_1) \in U_1^\mathcal{I}$ and $(e, v_2) \in U_2^\mathcal{I}$ for which $\mathcal{I}$ implies $=(v_1, v_2)$.
- Instead of homomorphisms, we consider *(abstract) homomorphisms* $\pi$ of a Boolean CQ $\phi$ into $\mathcal{I}$, which are defined similarly as before, with the exception that elements of $\mathsf{N_{CV}}(\phi)$ may also be mapped to $\mathsf{terms}(\mathcal{I})$, and the satisfaction conditions are modified as follows:
  - For every attribute atom $U(x, v) \in \phi$, there must exist a $w \in \mathsf{terms}(\mathcal{I})$ such that $(\pi(x), w) \in U^\mathcal{I}$ and $=(w, \pi(v))$ is implied by $\mathcal{I}$.
  - For every value comparison atom $\Pi(v_1, \dots, v_m) \in \phi$, the atom $\Pi(\pi(v_1), \dots, \pi(v_m))$ must be implied by $\mathcal{I}$.

All other notions like satisfaction of knowledge bases are defined as for ordinary interpretations, and we use $\models_\mathsf{a}$ instead of $\models$ to differentiate the satisfaction/entailment relations, and similarly write $\mathsf{ans}_\mathsf{a}$ instead of $\mathsf{ans}$.

The first step in finding an (abstract) canonical model $\mathcal{I}_\mathcal{K}$ of a KB $\mathcal{K}$ is to show that it is actually a model, i.e., that $\mathcal{K}$ is consistent. However, it is not enough to show that $\mathcal{I}_\mathcal{K} \models_\mathsf{a} \mathcal{K}$, since this ignores the satisfiability of the constraint sets in $\mathcal{I}_\mathcal{K}$.

**Definition 5.3** (Solution)**.** Let $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}}, (\Gamma_e)_{e\in\Delta^{\mathcal{I}}})$ be an abstract interpretation. A *solution* $f$ of $\mathcal{I}$ is a variable assignment for $\mathsf{Var}(\mathcal{I})$ that satisfies all constraint sets $\Gamma_e$, $e \in \Delta^{\mathcal{I}}$.

Given a solution $f$ of $\mathcal{I}$, the *instance* $f(\mathcal{I})$ of $\mathcal{I}$ defined by $f$ is the ordinary interpretation obtained from $\mathcal{I}$ by replacing all variables according to $f$ and discarding the constraint sets.

For the rest of this paper, let $\mathcal{D}$ be a cr-admissible concrete domain and $\mathcal{K} = \langle \mathcal{A}, \mathcal{T}\rangle$ be a KB formulated in $\mathit{DL\text{-}Lite}_{core}^{(\mathcal{HF})}(\mathcal{D})$. The goal of this section is to find an abstract canonical model $\mathcal{I}_{\mathcal{K}}$ of $\mathcal{K}$, which should satisfy the following properties:

- $\mathcal{I}_{\mathcal{K}}$ characterizes the consistency of $\mathcal{K}$ in the sense that $\mathcal{K}$ is consistent iff $\mathcal{I}_{\mathcal{K}} \models_{\mathsf{a}} \mathcal{K}$ and $\mathcal{I}_{\mathcal{K}}$ has a solution (see Lemma 5.8). Although an arbitrary solution $f$ of $\mathcal{I}_{\mathcal{K}}$ need not satisfy $f(\mathcal{I}_{\mathcal{K}}) \models \mathcal{K}$, we will construct a *canonical solution* $f_{\mathcal{K}}$ for which this is the case.
- If $\mathcal{K}$ is consistent, then $\mathcal{I}_{\mathcal{K}}$ is a canonical model in the sense that $\mathsf{ans}_{\mathsf{a}}(\phi, \mathcal{I}_{\mathcal{K}}) = \mathsf{cert}(\phi, \mathcal{K})$ holds for all safe CQs $\phi$ formulated over the signature of $\mathcal{K}$ (see Lemma 5.9). Although we could also show that the canonical instance $f_{\mathcal{K}}(\mathcal{I}_{\mathcal{K}})$ is a canonical model in the usual sense, it is much easier to prove our correctness results (Lemmas 5.9 and 6.4) by staying at the level of abstract interpretations.

## 5.2 The Abstract Canonical Model

We construct the abstract canonical model $\mathcal{I}_{\mathcal{K}}$ of $\mathcal{K}$ as the limit of a sequence of abstract interpretations $\mathcal{I}_{\mathcal{K}}^{(\ell)} = \left(\Delta^{\mathcal{I}_{\mathcal{K}}^{(\ell)}}, \cdot^{\mathcal{I}_{\mathcal{K}}^{(\ell)}}, \left(\Gamma_e^{(\ell)}\right)_{e\in\Delta^{\mathcal{I}_{\mathcal{K}}^{(\ell)}}}\right)$, $\ell \geq 0$. The initial interpretation $\mathcal{I}_{\mathcal{K}}^{(0)}$ is based on the assertions in $\mathcal{A}$:

- $\Delta^{\mathcal{I}_{\mathcal{K}}^{(0)}} := \mathsf{N_I}(\mathcal{K})$,
- $a^{\mathcal{I}_{\mathcal{K}}^{(0)}} := a$ and $\Gamma_a^{(0)} := \emptyset$ for all $a \in \mathsf{N_I}(\mathcal{K})$, and
- $X^{\mathcal{I}_{\mathcal{K}}^{(0)}} := \{e \mid X(e) \in \mathcal{A}\}$ for all $X \in \mathsf{N_C} \cup \mathsf{N_R} \cup \mathsf{N_A}$.

For $\ell \geq 0$, $\mathcal{I}_{\mathcal{K}}^{(\ell+1)}$ is obtained from $\mathcal{I}_{\mathcal{K}}^{(\ell)}$ by applying one of the following *completion rules* to an inclusion $X_1 \sqsubseteq X_2 \in \mathcal{T}$ and an element $e \in X_1^{\mathcal{I}_{\mathcal{K}}^{(\ell)}} \setminus X_2^{\mathcal{I}_{\mathcal{K}}^{(\ell)}}$:

**(CR1)** If $X_2 \in \mathsf{N_C} \cup \mathsf{N_R} \cup \mathsf{N_A}$, then $X_2^{\mathcal{I}_{\mathcal{K}}^{(\ell+1)}} := X_2^{\mathcal{I}_{\mathcal{K}}^{(\ell)}} \cup \{e\}$.

**(CR2)** If $X_2 = \exists P$ with $P \in \mathsf{N_R}$, we set $\Delta^{\mathcal{I}_{\mathcal{K}}^{(\ell+1)}} := \Delta^{\mathcal{I}_{\mathcal{K}}^{(\ell)}} \cup \{e_P\}$ and $P^{\mathcal{I}_{\mathcal{K}}^{(\ell+1)}} := P^{\mathcal{I}_{\mathcal{K}}^{(\ell)}} \cup \{(e, e_P)\}$, where $e_P$ is a fresh element.

**(CR3)** If $X_2 = \exists P^-$ with $P \in \mathsf{N_R}$, set $\Delta^{\mathcal{I}_{\mathcal{K}}^{(\ell+1)}} := \Delta^{\mathcal{I}_{\mathcal{K}}^{(\ell)}} \cup \{e_{P^-}\}$ and $P^{\mathcal{I}_{\mathcal{K}}^{(\ell+1)}} := P^{\mathcal{I}_{\mathcal{K}}^{(\ell)}} \cup \{(e_{P^-}, e)\}$, where $e_{P^-}$ is a fresh element.

**(CR4)** If $X_2 = \exists U_1, \ldots, U_m.\Pi$, then for each $i$, $1 \leq i \leq m$, we set $U_i^{\mathcal{I}_{\mathcal{K}}^{(\ell+1)}} := U_i^{\mathcal{I}_{\mathcal{K}}^{(\ell)}} \cup \{(e, v_i)\}$, where $v_i$ is a fresh element of $\mathsf{N_{CV}}$, and $\Gamma_e^{(\ell+1)} := \Gamma_e^{(\ell)} \cup \{\Pi(v_1, \ldots, v_m)\}$.

**(CR5)** If $X_2 = P^-$ with $P \in \mathsf{N_R}$ and $e = (e_1, e_2)$, then we set $P^{\mathcal{I}_{\mathcal{K}}^{(\ell+1)}} := P^{\mathcal{I}_{\mathcal{K}}^{(\ell)}} \cup \{(e_2, e_1)\}$.

The interpretation of all other symbols under $\mathcal{I}_{\mathcal{K}}^{(\ell+1)}$ is the same as under $\mathcal{I}_{\mathcal{K}}^{(\ell)}$, and all other sets $\Gamma_e^{(\ell+1)}$ are defined to be $\Gamma_e^{(\ell)}$.

For attribute range restrictions, we need the following additional completion rule, which is similar to **(CR4)**, but only applies if the attribute values already exist:

**(CR6)** Let $B \sqsubseteq \forall U_1, \dots, U_m.\Pi \in \mathcal{T}$ and $e \in B^{\mathcal{I}_\mathcal{K}^{(\ell)}}$. If there are $v_i$ with $(e, v_i) \in U_i^{\mathcal{I}_\mathcal{K}^{(\ell)}}$, $1 \leq i \leq m$, and $\Gamma_e^{(\ell)}$ does not imply $\Pi(v_1, \dots, v_m)$, then set $\Gamma_e^{(\ell+1)} \coloneqq \Gamma_e^{(\ell)} \cup \{\Pi(v_1, \dots, v_m)\}$.

The *(abstract) canonical model* $\mathcal{I}_\mathcal{K} = \left(\Delta^{\mathcal{I}_\mathcal{K}}, \cdot^{\mathcal{I}_\mathcal{K}}, \left(\Gamma_e\right)_{e \in \Delta^{\mathcal{I}_\mathcal{K}}}\right)$ is defined as the limit of this inductive procedure, i.e., it is obtained by applying the completion rules starting with $\mathcal{I}_\mathcal{K}^{(0)}$ in a fair manner (meaning that each applicable completion rule should be applied at some point). This is possible since the the set of all symbols relevant for this construction (concept names, predicates, etc.) is finite. Note that different domain elements do not share concrete domain variables (neither in the interpretations of attributes nor in the sets $\Gamma_e$).

As usual, the abstract canonical model can be embedded into every model of $\mathcal{K}$ in the following sense.

**Lemma 5.4.** *Let $\mathcal{J} = (\Delta^\mathcal{J}, \cdot^\mathcal{J})$ be a model of $\mathcal{K}$. Then there exist a function $f_\mathsf{o} \colon \Delta^{\mathcal{I}_\mathcal{K}} \to \Delta^\mathcal{J}$ and a solution $f_\mathsf{v}$ of $\mathcal{I}_\mathcal{K}$ such that, for all $A \in \mathsf{N_C}$, $P \in \mathsf{N_R}$, $U \in \mathsf{N_A}$, $a \in \mathsf{N_I}(\mathcal{K})$, $e, e' \in \Delta^{\mathcal{I}_\mathcal{K}}$, and all object terms $v$, we have*

(E1) $f_\mathsf{o}(a) = a^\mathcal{J}$;

(E2) $e \in A^{\mathcal{I}_\mathcal{K}}$ *implies* $f_\mathsf{o}(e) \in A^\mathcal{J}$;

(E3) $(e, e') \in P^{\mathcal{I}_\mathcal{K}}$ *implies* $(f_\mathsf{o}(e), f_\mathsf{o}(e')) \in P^\mathcal{J}$; *and*

(E4) $(e, v) \in U^{\mathcal{I}_\mathcal{K}}$ *implies* $(f_\mathsf{o}(e), f_\mathsf{v}(v)) \in U^\mathcal{J}$.

*Proof.* We construct $f_\mathsf{o}$ and $f_\mathsf{v}$ by induction on the construction of $\mathcal{I}_\mathcal{K}$. We start by setting $f_\mathsf{o}(a) \coloneqq a^\mathcal{J}$ for all $a \in \mathsf{N_I}(\mathcal{K})$ and keeping $f_\mathsf{v}$ undefined everywhere. The conditions (E1)–(E4) are satisfied for the initial interpretation $\mathcal{I}_\mathcal{K}^{(0)}$ and the sets $\Gamma_a^{(0)} = \emptyset$ since $\mathcal{J}$ is a model of $\mathcal{A}$.

Assume now that $\mathcal{I}_\mathcal{K}^{(\ell)}$, $f_\mathsf{o}$, and $f_\mathsf{v}$ have already been partially constructed such that (E1)–(E4) are satisfied and $f_\mathsf{v}$ solves all constraint sets $\Gamma_e^{(\ell)}$. We consider the next application of a completion rule to $\mathcal{I}_\mathcal{K}^{(\ell)}$. Assume that the rule application was triggered by an inclusion $X_1 \sqsubseteq X_2 \in \mathcal{T}$ and an element $e \in X_1^{\mathcal{I}_\mathcal{K}^{(\ell)}} \setminus X_2^{\mathcal{I}_\mathcal{K}^{(\ell)}}$. By the induction hypothesis, we have that the image of $e$ under $f_\mathsf{o}$ and $f_\mathsf{v}$ belongs to $X_1^\mathcal{J}$; we show this only for the case that $X_1 = \exists U_1, \dots, U_m.\Pi$, and hence $e$ is an element of $\Delta^{\mathcal{I}_\mathcal{K}^{(\ell)}}$ (the other cases can be handled by similar arguments). In that case, we know that there are terms $v_i$ with $(e, v_i) \in U_i^{\mathcal{I}_\mathcal{K}^{(\ell)}}$, $1 \leq i \leq m$, such that $\Gamma_e^{(\ell)}$ implies $\Pi(v_1, \dots, v_m)$. By the induction hypothesis, we know that $(f_\mathsf{o}(e), f_\mathsf{v}(v_i)) \in U_i^\mathcal{J}$ and that $f_\mathsf{v}$ solves all atoms in $\Gamma_e^{(\ell)}$. By the above implication, it also solves $\Pi(v_1, \dots, v_m)$, and hence $f_\mathsf{o}(e)$ satisfies $\exists U_1, \dots, U_m.\Pi$ in $\mathcal{J}$.

We now make a case distinction on the type of rule that was applied.

**(CR1)** Consider the case where $X_2 \in \mathsf{N_C}$, and thus $X_1$ is a basic concept. Since $f_\mathsf{o}(e) \in X_1^\mathcal{J}$ and $\mathcal{J} \models \mathcal{T}$, we have $f_\mathsf{o}(e) \in X_2^\mathcal{J}$. This means that adding $e$ to $X_2^{\mathcal{I}_\mathcal{K}^{(\ell)}}$ does not violate (E2). The cases $X_2 \in \mathsf{N_R}$ and $X_2 \in \mathsf{N_A}$ can be treated similarly.

**(CR2)** Since $X_1$ must be a basic concept, we again know that $f_\mathsf{o}(e) \in X_1^\mathcal{J} \subseteq X_2^\mathcal{J} = (\exists P)^\mathcal{J}$. Hence, there must exist an element $e' \in \Delta^\mathcal{J}$ such that $(f_\mathsf{o}(e), e') \in P^\mathcal{J}$. We can thus define $f_\mathsf{o}(e_P) \coloneqq e'$ for the fresh element $e_P$ introduced in the rule, in order to satisfy (E3).

**(CR3)** This rule can be treated similarly.

**(CR4)** We again have $f_\mathsf{o}(e) \in X_1^\mathcal{J} \subseteq X_2^\mathcal{J} = (\exists U_1, \dots, U_m.\Pi)^\mathcal{J}$. This implies that there are $d_i$ with $(f_\mathsf{o}(e), d_i) \in U_i^\mathcal{J}$, $1 \leq i \leq m$, such that $(d_1, \dots, d_m) \in \Pi^\mathcal{D}$. We define $f_\mathsf{v}(v_i) \coloneqq d_i$ for all the fresh variables $v_i$ introduced by the completion rule, and hence (E4) remains satisfied and $f_\mathsf{v}$ also solves the new atom $\Pi(v_1, \dots, v_m)$ in $\Gamma_e^{(\ell+1)}$.

**(CR5)** We have $(f_{\mathsf{o}}(e_1), f_{\mathsf{o}}(e_2)) \in X_1^{\mathcal{J}} \subseteq X_2^{\mathcal{J}} = (P^-)^{\mathcal{J}}$, where $e = (e_1, e_2)$. This implies $(f_{\mathsf{o}}(e_2), f_{\mathsf{o}}(e_1)) \in P^{\mathcal{J}}$, and thus $(e_2, e_1)$ can be added to $P^{\mathcal{I}_{\mathcal{K}}^{(\ell)}}$ without violating (E3).

**(CR6)** If this rule was applied to an attribute range constraint $B \sqsubseteq \forall U_1, \dots, U_m.\Pi \in \mathcal{T}$, $e \in \Delta^{\mathcal{I}_{\mathcal{K}}^{(\ell)}}$, and terms $v_1, \dots, v_m$, then we know as above that $f_{\mathsf{o}}(e) \in B^{\mathcal{J}}$. By the induction hypothesis, we also have $(f_{\mathsf{o}}(e), f_{\mathsf{v}}(v_i)) \in U_i^{\mathcal{J}}$, $1 \leq i \leq m$. Since $\mathcal{J} \models \mathcal{T}$, we obtain $(f_{\mathsf{v}}(v_1), \dots, f_{\mathsf{v}}(v_m)) \in \Pi^{\mathcal{D}}$. Hence, the new atom $\Pi(v_1, \dots, v_m)$ is satisfied by $f_{\mathsf{v}}$. □

## 5.3 ABox Completion

For our combined rewriting, we also consider a variant of the above construction where the completion rules are only applied to the individual names $\mathsf{N}_{\mathsf{I}}(\mathcal{K})$ occurring in the input ABox. More precisely, the variable $e$ in the definition of **(CR1)**–**(CR6)** must either be an element of $\mathsf{N}_{\mathsf{I}}(\mathcal{K})$, or be of the form $(a, e')$ or $(e', a)$ with $a \in \mathsf{N}_{\mathsf{I}}(\mathcal{K})$. In this way, we first obtain the *ABox completion* $\mathcal{I}_{\mathcal{A}}^* = \left(\Delta^{\mathcal{I}_{\mathcal{A}}^*}, \cdot^{\mathcal{I}_{\mathcal{A}}^*}, \left(\Gamma_e^*\right)_{e \in \Delta^{\mathcal{I}_{\mathcal{A}}^*}}\right)$, which may contain fresh role successors for individual names, but they are only used as placeholders and are not further expanded. Most importantly, the interpretation of all basic concepts on $\mathsf{N}_{\mathsf{I}}(\mathcal{K})$ under $\mathcal{I}_{\mathcal{A}}^*$ coincides with the one under $\mathcal{I}_{\mathcal{K}}$.

**Lemma 5.5.** *For every basic concept $B$ and $a \in \mathsf{N}_{\mathsf{I}}(\mathcal{K})$, we have $a \in B^{\mathcal{I}_{\mathcal{A}}^*}$ iff $a \in B^{\mathcal{I}_{\mathcal{K}}}$.*

*Proof.* Since we block some of the applications of completion rules, $\mathcal{I}_{\mathcal{A}}^*$ is a subinterpretation of $\mathcal{I}_{\mathcal{K}}$. Hence, it suffices to show that $a \in B^{\mathcal{I}_{\mathcal{K}}}$ implies $a \in B^{\mathcal{I}_{\mathcal{A}}^*}$. Assume that this does not hold, and let $\ell \geq 0$ be the minimal index for which there exist a basic concept $B$ and $a \in \mathsf{N}_{\mathsf{I}}(\mathcal{K})$ with $a \in B^{\mathcal{I}_{\mathcal{K}}^{(\ell)}}$, but not $a \in B^{\mathcal{I}_{\mathcal{A}}^*}$. Thus, $\mathcal{I}_{\mathcal{K}}^{(\ell-1)}$ agrees with $\mathcal{I}_{\mathcal{A}}^*$ on the interpretation of basic concepts at named individuals, and the application of the completion rule used to obtain $\mathcal{I}_{\mathcal{K}}^{(\ell)}$ from $\mathcal{I}_{\mathcal{K}}^{(\ell-1)}$ is disallowed in the construction of $\mathcal{I}_{\mathcal{A}}^*$. We consider the type of this completion rule.

**(CR1)** Consider a concept inclusion $B_2 \sqsubseteq A \in \mathcal{T}$, where $A \in \mathsf{N}_{\mathsf{C}}$. This inclusion must have been applied directly to $a$ since the interpretation of concept names at different individuals would not affect the interpretation of basic concepts at $a$. Moreover, we must have $A = B$ since no other basic concept can be affected by this rule. Since $a \in B_2^{\mathcal{I}_{\mathcal{K}}^{(\ell-1)}}$, we also have $a \in B_2^{\mathcal{I}_{\mathcal{A}}^*}$. Hence, the completion rule is also applicable in the construction in $\mathcal{I}_{\mathcal{A}}^*$, which means that $a \in B^{\mathcal{I}_{\mathcal{A}}^*}$, contradicting the assumption.

In the case that $R \sqsubseteq S \in \mathcal{T}$ was applied to a pair $(e, e')$, we know that $B = \exists S$ and $e = a$, or $B = \exists S^-$ and $e' = a$. Hence, we know that $a \in (\exists R)^{\mathcal{I}_{\mathcal{A}}^*}$ or $a \in (\exists R^-)^{\mathcal{I}_{\mathcal{A}}^*}$, respectively. This means that the completion rule was also applicable in the construction of $\mathcal{I}_{\mathcal{A}}^*$, and we obtain $a \in B^{\mathcal{I}_{\mathcal{A}}^*}$ in either case, again yielding a contradiction.

The case of attribute names can be handled in the same way.

**(CR2)** For an inclusion $B_2 \sqsubseteq \exists P \in \mathcal{T}$, we again know that it must have been applied directly to $a$ since the rule introduces a fresh domain element. Hence, we can apply the same arguments as above to obtain a contradiction.

**(CR3)** This case can be handled by similar arguments.

**(CR4)** For an inclusion $B_2 \sqsubseteq \exists U_1, \dots, U_m.\Pi$, we again know that the rule must have been applied directly to $a$ since it affects only the attributes and constraints at a single domain element. Moreover, we have $a \in B_2^{\mathcal{I}_{\mathcal{A}}^*}$, i.e., the rule is also applicable in the construction of $\mathcal{I}_{\mathcal{A}}^*$, and $B$ must be of the form $\exists U_1', \dots, U_k'.\Pi'$ such that an atom of the form $\Pi'(v_1', \dots, v_k')$ is implied by $\Gamma_a^{(\ell)}$. However, for each term $v$ in $\Gamma_a^{(\ell-1)}$, there is an attribute name $U$ such that $(a, v) \in U^{\mathcal{I}_{\mathcal{K}}^{(\ell-1)}}$, and we know that $\Gamma_a^{(*)}$ must also contain

a $U$-filler $v^*$ for $a$ since otherwise $\mathcal{I}^*_\mathcal{A}$ and $\mathcal{I}^{(\ell-1)}_\mathcal{K}$ would differ in the interpretation of a basic concept of the form $\exists U.\top_\mathcal{D}$ at $a$. Furthermore, for each atom in $\Gamma^{(\ell-1)}_a$, the atom obtained by replacing each $v$ with $v^*$ must be implied by $\Gamma^{(*)}_a$ since otherwise we can similarly find an attribute restriction on which $\mathcal{I}^{(\ell-1)}_\mathcal{K}$ and $\mathcal{I}^*_\mathcal{A}$ disagree. Since $\mathcal{I}^*_\mathcal{A}$ also contains (or implies) an atom of the form $\Pi(v_1, \ldots, v_m)$, we know that $\Gamma^{(*)}_a$ implies $\Gamma^{(\ell)}_a$ (modulo a substitution), and hence $B = \exists U'_1, \ldots, U'_k.\Pi'$ is satisfied by $a$ in $\mathcal{I}^*_\mathcal{A}$, which is a contradiction.

**(CR5)** This rule can be handled similarly to **(CR1)**.

**(CR6)** This rule can also only affect basic concepts of the form $B = \exists U'_1, \ldots, U'_k.\Pi'$, and can be handled by the same arguments as for **(CR4)**. □

Since $\mathcal{I}^*_\mathcal{A}$ is a subinterpretation of $\mathcal{I}_\mathcal{K}$, the abstract canonical model can equivalently be constructed starting from $\mathcal{I}^*_\mathcal{A}$ instead of $\mathcal{I}^{(0)}_\mathcal{K}$, and we will sometimes assume that it was actually constructed in this alternative way. This representation separates the influence of the ABox from the remainder of the canonical model, which is helpful for deriving our combined rewriting in Section 6. Via instantiation with the canonical solution described in the next section, the ABox completion $\mathcal{I}^*_\mathcal{A}$ will be used to obtain a finite interpretation $\mathcal{I}(\mathcal{K})$ that can be used to evaluate the rewritten query (cf. Definition 4.2).

## 5.4 A Canonical Solution

To construct a canonical solution $f_\mathcal{K}$ of $\mathcal{I}_\mathcal{K}$, we first need to identify critical inferences that should be preserved by this variable assignment. For example, consider the case that $\mathcal{I}_\mathcal{K}$ contains two attribute fillers of $e$, say $(e, u) \in U^{\mathcal{I}_\mathcal{K}}$ and $(e, v) \in V^{\mathcal{I}_\mathcal{K}}$, and $\mathcal{T}$ contains the constraint $\mathsf{disj}(\exists U, V.=, \exists U, V.=)$, stating that $U$- and $V$-values should never be equal. Clearly, although $\mathcal{I}_\mathcal{K}$ may satisfy this constraint, i.e., $\Gamma_e$ does not imply $=(u, v)$, this does not have to be the case in all solutions of $\mathcal{I}_\mathcal{K}$, unless the values of $u$ and $v$ are forced to be different by some constraints in $\Gamma_e$. However, since the canonical instance $f_\mathcal{K}(\mathcal{I}_\mathcal{K})$ should satisfy $\mathcal{T}$ whenever possible, in the construction of $f_\mathcal{K}$ we have to make sure that certain atoms like $=(u, v)$ are not satisfied unnecessarily, i.e., unless they are implied by the constraints.

To identify the set of all these atoms, we first build a set $\mathfrak{R}$ of relevant concrete domain predicates. It will contain all predicates occurring in $\mathcal{T}$, e.g., $=$ in the above example, and additional predicates of the form $=_d$ that may become relevant for the canonical solution. For an object $\mathcal{X}$ (a knowledge base or a constraint set), we first define the set $\mathfrak{R}_\mathcal{X}$ of all relevant concrete domain predicates in $\mathcal{X}$ as

$$\mathfrak{R}_\mathcal{X} \coloneqq \{\Pi \mid \Pi \text{ occurs in } \mathcal{X}\} \cup \{=_d \mid d \in \Delta^\mathcal{D}(\mathcal{X})\}.$$

In addition to the predicates in $\mathfrak{R}_\mathcal{K}$, we also have to consider constants that can be implied by the constraint sets $\Gamma_e$, which are combinations of atoms using predicates from $\mathfrak{R}_\mathcal{K}$. Due to the completion rules, the maximal number of terms in such constraint sets $\Gamma_e$ is bounded by the number of constants in $\Delta^\mathcal{D}(\mathcal{K})$ and the number of occurrences of attribute names in attribute restrictions on the right-hand side of inclusions in $\mathcal{T}$. More precisely, we consider the maximal number

$$n_\mathcal{T} \coloneqq \sum_{B \sqsubseteq \exists U_1, \ldots, U_m.\Pi \in \mathcal{T}} m$$

of variables that can occur in such a constraint set, and derive the following set of all possible constraint sets:

$$\mathbf{\Gamma}_\mathcal{X} \coloneqq \{\Gamma \mid \Gamma \text{ is satisfiable, } |\mathsf{Var}(\Gamma)| \leq n_\mathcal{T},\ \mathfrak{R}_\Gamma \subseteq \mathfrak{R}_\mathcal{X},\ \Delta^\mathcal{D}(\Gamma) \subseteq \Delta^\mathcal{D}(\mathcal{X})\}.$$

If $\mathcal{X}$ is finite and we consider $\mathbf{\Gamma}_{\mathcal{X}}$ modulo the renaming of variables, then there can only be finitely many such combinations $\Gamma$. By construction of $\mathcal{I}_{\mathcal{K}}$, each constraint set $\Gamma_e$ of the abstract canonical model of a consistent $\mathcal{K}$ is an element of $\mathbf{\Gamma}_{\mathcal{K}}$ since it is satisfiable by Lemma 5.4. Furthermore, $\Gamma_e$ belongs to $\mathbf{\Gamma}_{\mathcal{T}}$ whenever $e \notin \mathsf{N_I}$.

By the above observations, also the set

$$\mathfrak{R}_{\mathcal{X},1} \coloneqq \{=_d \mid =_d(v) \text{ is implied by } \Gamma \in \mathbf{\Gamma}_{\mathcal{X}}\}$$

is finite since each constraint set can imply at most one atom $=_d(v)$ for each variable $v$. We further consider the finitely many predicates in

$$\mathfrak{R}_{\mathcal{X},2} \coloneqq \{=_d \mid =_d(v) \text{ is implied by } \Gamma \in \mathbf{\Gamma}_{\mathcal{X}}[\mathfrak{R}_{\mathcal{X},1}]\},$$

where $\mathbf{\Gamma}_{\mathcal{X}}[\mathfrak{R}_{\mathcal{X},1}]$ is defined as $\mathbf{\Gamma}_{\mathcal{X}}$ above, but additionally allows to use the predicates in $\mathfrak{R}_{\mathcal{X},1}$. Hence, it holds that $\mathfrak{R}_{\mathcal{X},1} \subseteq \mathfrak{R}_{\mathcal{X},2}$.

Note that $\mathfrak{R}_{\mathcal{X},2}$ computable since $\mathcal{D}$ is polynomial and constructive: given a (connected subset of) $\Gamma \in \mathbf{\Gamma}_{\mathcal{X}}[\mathfrak{R}_{\mathcal{X},1}]$, we can compute a solution $f$ of $\Gamma$, and then check whether $\Gamma$ implies one of the atoms $=_{f(v)}(v)$. Finally, we define $\mathfrak{R} \coloneqq \mathfrak{R}_{\mathcal{K}} \cup \mathfrak{R}_{\mathcal{K},2} \cup \{=\}$. The predicates in this set are sufficient to find the canonical solution we are looking for.

In the following, we assume that $\Delta^{\mathcal{I}_{\mathcal{K}}} = \mathbb{N}$, and hence we have a natural enumeration $\Gamma_1, \Gamma_2, \ldots$ of the constraint sets that we want to solve. We iteratively build a partial mapping $f_{\mathcal{K}}^{(j)} \colon \mathsf{Var}(\mathcal{I}_{\mathcal{K}}) \to \Delta^{\mathcal{D}}$, $j \geq 0$, which specifies how to replace the variables in $\Gamma_1, \ldots, \Gamma_j$ by actual values. Initially, $f_{\mathcal{K}}^{(0)}$ is undefined everywhere. We now assume that we have already constructed a partial mapping $f_{\mathcal{K}}^{(j)}$, $j \geq 0$, and try to find a valuation for the variables in the next constraint set $\Gamma_{j+1}$. For this purpose, we construct the following sets of positive and negative constraints, respectively:

$$\mathsf{Pos}_{j+1} \coloneqq \Gamma_{j+1} \cup \big\{=_d(v) \mid v \in \mathsf{N_{CV}},\ f_{\mathcal{K}}^{(j)}(v) = d\big\},$$
$$\mathsf{Neg}_{j+1} \coloneqq \big\{\Pi(v_1, \ldots, v_m) \mid \Pi \in \mathfrak{R},\ v_1, \ldots, v_m \in \mathsf{terms}\big(\Gamma_{j+1}\big) \cup \Delta^{\mathcal{D}}(\mathcal{K}) \cup f_{\mathcal{K}}^{(j)}(\mathsf{N_{CV}})\big\}.$$

Intuitively, the solution $f_{\mathcal{K}}^{(j+1)}$ we are looking for should satisfy $\Gamma_{j+1}$ and agree with $f_{\mathcal{K}}^{(j)}$ on the variables of the previous constraint sets, but it should not satisfy any atoms formulated over $\mathfrak{R}$ and the relevant terms from $\mathcal{I}_{\mathcal{K}}$.

Now we can try to solve the positive constraints while respecting the negative constraints as far as possible. We construct the restricted set

$$\mathsf{Neg}_{j+1}^{-} \coloneqq \{\Pi(v_1, \ldots, v_m) \in \mathsf{Neg}_{j+1} \mid \mathsf{Pos}_{j+1} \text{ does not imply } \Pi(v_1, \ldots, v_m)\}$$

of all negative constraint atoms that are not already implied by the positive constraints. Both $\mathsf{Pos}_{j+1}$ and $\mathsf{Neg}_{j+1}^{-}$ are finite since $\Gamma_{j+1}$ is finite and only finitely many values of $f_{\mathcal{K}}^{(j)}$ have already been defined. We now consider the set

$$\mathsf{sol}_{j+1} \coloneqq \mathsf{sol}\big(\bigwedge \mathsf{Pos}_{j+1}\big) \setminus \mathsf{sol}\big(\bigvee \mathsf{Neg}_{j+1}^{-}\big).$$

Due to *(infinitediff)*, there are only three options for the cardinality of this set of solutions:

- If $|\mathsf{sol}_{j+1}| = 1$, then we have no choice but to replace the variables of $\Gamma_{j+1}$ according to the single variable assignment $f \in \mathsf{sol}_{j+1}$, i.e., we set $f_{\mathcal{K}}^{(j+1)}(v) \coloneqq f(v)$ for all $v \in \mathsf{Var}(\mathsf{Pos}_{j+1})$. Note that any variable $v$ that already had a value under $f_{\mathcal{K}}^{(j)}$ is restricted by the atom $=_d(v)$ in $\mathsf{Pos}_{j+1}$ with $d = f_{\mathcal{K}}^{(j)}(v)$, and hence we have $f_{\mathcal{K}}^{(j+1)}(v) = f_{\mathcal{K}}^{(j)}(v)$.
- If $\mathsf{sol}_{j+1}$ is empty, then $\mathsf{Pos}_{j+1}$ must already be unsatisfiable, and hence $\mathcal{K}$ is inconsistent. In this case, we choose arbitrary concrete domain elements to replace the variables.

- Otherwise, $\mathsf{sol}_{j+1}$ must contain infinitely many elements. We choose one such variable assignment and obtain $f_{\mathcal{K}}^{(j+1)}$ as in the first case.

The variable assignment resulting from this infinite construction is denoted by $f_{\mathcal{K}}$. This construction ensures that all necessary concrete domain restrictions are satisfied and that no unnecessary ones from $\mathfrak{R}$ are satisfied. For instance, no value assigned by $f_{\mathcal{K}}^{(j)}$ will be reused by $f_{\mathcal{K}}^{(j+1)}$, unless this is enforced by the atoms in $\Gamma_{j+1}$.

We first show that this construction is not too restrictive, i.e., if $\mathcal{I}_{\mathcal{K}}$ has any solution at all, then $f_{\mathcal{K}}$ is also a valid solution of $\mathcal{I}_{\mathcal{K}}$.

**Lemma 5.6.** *If there is a solution of $\mathcal{I}_{\mathcal{K}}$, then all sets $\mathsf{sol}_j$ considered for the construction of $f_{\mathcal{K}}$ are non-empty, and hence $f_{\mathcal{K}}$ is also a solution of $\mathcal{I}_{\mathcal{K}}$.*

*Proof.* Let $f$ be a solution of $\mathcal{I}_{\mathcal{K}}$. We show the claim by induction on the construction of $f_{\mathcal{K}}^{(j)}$, $j \geq 0$. Assume that it holds for some $j \geq 0$, and consider $f_{\mathcal{K}}^{(j+1)}$. Since different constraint sets do not share variables and $f$ solves $\Gamma_{j+1}$, we can construct a solution $f'$ of $\mathsf{Pos}_{j+1}$ by setting $f'(v) \coloneqq f_{\mathcal{K}}^{(j)}(v)$ for all variables $v$ for which $f_{\mathcal{K}}^{(j)}(v)$ is defined, and $f'(v) \coloneqq f(v)$ for all $v \in \mathsf{Var}(\Gamma_{j+1})$. Since we excluded in $\mathsf{Neg}_{j+1}^{-}$ all atoms from $\mathsf{Neg}_{j+1}$ that are implied by $\mathsf{Pos}_{j+1}$, by *(infinitediff)* we know that $\mathsf{sol}_{j+1}$ cannot be empty: if there is only the one solution $f'$, then it cannot be contained in $\mathsf{sol}_V\big(\bigvee \mathsf{Neg}_{j+1}^{-}\big)$, and if there is more than one solution of $\mathsf{Pos}_{j+1}$, there must even be infinitely many that are not also solutions of $\bigvee \mathsf{Neg}_{j+1}^{-}$. □

We need another technical result that allows us to replace $\mathsf{Pos}_j$ by the constraint sets $\Gamma_1, \dots, \Gamma_j$ in some circumstances. This result crucially depends on the functionality of $\mathcal{D}$.

**Lemma 5.7.** *Assume that there is a solution of $\mathcal{I}_{\mathcal{K}}$ and let $v_1, \dots, v_m \in \mathsf{terms}(\mathcal{I}_{\mathcal{K}})$ and $\Pi \in \mathfrak{R}_{\mathcal{T}}$. If $\Pi(v_1, \dots, v_m)$ is satisfied by $f_{\mathcal{K}}^{(j)}$, $j \geq 0$, then this atom is implied by $\Gamma_1 \cup \dots \cup \Gamma_j$.*

*Proof.* From the assumptions it immediately follows that the atom is contained in $\mathsf{Neg}_j$, and thus implied by $\mathsf{Pos}_j$. We now show the claim by induction on $j$. If $j = 0$, we know that $v_1, \dots, v_m$ must be constants occurring in $\mathcal{A}$, and hence the claim is trivial. Let now $j > 0$. If the atom does not contain any variables from $\Gamma_j$, then it is also contained in $\mathsf{Neg}_{j-1}$, and the claim follows by the induction hypothesis. If this atom contains only variables from $\Gamma_j$, then it is also implied by $\Gamma_j$, which again yields the claim.

It remains to consider the case that $\Pi(v_1, \dots, v_m)$ contains both variables from $\Gamma_j$ as well as variables from some constraint sets that were considered earlier in the construction. Let $d_i \coloneqq f_{\mathcal{K}}^{(j)}(v_i)$, $1 \leq i \leq m$, and $v_{j_1}, \dots, v_{j_n}$ be the variables among $v_1, \dots, v_m$ that do not occur in $\Gamma_j$. Then $\Gamma_j \wedge \bigwedge_{k=1}^{n} =_{d_{j_k}}(v_{j_k})$ implies $\Pi(v_1, \dots, v_m) \wedge \bigwedge_{k=1}^{n} =_{d_{j_k}}(v_{j_k})$. By the functionality of $\mathcal{D}$ and Lemma 5.6, we know that the former conjunction has exactly one solution for the variables $v_1, \dots, v_m$, and hence it implies at least one atom of the form $=_{d_\ell}(v_\ell)$, $1 \leq \ell \leq m$, where $v_\ell \in \mathsf{Var}(\Gamma_j)$. But this atom must already be implied by $\Gamma_j$ since this set does not share variables with the other constraint sets. Since $\Gamma_j \in \mathbf{\Gamma}_{\mathcal{K}}$, we obtain that $=_{d_\ell} \in \mathfrak{R}_{\mathcal{K},1}$.

Consider now the conjunction $\Pi(v_1, \dots, v_m) \wedge =_{d_\ell}(v_\ell)$. Functionality of $\mathcal{D}$ implies that the only solution of this formula maps each $v_{j_k}$ to $d_{j_k}$, $1 \leq k \leq n$, and hence we have $=_{d_{j_k}} \in \mathfrak{R}_{\mathcal{K},2}$ for all $k$, $1 \leq k \leq n$. But then each atom $=_{d_{j_k}}(v_{j_k})$, $1 \leq k \leq n$, must be implied by some $\Gamma_{j'}$, $j' < j$, that contains the variable $v_{j_k}$, due to the construction of $\mathsf{Neg}_{j'}^{-}$. This shows that $\Gamma_1 \cup \dots \cup \Gamma_j$ implies $\Pi(v_1, \dots, v_m) \wedge \bigwedge_{k=1}^{n} =_{d_{j_k}}(v_{j_k})$, which in turn implies $\Pi(v_1, \dots, v_m)$, as required. □

## 5.5 The Abstract Canonical Model is Canonical

We now show the claimed properties of $\mathcal{I}_{\mathcal{K}}$, starting with a characterization of consistency of $\mathcal{K}$.

**Lemma 5.8.** *$\mathcal{K}$ is consistent iff $\mathcal{I}_\mathcal{K} \models_\mathsf{a} \mathcal{K}$ and $\mathcal{I}_\mathcal{K}$ has a solution.*

*Proof.* For the "only if"-direction, let $\mathcal{J}$ be a model of $\mathcal{K}$, and $f_\mathsf{v}$ be the corresponding solution of $\mathcal{I}_\mathcal{K}$ that exists by Lemma 5.4. $\mathcal{I}_\mathcal{K}$ clearly satisfies the ABox of $\mathcal{K}$ due to the construction of $\mathcal{I}_\mathcal{K}^{(0)}$. Furthermore, all inclusions and attribute range constraints are satisfied due to the completion rules.

Assume now that $\mathcal{I}_\mathcal{K}$ violates a functionality constraint $\mathsf{funct}(R) \in \mathcal{T}$. Then, either (i) $\mathcal{I}_\mathcal{K}^{(0)}$ already violates the constraint, or (ii) this violation is the direct result of an application of a completion rule to some $\mathcal{I}_\mathcal{K}^{(\ell)}$, $\ell \geq 0$. In case (i), there must be $a_1, a_2, a_3 \in \mathsf{N_I}$ such that $a_2 \neq a_3$ and $(a_1, a_2), (a_1, a_3) \in R^{\mathcal{I}_\mathcal{K}^{(0)}} \subseteq R^{\mathcal{I}_\mathcal{K}}$. By Lemma 5.4 and the UNA, $\mathcal{J}$ must violate $\mathsf{funct}(R)$, which contradicts our assumption that it is a model of $\mathcal{K}$. Case (ii) cannot be caused by **(CR1)** or **(CR5)** since these rules cannot modify the interpretation of a functional role (recall that such roles are not allowed to occur on the right-hand side of an inclusion). Furthermore **(CR4)** and **(CR6)** do not affect the interpretation of roles. But for any of the remaining two rules this would mean that the element $e$ that triggered the rule was already present in $(\exists R)^{\mathcal{I}_\mathcal{K}^{(\ell)}}$ before the application of the rule, which contradicts the fact that these completion rules only add new elements if the existential restriction was not already satisfied.

Suppose now that $\mathcal{I}_\mathcal{K}$ violates a concept (or role) disjointness constraint $\mathsf{disj}(X_1, X_2)$ in $\mathcal{T}$, i.e., there is an element of $\Delta^{\mathcal{I}_\mathcal{K}}$ (or $\Delta^{\mathcal{I}_\mathcal{K}} \times \Delta^{\mathcal{I}_\mathcal{K}}$) that is contained in $X_1^{\mathcal{I}_\mathcal{K}} \cap X_2^{\mathcal{I}_\mathcal{K}}$. By Lemma 5.4, $\mathcal{J}$ must violate the constraint, which again yields a contradiction. Finally, consider an attribute disjointness constraint $\mathsf{disj}(U_1, U_2) \in \mathcal{T}$ and assume that $(e, v_1) \in U_1^{\mathcal{I}_\mathcal{K}}$, $(e, v_2) \in U_2^{\mathcal{I}_\mathcal{K}}$, and $=\!(v_1, v_2)$ is implied by $\Gamma_e$. Since $f_\mathsf{v}$ satisfies $\Gamma_e$, the pair $(e, d)$ with $d \coloneqq f_\mathsf{v}(v_1) = f_\mathsf{v}(v_2)$ must belong to the interpretation of both $U_1$ and $U_2$ under $\mathcal{J}$, which contradicts the fact that $\mathcal{J}$ satisfies the disjointness constraint.

For the "if"-direction, assume that $\mathcal{I}_\mathcal{K}$ has a solution and $\mathcal{I}_\mathcal{K} \models_\mathsf{a} \mathcal{K}$. By Lemma 5.6, $f_\mathcal{K}$ is a solution of $\mathfrak{I}_\mathcal{K}$, and we now show that $f_\mathcal{K}(\mathcal{I}_\mathcal{K})$ is a model of $\mathcal{K}$. It is easy to see $f_\mathcal{K}(\mathcal{I}_\mathcal{K})$ satisfies the ABox $\mathcal{A}$ due to the construction of $\mathcal{I}_\mathcal{K}^{(0)}$, which does not contain any variables and is a subinterpretation of $\mathcal{I}_\mathcal{K}$. Furthermore, inclusions and disjointness constraints over roles and functionality constraints are obviously not affected by the solution.

We now show that we have $B^{\mathcal{I}_\mathcal{K}} = B^{f_\mathcal{K}(\mathcal{I}_\mathcal{K})}$ for all basic concepts $B$ occurring in $\mathcal{T}$. The claim for concept names and existential restrictions is immediate from the fact that they do not involve the concrete domain. Consider now an attribute restriction $\exists U_1, \dots, U_m.\Pi$, $e \in \Delta^{\mathcal{I}_\mathcal{K}}$, and $(e, d_i) \in U_i^{f_\mathcal{K}(\mathcal{I}_\mathcal{K})}$, $1 \leq i \leq m$, such that $(d_1, \dots, d_m) \in \Pi^\mathcal{D}$. By construction, there are terms $v_i$ with $(e, v_i) \in U_i^{\mathcal{I}_\mathcal{K}}$ and $f_\mathcal{K}(v_i) = d_i$, $1 \leq i \leq m$. Since $\Pi \in \mathfrak{R}_\mathcal{K}$ and $f_\mathcal{K}$ satisfies $\Pi(v_1, \dots, v_m)$, this atom must be implied by the set $\mathsf{Pos}_e$. Since $v_1, \dots, v_m$ only contains variables from $\Gamma_e$, the atom $\Pi(v_1, \dots, v_m)$ is already implied by $\Gamma_e$. This shows that $e \in (\exists U_1, \dots, U_m.\Pi)^{\mathcal{I}_\mathcal{K}}$. Conversely, assume that this holds due to some terms $v_1, \dots, v_m$ occurring in $\Gamma_e$. Then we have $(f_\mathcal{K}(v_1), \dots, f_\mathcal{K}(v_m)) \in \Pi^\mathcal{D}$ since $f_\mathcal{K}$ satisfies $\Gamma_e$, which proves the other direction of the inclusion.

Since the behavior of the basic concepts is not changed by the solution $f_\mathcal{K}$, we can immediately infer that all inclusions and disjointness constraints on concepts remain satisfied in $f_\mathcal{K}(\mathcal{I}_\mathcal{K})$. Consider now an attribute inclusion $U_1 \sqsubseteq U_2 \in \mathcal{T}$. Since every pair $(e, v) \in U_1^{\mathcal{I}_\mathcal{K}}$ is contained in $U_2^{\mathcal{I}_\mathcal{K}}$, we obtain the same relation after instantiation. For an attribute disjointness constraint $\mathsf{disj}(U_1, U_2) \in \mathcal{T}$, assume that there is a pair $(e, d) \in U_1^{f_\mathcal{K}(\mathcal{I}_\mathcal{K})} \cap U_2^{f_\mathcal{K}(\mathcal{I}_\mathcal{K})}$. By construction, there must be terms $v_1, v_2$ such that $(e, v_1) \in U_1^{\mathcal{I}_\mathcal{K}}$, $(e, v_2) \in U_2^{\mathcal{I}_\mathcal{K}}$, and $f_\mathcal{K}(v_1) = f_\mathcal{K}(v_2) = d$. Since the atom $=\!(v_1, v_2)$ is contained in $\mathsf{Neg}_e$, it must be implied by $\mathsf{Pos}_e$, and hence by $\Gamma_e$. This contradicts our assumption that $\mathcal{I}_\mathcal{K}$ satisfies $\mathcal{T}$. Finally, for an attribute range constraint $B \sqsubseteq \forall U_1, \dots, U_m.\Pi \in \mathcal{T}$, consider any $e \in B^{f_\mathcal{K}(\mathcal{I}_\mathcal{K})} = B^{\mathcal{I}_\mathcal{K}}$ and $(e, d_i) \in U_i^{f_\mathcal{K}(\mathcal{I}_\mathcal{K})}$, $1 \leq i \leq m$, which means that there are terms $v_i$ with $(e, v_i) \in U_i^{\mathcal{I}_\mathcal{K}}$ and $f_\mathcal{K}(v_i) = d_i$, $1 \leq i \leq m$. Since

$\mathcal{I}_\mathcal{K}$ satisfies $\mathcal{T}$, we know that $\Gamma_e$ implies $\Pi(v_1, \ldots, v_m)$. Since $f_\mathcal{K}$ satisfies this set, we obtain $(d_1, \ldots, d_m) \in \Pi^\mathcal{D}$, as required. □

It remains to show that $\mathcal{I}_\mathcal{K}$ yields the same answers as $\mathcal{K}$ when evaluating safe CQs.

**Lemma 5.9.** *Let $\phi$ be a safe CQ over the signature of $\mathcal{K}$. If $\mathcal{K}$ is consistent, then we have* $\mathsf{cert}(\phi, \mathcal{K}) = \mathsf{ans}_\mathsf{a}(\phi, \mathcal{I}_\mathcal{K})$.

*Proof.* By the definition of certain answers, it suffices to verify that, for every safe and *Boolean* CQ $\phi$, it holds that $\mathcal{K} \models \phi$ iff $\mathcal{I}_\mathcal{K} \models_\mathsf{a} \phi$.

For the "if"-direction, let $\pi \colon \mathsf{terms}(\phi) \to \Delta^{\mathcal{I}_\mathcal{K}} \cup \Delta^\mathcal{D} \cup \mathsf{Var}(\mathcal{I}_\mathcal{K})$ be a homomorphism of $\phi$ into $\mathcal{I}_\mathcal{K}$, and $\mathcal{J} = (\Delta^\mathcal{J}, \cdot^\mathcal{J})$ be a model of $\mathcal{K}$. By Lemma 5.4, there are two functions $f_\mathsf{o} \colon \Delta^{\mathcal{I}_\mathcal{K}} \to \Delta^\mathcal{J}$ and $f_\mathsf{v} \colon \mathsf{Var}(\mathcal{I}_\mathcal{K}) \to \Delta^\mathcal{D}$ that embed $\mathcal{I}_\mathcal{K}$ into $\mathcal{J}$. We now define the function $\pi' \colon \mathsf{terms}(\phi) \to \Delta^\mathcal{J} \cup \Delta^\mathcal{D}$ by setting $\pi'(x) \coloneqq f_\mathsf{o}(\pi(x))$ for all object terms $x$, and $\pi'(v) \coloneqq f_\mathsf{v}(\pi(v))$ for all value terms $v$. It is straightforward to check that all object variables are mapped into $\Delta^\mathcal{J}$, all concrete domain variables are mapped into $\Delta^\mathcal{D}$, each $a \in \mathsf{N_I}(\phi)$ is mapped to $a^\mathcal{J}$, and all elements of $\Delta^\mathcal{D}$ are mapped to themselves.

We now verify that $\pi'$ is indeed a homomorphism of $\phi$ into $\mathcal{J}$. For any concept atom $A(x) \in \phi$, we have $\pi(x) \in A^{\mathcal{I}_\mathcal{K}}$ by assumption, and hence $\pi'(x) \in A^\mathcal{J}$ by (E2). Similar arguments apply for role atoms, attribute atoms, and object equality atoms. Finally, consider a value comparison atom $\Pi(v_1, \ldots, v_m) \in \phi$, where $\Pi$ is an $m$-ary predicate of $\mathcal{D}$. Since a finite union of constraint sets $\Gamma_e$ implies $\Pi(\pi(v_1), \ldots, \pi(v_m))$, and $f_\mathsf{v}$ solves all these sets, we know that $(\pi'(v_1), \ldots, \pi'(v_m)) = (f_\mathsf{v}(\pi(v_1)), \ldots, f_\mathsf{v}(\pi(v_m))) \in \Pi^\mathcal{D}$, as required.

For the "only if"-direction, assume that $\mathcal{K}$ is consistent and $\mathcal{K} \models \phi$. By the proof of Lemma 5.8, we know that $f_\mathcal{K}$ is a solution of $\mathcal{I}_\mathcal{K}$ and $f_\mathcal{K}(\mathcal{I}_\mathcal{K}) \models \mathcal{K}$, and hence $f_\mathcal{K}(\mathcal{I}_\mathcal{K}) \models \phi$. Let $\pi$ be a homomorphism of $\phi$ into $f_\mathcal{K}(\mathcal{I}_\mathcal{K})$. We define a homomorphism $\pi'$ of $\phi$ into the abstract model $\mathcal{I}_\mathcal{K}$ by setting $\pi'(x) \coloneqq \pi(x)$ for all object terms and concrete domain values in $\phi$. Since $\phi$ is safe, each of the remaining terms $v \in \mathsf{N_{CV}}(\phi)$ must satisfy case a) or b) of Definition 4.3. In case a), $v$ occurs in at least one attribute atom $U(x, v)$ in $\phi$, and thus there exists a term $v' \in \mathsf{terms}\big(\Gamma_{\pi(x)}\big)$ such that $(\pi(x), v') \in U^{\mathcal{I}_\mathcal{K}}$ and $\pi(v) = f_\mathcal{K}(v')$, and we define $\pi'(v) \coloneqq v'$. Otherwise, case b) applies, which means that an atom of the form $=_d(v)$ occurs in $\phi$. Since $\pi$ solves this atom, we must have $\pi(v) = d \in \Delta^\mathcal{D}$, and we set $\pi'(v) \coloneqq d$.

This mapping obviously satisfies all concept, role, and object equality atoms in $\phi$. For an attribute atom $U(x, v)$ that is satisfied by $\pi$ in $f_\mathcal{K}(\mathcal{I}_\mathcal{K})$, consider first the case that $v$ is a constant. Then we know that $(\pi(x), v) \in U^{f_\mathcal{K}(\mathcal{I}_\mathcal{K})}$. If this tuple also occurs in $U^{\mathcal{I}_\mathcal{K}}$, then we are done. Otherwise, there must be a variable $w$ such that $(\pi(x), w) \in U^{\mathcal{I}_\mathcal{K}}$ and $f_\mathcal{K}(w) = v$. Since $=_v \in \mathfrak{R}_\mathcal{K}$, by Lemma 5.7 the atom $=(v, w)$ is implied by $\mathcal{I}_\mathcal{K}$. Hence, the atom $U(x, v)$ is also satisfied under $\pi'$ in the abstract interpretation $\mathcal{I}_\mathcal{K}$. If $v$ is a (nondistinguished) variable, then we know that case a) from above applies due to the atom $U(x, v)$ in $\phi$. Let $U'(x', v)$ be the attribute atom that was chosen to define $\pi'(v)$, i.e., we have $(\pi(x'), w') \in U'^{\mathcal{I}_\mathcal{K}}$, $\pi(v) = f_\mathcal{K}(w')$, and $\pi'(v) = w'$. Similarly, we know that there exists a term $w$ such that $(\pi(x), w) \in U^{\mathcal{I}_\mathcal{K}}$ and $\pi(v) = f_\mathcal{K}(w)$. This means that $f_\mathcal{K}(w) = f_\mathcal{K}(w')$, and hence by Lemma 5.7 the atom $=(w, w')$ is implied by $\mathcal{I}_\mathcal{K}$. Since $\pi'(v) = w'$, the mapping $\pi'$ satisfies the attribute atom $U(x, v)$.

Finally, consider any $\Pi(v_1, \ldots, v_m) \in \phi$. For every variable $v_i$, $1 \leq i \leq m$, to which case a) applies, there is an attribute atom $U_i(x_i, v_i)$ and a term $v_i' \in \mathsf{terms}\big(\Gamma_{\pi'(x_i)}\big)$ as chosen above. If case b) applies or $v_i$ is a constant, then $\pi(v_i) = \pi'(v_i)$ is a constant, and we set $v_i' \coloneqq \pi'(v_i)$. We thus have $(f_\mathcal{K}(v_1'), \ldots, f_\mathcal{K}(v_m')) = (\pi(v_1), \ldots, \pi(v_m)) \in \Pi^\mathcal{D}$ and $\pi'(v_i) = v_i'$, $1 \leq i \leq m$. By Lemma 5.7, we obtain that $\Pi(v_1', \ldots, v_m')$ is implied by $\mathcal{I}_\mathcal{K}$, which shows that $\pi'$ satisfies $\Pi(v_1, \ldots, v_m)$. □

# 6 Rewriting CQs with Built-in Predicates

To obtain a combined rewriting, we extend the approach from [10, 34]. The idea is to construct the rewriting $\Phi_{\mathcal{T}}$ of the initial CQ $\phi$ w.r.t. the TBox $\mathcal{T}$ by iterative application of several operators (called $\mathsf{reduce}$, $\mathsf{split}$, $\mathsf{infer}_{\mathcal{T}}$, and $\mathsf{infer}_{\mathcal{D}}$). Variants of the two basic operators $\mathsf{reduce}$ and $\mathsf{infer}_{\mathcal{T}}$ have first been used in [10, 34]. The former tries to unify redundant atoms in CQs, while the latter applies the TBox inclusions as rewrite rules. Intuitively, $A \sqsubseteq B \in \mathcal{T}$ means that any certain answer to $A(x)$ is also a certain answer to $B(x)$, and hence $A(x)$ should be included in the rewriting of $B(x)$. We need to extend $\mathsf{infer}_{\mathcal{T}}$ to deal also with attribute range restrictions, which behave similarly to inclusions. A special case of this extension for unary concrete domains can be found in [38].

To deal with concrete domain predicates of higher arity, we introduce two new operators. The operator $\mathsf{split}$ allows to "split" two occurrences of a concrete domain variable into separate variables, as long as they are both restricted to the same value by a predicate of the form $=_d$. The operator $\mathsf{infer}_{\mathcal{D}}$ behaves like $\mathsf{infer}_{\mathcal{T}}$, but takes care of implications in the concrete domain instead of the abstract domain.

## 6.1 The Basic Operators

Formally, $\Phi_{\mathcal{T}}$ is the result of iteratively applying

$$\mathsf{step}(\Phi) \coloneqq \Phi \cup \mathsf{reduce}(\Phi) \cup \mathsf{split}(\Phi) \cup \mathsf{infer}_{\mathcal{T}}(\Phi) \cup \mathsf{infer}_{\mathcal{D}}(\Phi)$$

to the initial set $\{\phi\}$, until we reach a fixed-point. In such sets of CQs, we regard CQs as equal if they are equivalent modulo a renaming of the nondistinguished variables. We first define the two operators $\mathsf{reduce}$ and $\mathsf{split}$.

A *substitution* w.r.t. an CQ $\phi$ is a function $\sigma\colon \mathsf{N_V}(\phi) \to \mathsf{terms}(\phi)$ with the property that all variables are mapped to terms of the corresponding type, e.g., elements of $\mathsf{N_{OV}}$ are mapped to $\mathsf{N_I} \cup \mathsf{N_{OV}}$. Such a substitution allows us to unify any variable of $\phi$ with any (compatible) term occurring in $\phi$. We denote by $\mathsf{subst}(\phi)$ the set of all such substitutions, and by $\sigma(\phi)$ the CQ that is obtained from $\phi$ by replacing all variables according to $\sigma$ and subsequently removing duplicate atoms.[3] We now define

$$\mathsf{reduce}(\Phi) \coloneqq \{\sigma(\phi) \mid \phi \in \Phi,\ \sigma \in \mathsf{subst}(\phi)\}.$$

Analogously, we define the operator $\mathsf{split}$, which can separate multiple occurrences of a concrete domain variable, as long as this variable is restricted to a constant value (this is illustrated below by (R6) in Example 6.1). More formally, given a set $\Phi$ of CQs, the set $\mathsf{split}(\Phi)$ contains all CQs that can be obtained from an element $\phi \in \Phi$ that contains an atom of the form $=_d(v)$ by replacing one other occurrence of $v$ with a fresh nondistinguished variable $v'$ and adding the atom $=_d(v')$. Observe that without condition b) in Definition 4.3, the resulting CQ would not necessarily be safe, although it is clearly equivalent to the original CQ.

Before we define the $\mathsf{infer}$ operators, we want to illustrate them on an example.

**Example 6.1.** Consider again $\mathcal{D}_{\mathbb{Q}}$ and $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ from Example 3.1, and the CQ

$$\phi\colon (x) \leftarrow \mathsf{Alert}(x)$$

that asks for all patients with alerts. The only certain answer to $\phi$ w.r.t. $\mathcal{K}$ is $p_1$. To obtain this answer without referring to the TBox $\mathcal{T}$, we have to apply several rewriting steps. In the following, at each step we discuss only one CQ from $\Phi_{\mathcal{T}}$.

[3] The substitution also has to be applied to the tuple of answer variables, and hence the resulting tuple may contain constants and multiple occurrences of the same variable, e.g., $(x, x, a) \leftarrow A(x)$. This does not affect the semantics if we use the convention that constants are not affected by applying a potential answer.

(R1) The inclusion $\exists \mathsf{maxHR}, \mathsf{hr}.{+_5} \sqsubseteq \mathsf{Alert}$ tells us that every individual $x$ that satisfies the left-hand side concept also satisfies $\mathsf{Alert}(x)$, and hence is a certain answer to $\phi$. Thus, the operator $\mathsf{infer}_{\mathcal{T}}$ applies this inclusion to $\phi$ to obtain the following CQ, which simulates the concept $\exists \mathsf{maxHR}, \mathsf{hr}.{+_5}$ with the help of the fresh nondistinguished variables $v$ and $w$:

$$(x) \leftarrow \mathsf{maxHR}(x, v) \wedge \mathsf{hr}(x, w) \wedge {+_5}(v, w).$$

This corresponds to the usual backward chaining rule for *DL-Lite* inclusions, but in contrast to previous work now also refers to a binary concrete domain predicate.

(R2) In $\mathcal{D}_{\mathbb{Q}}$, it holds that ${=_{160}}(v) \wedge {=_{155}}(w)$ implies ${+_5}(v, w)$. Hence, $\mathsf{infer}_{\mathcal{D}}$ can apply this implication in the same way as $\mathsf{infer}_{\mathcal{T}}$ above, replacing ${+_5}(v, w)$ by the conjunction ${=_{160}}(v) \wedge {=_{155}}(w)$:

$$(x) \leftarrow \mathsf{maxHR}(x, v) \wedge {=_{160}}(v) \wedge \mathsf{hr}(x, w) \wedge {=_{155}}(w).$$

Note that this step introduces the new predicate $=_{155}$, which is not present in $\phi$ or $\mathcal{T}$. In order to avoid an infinite rewriting, we obviously have to restrict the kinds of implications that can be applied in this way.

(R3) Since the maximum heart rate of every 60-year-old is 160, in the context of this CQ the atom ${=_{160}}(v)$ is implied by $\mathsf{age}(x, u) \wedge {=_{60}}(u)$, and hence can be replaced by this conjunction. This is similar to an inclusion, but applying $\mathsf{infer}_{\mathcal{T}}$ to the attribute range constraint $\exists \mathsf{age}.{=_{60}} \sqsubseteq \forall \mathsf{maxHR}.{=_{160}}$ in this way does not remove the attribute atom $\mathsf{maxHR}(x, v)$. Hence, we obtain

$$(x) \leftarrow \mathsf{maxHR}(x, v) \wedge \mathsf{age}(x, u) \wedge {=_{60}}(u) \wedge \mathsf{hr}(x, w) \wedge {=_{155}}(w).$$

(R4) Since $v$ does not occur anywhere else, the atom $\mathsf{maxHR}(x, v)$ can now be removed by applying $\mathsf{infer}_{\mathcal{T}}$ to the inclusion $\mathsf{Patient} \sqsubseteq \exists \mathsf{maxHR}.{\top_{\mathcal{D}_{\mathbb{Q}}}}$ (we assume that the atom ${\top_{\mathcal{D}_{\mathbb{Q}}}}(v)$ is implicitly satisfied for all concrete domain variables). This yields

$$(x) \leftarrow \mathsf{Patient}(x) \wedge \mathsf{age}(x, u) \wedge {=_{60}}(u) \wedge \mathsf{hr}(x, w) \wedge {=_{155}}(w).$$

This query can now be evaluated directly over $\mathcal{A}$ to obtain the expected certain answer $p_1$ via the homomorphism mapping $u$ to 60 and $w$ to 155.

To illustrate the operator $\mathsf{split}$, consider now the different CQ

$$(x) \leftarrow \mathsf{maxHR}(x, v) \wedge \mathsf{hr}(x, v)$$

and the modified ABox $\mathcal{A}'$, where $\mathsf{hr}(p_1, 155)$ is replaced by $\mathsf{hr}(p_1, 160)$. Again, the goal is to obtain the certain answer $p_1$ by rewriting the query. However, one cannot directly apply the inclusion $\mathsf{Patient} \sqsubseteq \exists \mathsf{maxHR}.{\top_{\mathcal{D}_{\mathbb{Q}}}}$ to eliminate the first atom since the variable $v$ also occurs in the second atom, i.e., we would lose the information that the maximum temperature is the same as the measured temperature. Instead, we perform the following rewriting steps:

(R5) We make the query more restrictive by introducing the atom ${=_{160}}(v)$ via $\mathsf{infer}_{\mathcal{D}}$. This is correct since this atom implies ${\top_{\mathcal{D}_{\mathbb{Q}}}}(v)$, which is implicitly satisfied. We obtain

$$(x) \leftarrow \mathsf{maxHR}(x, v) \wedge \mathsf{hr}(x, v) \wedge {=_{160}}(v).$$

(R6) Now we can apply the operator $\mathsf{split}$ to $v$ in order to separate its two occurrences as follows:

$$(x) \leftarrow \mathsf{maxHR}(x, v) \wedge {=_{160}}(v) \wedge \mathsf{hr}(x, w) \wedge {=_{160}}(w).$$

This is correct since both variables are still bound to the same constant.

(R7) Now we can rewrite the first two atoms as before to get

$$(x) \leftarrow \mathsf{Patient}(x) \wedge \mathsf{age}(x, u) \wedge =_{60}(u) \wedge \mathsf{hr}(x, w) \wedge =_{160}(w),$$

from which we obtain the desired answer.

Based on this intuition, we can define the operator

$$\mathsf{infer}_{\mathcal{T}}(\Phi) \coloneqq \{\sigma(\phi'') \mid \phi' \in \Phi,\ \phi' \to_{\mathcal{T}} \phi'',\ \sigma \in \mathsf{subst}(\phi'')\},$$

where the relation $\phi' \to_{\mathcal{T}} \phi''$ holds for two safe CQs $\phi', \phi''$ if one of the following cases applies:

- *Inclusions* (cf. (R1) and (R4)):

  There exist an atom $X_2(\vec{x})$ in $\phi'$ and $X_1 \sqsubseteq X_2$ in $\mathcal{T}$, and $\phi''$ is obtained from $\phi'$ by replacing $X_2(\vec{x})$ with $X_1(\vec{x})$. Here, $\vec{x}$ denotes a vector of terms matching the type of $X_2$, e.g., $\vec{x}$ is an object term in case $X_2$ is a basic concept.

  As usual, the expression $(\exists R)(x)$ stands for an atom $R(x, y)$, where $y$ is a unique nondistinguished variable, i.e., it does not occur elsewhere in the CQ. Similarly, $(\exists U_1, \dots, U_m.\Pi)(x)$ abbreviates the *set* of atoms $\{U_1(x, v_1), \dots, U_m(x, v_m), \Pi(v_1, \dots, v_m)\}$, where $v_1, \dots, v_m$ are unique nondistinguished variables. We also allow that $X_2(\vec{x})$ comprises only a subset of these atoms, as long as it includes at least one attribute atom.

- *Attribute range constraints* (cf. (R3)):

  There exist $\Pi(v_1, \dots, v_m)$ in $\phi'$ and $B \sqsubseteq \forall U_1, \dots, U_m.\Pi$ in $\mathcal{T}$, and $\phi''$ is obtained from $\phi'$ by replacing $\Pi(v_1, \dots, v_m)$ with the conjunction of the atoms $B(x), U_1(x, v_1), \dots, U_m(x, v_m)$, where $x$ is an object variable in $\phi'$ that at least one $v_i$ is bound to.

As in previous rewriting algorithms, this operator does not introduce new object variables (except if they occur only once). This is necessary to bound the size of the produced CQs.

## 6.2 Concrete Domain Implications

The operator $\mathsf{infer}_{\mathcal{D}}$ is defined in the same way, based on a similar relation $\to_{\mathcal{D}}$ on CQs. A first naive idea would be to define $\phi' \to_{\mathcal{D}} \phi''$ as follows:

- *Concrete domain implications* (cf. (R2) and (R5)):

  There exists an atom $\Pi(v_1, \dots, v_m)$ in $\phi'$ that is implied by a $\mathcal{D}$-conjunction $\psi$ such that $\phi''$ is obtained by replacing $\Pi(v_1, \dots, v_m)$ with $\psi$, and adding new attribute atoms $U(x, v)$ for the fresh variables $v$ in $\psi$ (where must $U$ occur in $\mathcal{T}$ and $x$ must occur in $\phi'$).

The additional attribute atoms $U(x, v)$ are necessary to ensure safety of the resulting CQ, and as above we make sure that they use only existing object variables. However, without a similar bound on the number of concrete domain variables, this operation may yield CQs of unbounded size, and hence an infinite rewriting. In fact, one could obtain an "infinite FO rewriting" in this way.

To avoid this problem, we introduce a bound on the number of concrete domain variables that are allowed to occur in the CQs produced by $\mathsf{infer}_{\mathcal{D}}$. Recall from Section 5 that an element of the abstract canonical model $\mathcal{I}_{\mathcal{K}}$ may have at most $n_{\mathcal{T}}$ associated concrete domain variables. Moreover, by Lemma 5.9 we know that we only need to consider $\mathcal{I}_{\mathcal{K}}$ when looking for certain answers to CQs. Hence, we only need to consider $n_{\mathcal{T}}$ concrete domain variables bound to each object variable $x$. By a similar argument, the constraint sets in $\mathcal{I}_{\mathcal{K}}$ can only use the predicates of $\mathfrak{R}_{\mathcal{T}}$, and hence we can restrict ourselves to those predicates in the rewriting.

We now call a CQ *bounded* if all its value comparison atoms are of the form

(B1) $\Pi(v_1, \ldots, v_m)$, where $\Pi \in \mathfrak{R}_\mathcal{T}$ and the variables among $v_1, \ldots, v_m$ are bound to at most one object variable $x$. In the set of all such atoms $\Pi(v_1, \ldots, v_m)$, there may occur only $n_\mathcal{T}$ concrete domain variables bound to the same object variable $x$, i.e., these atoms must constitute an element of $\mathbf{\Gamma}_\mathcal{T}$ (recall its definition from Section 5.4). In view of $\mathsf{split}$, however, we count multiple variables that are restricted to the same constant value as if they were only a single variable.

We now amend the definition of $\to_\mathcal{D}$ by requiring that the $\mathcal{D}$-conjunction $\psi$ introduces only atoms of this form, bound to some object variable $x$ that already occurs in $\phi'$. We also ensure that the new attribute atoms $U(x, v)$ bind the fresh variables $v$ only to the $x$ chosen above, i.e., the one associated to the atoms that $v$ occurs in.

Unfortunately, this is still not enough to obtain the desired rewriting. The reason is that the initial CQ $\phi$ itself need not satisfy (B1). In particular, it may contain value comparison atoms whose variables are bound to different object variables. However, due to the functionality of $\mathcal{D}$, such atoms can only be implied by the TBox if all of their variables already satisfy atoms of the form $=_d(v)$ (see also the proof of Lemma 5.7). It remains to find a *finite* set of values $d$ that are relevant in these situations. It turns out that it suffices to consider such values $d$ that are implied by some set of atoms of the form (B1), i.e., those occurring in $\mathfrak{R}_{\mathcal{T},2}$. Recall that, since $\mathcal{D}$ is polynomial and constructive, it is possible to construct $\mathfrak{R}_{\mathcal{T},2}$ (in polynomial time).

We now relax the definition of boundedness by allowing also the following kinds of value comparison atoms:

(B2) Atoms from the original CQ $\phi$, possibly after applying $\mathsf{reduce}$ or $\mathsf{split}$.

(B3) Atoms of the form $=_d(v)$, where $=_d \in \mathfrak{R}_{\mathcal{T},2}$.

In $\to_\mathcal{D}$, we now allow atoms of the form (B2) to be rewritten using a conjunction of atoms satisfying either (B1) or (B3) (possibly after applying a substitution to the resulting query). However, atoms of the form (B3) cannot be rewritten further. For completeness, we reproduce the full definition of $\phi' \to_\mathcal{D} \phi''$ here:

- *Concrete domain implications*:
  - There exists an atom $\Pi(v_1, \ldots, v_m)$ of the form (B2) or (B1) in $\phi'$
  - that is implied by a $\mathcal{D}$-conjunction $\psi$ such that
    - for each object variable $x$ in $\phi'$, $\psi$ may contain atoms of type (B1) bound to $x$, possibly using fresh concrete domain variables (that will then be bound to $x$);
    - if $\Pi(v_1, \ldots, v_m)$ is of the form (B2), but not of the form (B1), then $\psi$ may additionally contain atoms of type (B3) over the variables $\mathsf{N_{CV}} \cap \{v_1, \ldots, v_m\}$;

    and
  - $\phi''$ is obtained from $\phi'$ by replacing $\Pi(v_1, \ldots, v_m)$ with $\psi$, and adding new attribute atoms $U(x, v)$ for the fresh variables $v$ in $\psi$ that must be bound to $x$ (where $U$ must occur in $\mathcal{T}$).

Recall that we allow the CQ $\phi''$ to violate condition (B1), as long as it can be restored by an immediate application of a substitution (see the definition of $\mathsf{infer}_\mathcal{T}$).

Observe that unbound variables $v$ can be assumed to occur at most twice in a bounded, safe CQ: once in an atom $=_d(v)$ and once somewhere else. If this is not the case, we can split $v$ into two or more unbound variables. Moreover, we assume that unbound variables always co-occur with at least one bound variable in a value comparison atom. If this does not hold, then consider the conjunction of all value comparison atoms that contain only unbound variables (except those of the form $=_d(v)$), which by our assumption cannot occur in other atoms of the CQ. Since the

CQ is safe, all these variables must satisfy condition b) of Definition 4.3, i.e., they are restricted to a constant. Hence, this conjunction can have at most one solution. Since $\mathcal{D}$ is cr-admissible, we can decide whether it is satisfiable at all. If it is satisfiable, then we can safely remove all these atoms from the CQ; otherwise, the query cannot have any answers w.r.t. a consistent KB. We assume in the following that all CQs resulting from each rewriting step are preprocessed in this way, i.e., either they are removed from the rewriting, or the unbound atoms are removed (and possibly distinguished concrete domain variables replaced by the associated constants).

This concludes the description of the rewriting $\Phi_{\mathcal{T}}$.

## 6.3 Correctness of the Rewriting

As a first step, we need to show that the rewriting does not violate our assumptions on the safety and boundedness of the CQs.

**Lemma 6.2.** *The initial CQ $\phi$ is bounded. Furthermore, if $\phi$ is safe, then every CQ $\phi'$ in $\Phi_{\mathcal{T}}$ is also safe and bounded.*

*Proof.* The initial CQ is trivially bounded by condition (B2). Further note that reduce does not affect the safety conditions and cannot lead to a violation of the boundedness conditions. Moreover, split preserves safety by duplicating atoms of the form $=_d(v)$, and can explicitly not affect boundedness.

For step, we explicitly restrict to bounded CQs. For safety, observe that the application of attribute range restrictions can only remove value comparison atoms, and inclusions explicitly introduce only safe concrete domain variables. Regarding implications in the concrete domain, observe that all new variables are immediately bound to some object variable $x$ that was already present in the original CQ. □

From this, we obtain the following important fact.

**Lemma 6.3.** *The set $\Phi_{\mathcal{T}}$ is finite.*

*Proof.* We first analyze the number of variables that occur in an element $\phi'$ of $\Phi_{\mathcal{T}}$. For each object term $x$ in $\phi'$, the number of fresh concrete domain variables bound to $x$ (that were not already present in $\phi$) is bounded by $n_{\mathcal{T}}$ due to (B1). Although we do not count variables introduced by split, this operator can at most introduce one new variable for each occurrence of a variable in a constraint set of $\mathbf{\Gamma}_{\mathcal{T}}$, the number of which is also bounded by a function of the size of $\mathcal{T}$.

Hence, it remains to consider the number of possible object variables. Note that a fresh object variable $y$ can only be introduced by an inclusion with an existential restrictions $\exists R$ on the left-hand side, which must be applied to some existing object variable $x$. Moreover, $y$ can only be used further in the rewriting by applying an inclusion with $\exists R^-$ on the right-hand side to the atom $R(x, y)$ introduced by the previous rule (or similarly with a subrole $S$ of $R$), which means that the variable $x$ must be nondistinguished and not occur elsewhere in the CQ. But then applying this inclusion will remove $x$. In essence, we can only replace $x$ by $y$. This means that the number of object terms in any element of $\Phi_{\mathcal{T}}$ can be at most $2 \cdot |\mathsf{N_{OV}}(\phi)| + |\mathsf{N_I}(\phi)|$.

In total, the number of terms is bounded by a function in the sizes of $\mathcal{T}$ and $\phi$, and hence the number of sets of atoms over these terms using the concept, role, and attribute names of $\mathcal{T}$, as well as the concrete domain predicates from $\mathfrak{R}_{\mathcal{T}} \cup \mathfrak{R}_{\mathcal{T},2}$, is finite. □

We now take one step closer to our main rewritability result, which began by constructing the canonical model (see Lemma 5.9). We reduce answering a query over $\mathcal{I}_{\mathcal{K}}$ to answering its rewriting over the ABox completion $\mathcal{I}^*_{\mathcal{A}}$. Afterwards, the last step will be to construct a

finite interpretation from the abstract interpretation $\mathcal{I}^*_{\mathcal{A}}$, obtaining a combined rewriting as in Definition 4.2.

Since our rewriting directly deals with implications over $\mathcal{D}$, the proof of the following lemma holds different challenges than similar ones in [10, 38]. In particular, it is important to separate variables that occur together in query atoms $\Pi(v_1, \dots, v_m)$, but are bound to different elements of the domain of the canonical model. Due to the functionality of $\mathcal{D}$, such atoms can only be implied by the TBox if each variable is mapped to a fixed value. Hence, $\Pi(v_1, \dots, v_m)$ can be replaced by a set of atoms of the form (B3).

**Lemma 6.4.** *If $\mathcal{K}$ is consistent, then we have*

$$\mathsf{ans}_{\mathsf{a}}(\phi, \mathcal{I}_{\mathcal{K}}) = \bigcup_{\phi' \in \Phi_{\mathcal{T}}} \mathsf{ans}_{\mathsf{a}}(\phi', \mathcal{I}^*_{\mathcal{A}}).$$

*Proof.* For the $\supseteq$-direction, let $\mathfrak{a}$ be a potential answer with $\mathcal{I}^*_{\mathcal{A}} \models_{\mathsf{a}} \mathfrak{a}(\phi')$ for some element $\phi' \in \Phi_{\mathcal{T}}$. Since $\mathcal{I}_{\mathcal{K}}$ is an extension of $\mathcal{I}^*_{\mathcal{A}}$, this implies that $\mathcal{I}_{\mathcal{K}} \models_{\mathsf{a}} \mathfrak{a}(\phi')$. Moreover, we know that $\phi'$ is the result of a finite number of applications of the operators $\mathsf{reduce}$, $\mathsf{split}$, and $\mathsf{infer}$ to $\phi$. Hence, it suffices to show that for each of these operations every answer tuple of the result in $\mathcal{I}_{\mathcal{K}}$ is also an answer tuple of the original query in $\mathcal{I}_{\mathcal{K}}$.

Consider first an application of $\mathsf{reduce}$, in which a substitution was applied to an intermediate CQ $\phi''$, resulting in the CQ $\sigma(\phi'')$. Clearly, a homomorphism $\pi$ of $\mathfrak{a}(\sigma(\phi''))$ into $\mathcal{I}_{\mathcal{K}}$ yields a potential answer $\mathfrak{a}'$ and a homomorphism $\pi'$ of $\mathfrak{a}'(\phi'')$ into $\mathcal{I}_{\mathcal{K}}$ by setting $\mathfrak{a}'(x) \coloneqq \mathfrak{a}(\sigma(x))$ for all answer variables $x \in \mathsf{FVar}(\phi'')$, $\pi'(x) \coloneqq \pi(\sigma(x))$ for all $x \in \mathsf{terms}(\mathfrak{a}'(\phi'')) \setminus \mathsf{FVar}(\sigma(\phi''))$, and $\pi'(x) \coloneqq \mathfrak{a}(\sigma(x))$ for all $x \in \mathsf{terms}(\mathfrak{a}'(\phi'')) \cap \mathsf{FVar}(\sigma(\phi''))$. Recall that individual names and concrete domain elements are not affected by $\sigma$, and that variables must be replaced by terms of the same type. Moreover, we obtain the same answer tuple due to our definition of $\mathfrak{a}'$.

Assume now that $=_d(v)$ occurs in $\phi''$ and we applied $\mathsf{split}$ to obtain $\phi'''$, where another occurrence of $v$ is replaced by a fresh nondistinguished variable $v'$ and a new atom $=_d(v')$ is added. Let $\pi$ be a homomorphism of $\mathfrak{a}(\phi''')$ into $\mathcal{I}_{\mathcal{K}}$. Hence, we have $\pi(\mathfrak{a}(v)) = \pi(\mathfrak{a}(v')) = d$, which means that we can merge $v'$ into $v$ without changing the satisfaction of the query. This means that $\pi$ is also a homomorphism of $\mathfrak{a}(\phi'')$ into $\mathcal{I}_{\mathcal{K}}$.

Finally, consider the case of a CQ $\phi''$ that was used to obtain $\phi'''$ with $\phi'' \to_{\mathcal{T}} \phi'''$ via the operator $\mathsf{infer}_{\mathcal{T}}$. We consider the cases of the definition of $\to_{\mathcal{T}}$:

- Assume that there exists an inclusion $X_1 \sqsubseteq X_2 \in \mathcal{T}$ such that (part of) $X_2(\vec{x}) \in \phi''$ for appropriate terms $\vec{x}$, was replaced by $X_1(\vec{x})$ in order to obtain $\phi'''$. If an additional substitution was applied to satisfy (B1), this can be shown correct using the same arguments as for $\mathsf{reduce}$ above. Again, let $\pi$ be a homomorphism of $\mathfrak{a}(\phi''')$ into $\mathcal{I}_{\mathcal{K}}$. A homomorphism $\pi'$ of $\mathfrak{a}(\phi'')$ into $\mathcal{I}_{\mathcal{K}}$ can be constructed using the completion rules **(CR1)**–**(CR5)**. We consider here only the case of an inclusion of the form $\exists U_1, \dots, U_m.\Pi \sqsubseteq \exists U'_1, \dots, U'_k.\Pi'$; the other kinds of inclusions can be treated using similar, but simpler, arguments. By assumption, $\phi''$ contains a subset of $\{U'_1(x, v'_1), \dots, U'_k(x, v'_k), \Pi'(v'_1, \dots, v'_k)\}$, where the variables $v'_i$ do not occur in any other atoms of $\phi'''$. In $\phi'''$, these atoms were replaced by $\Pi(v_1, \dots, v_m)$ and $U_i(x, v_i)$, $1 \le i \le m$, where all concrete domain variables are fresh.

  Hence, $\mathsf{terms}(\mathfrak{a}(\phi''))$ and $\mathsf{terms}(\mathfrak{a}(\phi'''))$ differ only in the terms $v'_1, \dots, v'_k$ and $v_1, \dots, v_m$. Moreover, we know that $(\pi(\mathfrak{a}(x)), w_o) \in U_i^{\mathcal{I}_{\mathcal{K}}}$, $1 \le i \le m$, and the atoms $=(w_i, \pi(v_i))$ and $\Pi(\pi(v_1), \dots, \pi(v_m))$ are implied by the relevant constraint sets $\Gamma_e$. Since the constraint sets do not share variables, we can infer that $\Gamma_{\pi(\mathfrak{a}(x))}$ implies $\Pi(w_1, \dots, w_m)$. Thus, we obtain $\pi(\mathfrak{a}(x)) \in (\exists U_1, \dots, U_m.\Pi)^{\mathcal{I}^{(\ell)}_{\mathcal{K}}}$ for some finite $\ell \ge 0$. By **(CR4)**, there must exist terms $v''_i$ such that $(\pi(\mathfrak{a}(x)), v''_i) \in (U'_i)^{\mathcal{I}_{\mathcal{K}}}$, $1 \le i \le k$, and $\Pi'(v''_1, \dots, v''_k)$ is implied by $\Gamma_{\pi(\mathfrak{a}(x))}$. Hence, by defining $\pi'(v'_i) \coloneqq v''_i$ for those of the variables $v'_i$, $1 \le i \le k$, that occur in $\psi''$, and $\pi'(z) \coloneqq \pi(z)$ for all $z \in \mathsf{terms}(\mathfrak{a}(\phi''')) \setminus \{v_1, \dots, v_m\}$, we can satisfy the

atoms that were replaced in $\phi''$; all other atoms remain satisfied since the variables among $v'_1, \dots, v'_k$ do not occur in them.

- Consider $B \sqsubseteq \forall U_1, \dots, U_m.\Pi \in \mathcal{T}$ such that $\Pi(v_1, \dots, v_m)$ occurs in $\phi''$, and in $\phi'''$ this atom was replaced by $B(x)$ and $U_i(x, v_i)$, $1 \leq i \leq m$. Let $\pi$ be a homomorphism of $\mathfrak{a}(\phi''')$ into $\mathcal{I}_\mathcal{K}$. Hence, we have
  - $\pi(\mathfrak{a}(x)) \in B^{\mathcal{I}_\mathcal{K}}$,
  - there exist terms $w_i$, $1 \leq i \leq m$, such that $(\pi(\mathfrak{a}(x)), w_i) \in U_i^{\mathcal{I}_\mathcal{K}}$ and $=(w_i, \pi(\mathfrak{a}(v_i)))$ is implied by $\mathcal{I}_\mathcal{K}$.

  By **(CR6)**, we obtain that $\mathcal{I}_\mathcal{K}$ implies $\Pi(w_1, \dots, w_m)$, and hence $\Pi(\pi(\mathfrak{a}(v_1)), \dots, \pi(\mathfrak{a}(v_m)))$, which shows that $\pi$ is also a homomorphism of $\mathfrak{a}(\phi'')$ into $\mathcal{I}_\mathcal{K}$.

Finally, for $\mathsf{infer}_\mathcal{D}$, consider an atom $\Pi(v_1, \dots, v_m)$ and a $\mathcal{D}$-conjunction $\psi$ using variables from $\{v_1, \dots, v_m\}$ and additional fresh variables, such that $\psi$ implies $\Pi(v_1, \dots, v_m)$, and the latter atom was replaced in $\phi''$ by the atoms of $\psi$ and additional attribute atoms to obtain $\phi'''$. Again, we assume that no additional substitution was applied in this process. Let $\pi$ be a homomorphism of $\mathfrak{a}(\phi''')$ into $\mathcal{I}_\mathcal{K}$, which means that $\pi(\mathfrak{a}(\psi))$ is implied by $\mathcal{I}_\mathcal{K}$. If a variable $v_i$, $1 \leq i \leq m$, does not occur in $\psi$, then its value is irrelevant for the implication, and moreover it must still occur in $\phi'''$ since $\phi''$ is safe. Hence, $\mathcal{I}_\mathcal{K}$ implies $\Pi(\pi(\mathfrak{a}(v_1)), \dots, \pi(\mathfrak{a}(v_m)))$, i.e., $\pi$ satisfies the atom $\Pi(\mathfrak{a}(v_1), \dots, \mathfrak{a}(v_m))$ (as well as all other atoms) in $\mathfrak{a}(\phi'')$.

For the $\subseteq$-direction, we assume that $\mathcal{I}_\mathcal{K}$ was constructed starting from $\mathcal{I}^*_\mathcal{A}$ (for ease of presentation, we set $\mathcal{I}^{(0)}_\mathcal{K} := \mathcal{I}^*_\mathcal{A}$). Let now $\mathfrak{a}$ be a potential answer such that $\mathcal{I}_\mathcal{K} \models_\mathsf{a} \mathfrak{a}(\phi)$. We show that there exists a $\phi' \in \Phi_\mathcal{T}$ with $\mathcal{I}^*_\mathcal{A} \models_\mathsf{a} \mathfrak{a}'(\phi')$ such that

- $\mathfrak{a}(\phi)$ and $\mathfrak{a}'(\phi')$ yield the same answer tuple, and
- all atoms $=_d(v)$ of type (B3) in $\phi'$ are satisfied since $v$ is mapped (by the homomorphism and $\mathfrak{a}'$) to a constant or to a term occurring in $\mathcal{I}^*_\mathcal{A}$, and hence by Lemma 5.5 they are satisfied already in $\mathcal{I}^*_\mathcal{A}$.

Consider the smallest index $\ell \geq 0$ for which there is a $\phi' \in \Phi_\mathcal{T}$, a potential answer $\mathfrak{a}'$ with the above properties, and a homomorphism of $\mathfrak{a}'(\phi')$ into $\mathcal{I}^{(\ell)}_\mathcal{K}$. Such an index must exist since $\mathcal{I}_\mathcal{K} \models_\mathsf{a} \mathfrak{a}(\phi)$, $\phi \in \Phi_\mathcal{T}$, $\phi$ does not contain atoms of type (B3), and due to the fairness requirement in the construction of $\mathcal{I}_\mathcal{K}$. If $\ell = 0$, then we have proven the claim; we now consider the case that $\ell > 0$ and show that it is impossible. Let $\phi' \in \Phi_\mathcal{T}$ and $\pi$ be a homomorphism of $\mathfrak{a}'(\phi')$ into $\mathcal{I}^{(\ell)}_\mathcal{K}$. Since $\ell$ is minimal, we know that the last completion rule applied to obtain $\mathcal{I}^{(\ell)}_\mathcal{K}$ from $\mathcal{I}^{(\ell-1)}_\mathcal{K}$ was necessary to satisfy an atom of $\mathfrak{a}'(\phi')$ under $\pi$. We make a case distinction on the type of the rule that was applied.

- If **(CR1)** was applied, then there is an inclusion $X_1 \sqsubseteq X_2 \in \mathcal{T}$ and an element $e \in X_1^{\mathcal{I}^{(\ell-1)}_\mathcal{K}}$ that was added to $X_2^{\mathcal{I}^{(\ell)}_\mathcal{K}}$.

  We consider first the case that $X_2$ is a basic concept or a role. Since we assumed that the rule application is necessary to satisfy $\mathfrak{a}'(\phi')$ via $\pi$, there must be at least one atom $X_2(\vec{x}) \in \phi'$ such that $\pi(\mathfrak{a}'(\vec{x})) = e$. Hence, we can try to replace each such atom $X_2(\vec{x})$ by $X_1(\vec{x})$, according to $\to_\mathcal{T}$. If this is possible, then the resulting CQ is also an element of $\Phi_\mathcal{T}$ and is satisfied by $\mathfrak{a}'$ and $\pi$ in $\mathcal{I}^{(\ell-1)}_\mathcal{K}$, which contradicts our assumption on the minimality of $\ell$. The only problematic case is when $X_1$ is of the form $\exists U_1, \dots, U_m.\Pi$, since then we introduce new concrete domain variables $v_1, \dots, v_m$ that are bound to $x$. Hence, we have to show that there is a substitution that can enforce the restriction in (B1). First, note that $\Pi$ is an element of $\mathfrak{R}_\mathcal{T}$. Furthermore, any concrete domain variable bound to $x$ in $\phi'$

must be mapped by $\pi$ and $\mathfrak{a}'$ to some term $w$ of $\Gamma_e$ (or a term that is equivalent to $w$ due to the constraint sets); otherwise, we must have $e \in \mathsf{N_I}$, and hence $X_2(x)$ is satisfied already in $\mathcal{I}^*_{\mathcal{A}}$, which contradicts our assumption that $\ell > 0$. But then all terms mapped to terms that are equivalent to the same term $w$ in $\Gamma_e$ can be unified, since afterwards they still satisfy all affected attribute or value comparison atoms since they are still mapped to terms that are equivalent to $w$. This brings the total number of variables bound to $x$ down to at most $n_{\mathcal{T}}$, and thereby satisfies (B1).

Finally, consider the case that $X_1, X_2 \in \mathsf{N_A}$. By assumption, the new tuple $(e, w) \in X_2^{\mathcal{I}_{\mathcal{K}}^{(\ell)}}$ was necessary to satisfy at least one attribute atom $X_2(x, v) \in \phi'$ under $\pi$, i.e., we have $\pi(\mathfrak{a}'(x)) = e$ and the relevant constraint sets $\Gamma_{e'}^{(\ell)}$ imply $=(w, \pi(\mathfrak{a}'(v)))$. Since $(e, w) \in X_1^{\mathcal{I}_{\mathcal{K}}^{(\ell-1)}}$ and $\Gamma_{e'}^{(\ell-1)} = \Gamma_{e'}^{(\ell)}$, we can apply $\to_{\mathcal{T}}$ to replace all such atoms $X_2(x, v)$ with $X_1(x, v)$ to obtain an element of $\Phi_{\mathcal{T}}$ that is satisfied by $\mathfrak{a}'$ and $\pi$ in $\mathcal{I}_{\mathcal{K}}^{(\ell-1)}$, which again yields a contradiction.

- If **(CR4)** was applied, then there is an inclusion $B \sqsubseteq \exists U_1, \dots, U_m.\Pi$ in $\mathcal{T}$, an element $e \in B^{\mathcal{I}_{\mathcal{K}}^{(\ell-1)}}$, and $v_1, \dots, v_m$ such that $(e, v_i)$ is added to $U_i^{\mathcal{I}_{\mathcal{K}}^{(\ell)}}$, $1 \leq i \leq m$, and $\Pi(v_1, \dots, v_m)$ is added to $\Gamma_e^{(\ell)}$. By our assumption, $\phi'$ must contain at least one atom of the form

  (a) $U_i(x, v_i')$ such that $\pi(\mathfrak{a}'(x)) = e$ and $=(\pi(\mathfrak{a}'(v_i')), v_i)$ is implied by $\mathcal{I}_{\mathcal{K}}^{(\ell)}$; or

  (b) $\Pi'(v_1', \dots, v_k')$ such that $\Pi'(\pi(\mathfrak{a}'(v_1')), \dots, \pi(\mathfrak{a}'(v_k')))$ is implied by $\mathcal{I}_{\mathcal{K}}^{(\ell)}$.

  Before we can apply the inclusion $B \sqsubseteq \exists U_1, \dots, U_m.\Pi$ (via $\to_{\mathcal{T}}$) to $\phi'$, we first need to rewrite it as follows:

  1. Using $\mathsf{infer}_{\mathcal{D}}$, we rewrite the atoms of type (b), with the goal of putting atoms mapped by $\pi$ and $\mathfrak{a}'$ to $\Pi(v_1, \dots, v_m)$ into the rewriting. These will be the only remaining value comparison atoms that are not already satisfied in $\mathcal{I}_{\mathcal{K}}^{(\ell-1)}$.
  2. We unify all atoms of the above form using $\mathsf{reduce}$.
  3. We also unify all atoms of the form (a) above (depending on the attribute involved).
  4. We can apply the inclusion $B \sqsubseteq \exists U_1, \dots, U_m.\Pi$ to eliminate these atoms, obtaining an element of $\Phi_{\mathcal{T}}$ that is satisfied in $\mathcal{I}_{\mathcal{K}}^{(\ell)}$, which again contradicts our minimality assumption on $\ell$.

  For step 1, observe that atoms of the form (b) cannot be of type (B3), since then they must be of the form $=_d(v_1')$ and $\pi$ and $\mathfrak{a}'$ satisfies this atom already in $\mathcal{I}^*_{\mathcal{A}}$, contrary to our assumption that the completion rule was necessary to satisfy it.

  Hence, any atom $\Pi'(v_1', \dots, v_k')$ from (b) must be of the form (B2) or (B1). Let now $\Gamma$ be a minimal subset of the union of all sets $\Gamma_{e'}^{(\ell)}$ that implies $\Pi'(\pi(\mathfrak{a}'(v_1')), \dots, \pi(\mathfrak{a}'(v_k')))$. We need to consider only those sets containing the variables among $\pi(\mathfrak{a}'(v_1')), \dots, \pi(\mathfrak{a}'(v_k'))$, and hence $\Gamma$ is finite. We consider the set $\pi^{-1}(\Gamma)$ that is obtained from $\Gamma$ by replacing every variable of the form $\pi(\mathfrak{a}'(v_i'))$, $1 \leq i \leq k$, by $v_i'$. Then we have that $\Gamma' \coloneqq \pi^{-1}(\Gamma) \cup \{=_d(v_i') \mid 1 \leq i \leq k,\ \pi(\mathfrak{a}'(v_i')) = d \in \Delta^{\mathcal{D}}\}$ implies $\Pi'(v_1', \dots, v_k')$.

  Since the goal is to replace $\Pi'(v_1,' \dots, v_k')$ by $\Gamma'$, we need to ensure that it satisfies the boundedness conditions and the definition of $\to_.$ First, we ensure that all object variables that $v_1', \dots, v_k'$ are bound to are mapped to distinct domain elements of $\mathcal{I}_{\mathcal{K}}^{(\ell)}$; a violation of this property is easily repaired using a substitution, which can be applied after the rewriting step with $\to_.$

  We now ensure that $\Gamma'$ contains *ABox variables* $v_i'$, i.e., variables that are bound to object variables which are mapped to named individuals, only in atoms of the form $=_d(v_i')$ with $=_d \in \mathfrak{R}_{\mathcal{T},2}$. Observe that not all terms $v_i'$ can be ABox variables or be mapped to constants

by $\pi$ and $\mathfrak{a}'$, since otherwise $\Pi'(v'_1, \ldots, v'_k)$ would be satisfied already in $\mathcal{I}^*_{\mathcal{A}}$. Since we assumed that all unbound variables are mapped to constants by $\pi$ and $\mathfrak{a}'$, there must be at least one other variable $v'_{i'}$ that is bound to an object variable that is not mapped to a named individual. This can only be the case if $\Pi'(v'_1, \ldots, v'_k)$ is of type (B2) since $\phi'$ is bounded. Since $\Gamma'$ is satisfiable and contains at least two connected components and $\mathcal{D}$ is functional, each connected component involving $v'_{i'}$ must be equivalent to an atom of the form $=_{d'}(v'_{i'})$. Since this connected component is derived via $\pi^{-1}$ from (a subset of) the constraint set involving $\pi(v'_{i'})$, we know that $=_{d'} \in \mathfrak{R}_{\mathcal{T},1}$. Again by functionality of $\mathcal{D}$, we know that $=_{d'}(v'_{i'})$ and $\Pi'(v'_1, \ldots, v'_k)$ imply some atom of the form $=_d(v'_i)$. Hence, we have that $=_d \in \mathfrak{R}_{\mathcal{T},2}$. If the atom $=_d(v'_i)$ is already contained in $\Gamma'$ and $v'_i$ does not occur elsewhere in $\Gamma'$, we are finished. Otherwise, this atom must also be equivalent to the constraint set involving $\pi(\mathfrak{a}(v'_i))$, and hence we can replace the corresponding part of $\pi^{-1}(\Gamma)$ by $=_d(v'_i)$, which achieves our goal. Furthermore, since $\Pi'(v'_1, \ldots, v'_k)$ is of type (B2), we are allowed to use the atom $=_d(v'_i)$ for rewriting $\Pi'(v'_1, \ldots, v'_k)$.

The remaining elements of $\pi^{-1}(\Gamma)$ fall into the category (B1) since they are obtained from constraint sets over $\mathcal{T}$, they must involve the variables among $v'_1, \ldots, v'_k$, variables bound to different object variables must belong to different constraint sets, and any two constraint sets do not share variables.

We now show that the remaining atoms of the form $=_d(v'_i)$ with $\pi(\mathfrak{a}'(v'_i)) = d \in \Delta^{\mathcal{D}}$, where $v'_i$ is not an ABox variable, also comply with the definition of $\mathsf{infer}$. Since $v'_i$ occurs in $\phi'$, which is safe by Lemma 6.2, we must have one of the following cases:

- If an atom $=_{d'}(v'_i)$ occurs in $\phi'$, then we have $d = d'$ since this atom is satisfied by $\pi$ and $\mathfrak{a}'$. But then including this atom in $\Gamma'$ is safe since we only reintroduce an atom to $\phi'$ that is already present. This atom cannot actually be equal to $\Pi'(v'_1, \ldots, v'_k)$ since then it would already be satisfied in $\mathcal{I}^{(\ell-1)}_{\mathcal{K}}$.
- Otherwise, there is an atom $U(x, v'_i)$ in $\phi'$, i.e., we have $(\pi(\mathfrak{a}'(x)), w) \in U^{\mathcal{I}^{(\ell)}_{\mathcal{K}}}$ such that $=(w, d)$ is implied by $\Gamma^{(\ell)}_{\pi(\mathfrak{a}'(x))}$. This means that $=_d(v'_i)$ is implied by this constraint set when we replace $w$ by $v'_i$. Since $\pi(\mathfrak{a}'(x)) \notin \mathsf{N_I}$, we can replace $=_d(v'_i)$ in $\Gamma'$ by this modified constraint set, and maintain all properties above, i.e., that $\Gamma'$ implies $\Pi'(v'_1, \ldots, v'_k)$ and that all its atoms are satisfied by $\pi$ and $\mathfrak{a}'$ in $\mathcal{I}^{(\ell)}_{\mathcal{K}}$.

Furthermore, $\Gamma'$ uses only variables that already occur in $\phi'$ and fresh ones from $\mathsf{Var}(\mathcal{I}^{(\ell-1)}_{\mathcal{K}})$, for which we assume without loss of generality that they do not occur in $\phi'$. Finally, $\Gamma'$ can only use constants from $\Delta^{\mathcal{D}}(\mathcal{T})$ since we have already eliminated all constraint sets originating from named individuals above.

After these preparations, we can finally replace $\Pi'(v'_1, \ldots, v'_k)$ in $\phi'$ according to $\to_{\mathcal{D}}$ by the atoms of $\Gamma'$, where for each fresh variable $v$ in this set we know that it must occur in some $(\pi(\mathfrak{a}'(x)), v) \in U^{\mathcal{I}^{(\ell)}_{\mathcal{K}}}$, where $x$ already occurs in $\phi'$, and hence we can add the required atom $U(x, v)$ to the query. These atoms can be satisfied by mapping $v$ to itself, and the atoms in $\Gamma'$ are then also satisfied since they directly occur in the relevant sets $\Gamma^{(\ell)}_{e'}$ or are already satisfied by $\mathfrak{a}'$ and $\pi$. Moreover, the only new attribute atoms that are not already satisfied in $\mathcal{I}^{(\ell-1)}_{\mathcal{K}}$ are new ones of the form (a) above, which involve the fresh variables $v_i$ introduced by the completion rule. To satisfy (B1), i.e., bound the number of concrete domain variables, we may need to apply a substitution again, which can be obtained in the same way as in the case of **(CR1)** above.

This finishes the description of step 1. Via a series of rewriting operations, we have now obtained another query $\phi''$ from $\Phi_{\mathcal{T}}$ and a homomorphism $\pi'$ of $\mathfrak{a}'(\phi'')$ into $\mathcal{I}^{(\ell)}_{\mathcal{K}}$. The only atoms in $\phi''$ that are not already satisfied by $\mathfrak{a}'$ and $\pi'$ in $\mathcal{I}^{(\ell-1)}_{\mathcal{K}}$ are those of the form $\pi^{-1}(\Pi(v_1, \ldots, v_m))$ introduced by the above replacements, and the ones of the form (a) (possibly more than in $\phi'$). However, each of the atoms of the first kind is mapped by $\mathfrak{a}'$

and $\pi'$ to the same atom, namely $\Pi(v_1, \ldots, v_m)$, and hence for step 2 we can apply a substitution $\sigma$ that unifies these atoms.

For step 3, observe that similarly each atom $U_i(x, v_i')$ of type (a) is mapped by $\mathfrak{a}'$ and $\pi'$ to some $U_i(e, w_i)$ for which $=(v_i, w_i)$ is implied by $\mathcal{I}_{\mathcal{K}}^{(\ell)}$, and hence we can similarly ensure by an application of reduce that there is at most one such atom for each fresh variable $v_i$, and moreover that they use the same object variable $x$. Furthermore, $v_i'$ can be merged with the $i$-th term of the single remaining atom of the form $\pi^{-1}(\Pi(v_1, \ldots, v_m))$.

If the resulting merged atoms $U_i(x, v_i')$ and $\Pi(v_1', \ldots, v_m')$ are such that some $v_i'$, $1 \leq i \leq m$, that occurs in $U_i(x, v_i')$ is a constant or a distinguished variable, then we cannot directly apply the inclusion to rewrite these atoms. However, in such a case we know that $w_i = \mathfrak{a}'(v_i') \in \Delta^{\mathcal{D}}$ and $=_{w_i}(v_i)$ is implied by the constraint sets in $\mathcal{I}_{\mathcal{K}}^{(\ell)}$. Since $v_i$ is a fresh variable, this atom is already implied by $\Pi(v_1, \ldots, v_m)$, and hence in particular we have $=_{w_i} \in \mathfrak{R}_{\mathcal{T},1}$. We can now eliminate the occurrence of $v_i'$ in $U_i(x, v_i')$ by first introducing a new atom $=_{w_i}(v_i')$ of type (B3) (since it implies $\top_{\mathcal{D}}(v_i')$), and then splitting the term in order to obtain the atoms $\Pi(v_1', \ldots, v_i'', \ldots, v_m')$, $U_i(x, v_i'')$, $=_{w_i}(v_i')$, and $=_{w_i}(v_i'')$, where $v_i''$ is a fresh nondistinguished variable. The third atom is obviously already satisfied by $\mathfrak{a}'$, and $\pi'$ can easily be extended to $v_i''$ by setting $\pi'(v_i'') \coloneqq \pi'(\mathfrak{a}'(v_i')) = w_i$. We can then eliminate the fourth atom by replacing it with $\Pi(v_1'', \ldots, v_m'')$, using some fresh nondistinguished variables and attribute atoms $U_i(x, v_{i'}'')$. These atoms can be merged with $\Pi(v_1', \ldots, v_o'', \ldots, v_m')$ and $U_i(x, v_{i'}')$ as described above.

After step 3, all atoms $U_i(x, v_i')$ of type (a) must be such that $v_i'$ is nondistinguished, and furthermore equal to the $i$-th term of the single remaining atom of the form $\Pi(v_1', \ldots, v_m')$. All these operations can be done using split and reduce, and hence we stay inside $\Phi_{\mathcal{T}}$. It is straightforward to adapt the potential answer $\mathfrak{a}'$ of $\phi''$ into a new potential answer $\mathfrak{a}''$ of the resulting query that yields the same answer tuple (we may have merged some distinguished variables).

The result is yet another element $\phi'''$ of $\Phi_{\mathcal{T}}$ that is satisfied in $\mathcal{I}_{\mathcal{K}}^{(\ell)}$ via $\mathfrak{a}''$ and $\pi'$, and the only atoms that are not already satisfied $\mathcal{I}_{\mathcal{K}}^{(\ell-1)}$ form a subset of $(\exists U_1, \ldots, U_m.\Pi)(x)$ (we have eliminated all constants and distinguished variables $v_i'$ above). No other terms can be mapped to the new variables $v_i$ introduced by the completion rule, and hence the concrete domain variables in these atoms do not occur elsewhere in $\phi'''$. If none of the relevant attribute atoms $U_i(x, v_i')$ present, then we can simply remove $\Pi(v_1', \ldots, v_m')$ and the associated atoms of the form $=_d(v_i')$ from the CQ since $\pi'$ and $\mathfrak{a}''$ satisfy them, and hence they are valid in $\mathcal{D}$. Otherwise, we finally come to step 4 and can apply the operator $\mathsf{infer}_{\mathcal{T}}$ in order to obtain another element of $\Phi_{\mathcal{T}}$ that is satisfied in $\mathcal{I}_{\mathcal{K}}^{(\ell-1)}$ via $\mathfrak{a}''$ and $\pi'$, which yields a contradiction.

- If **(CR6)** was applied, then there is an attribute range constraint $B \sqsubseteq \forall U_1, \ldots, U_m.\Pi \in \mathcal{T}$ and an element $e \in B^{\mathcal{I}_{\mathcal{K}}^{(\ell-1)}}$ with $(e, v_i) \in U_i^{\mathcal{I}_{\mathcal{K}}^{(\ell-1)}}$, $1 \leq i \leq m$, and $\Pi(v_1, \ldots, v_m)$ was added to $\Gamma_e^{(\ell-1)}$ in order to obtain $\Gamma_e^{(\ell)}$. By our assumption, this atom is necessary to imply one or more atoms of the form $\Pi'(\pi(\mathfrak{a}'(v_1')), \ldots, \pi(\mathfrak{a}'(v_k')))$, where $\Pi'(v_1', \ldots, v_k')$ is a value comparison atom of $\phi'$. These atoms can be replaced using the same technique as for the rule **(CR4)**. Since **(CR6)** does not change the interpretation of the attribute names, the only atoms in the resulting CQ $\phi'' \in \Phi_{\mathcal{T}}$ that are not already satisfied by the obtained potential answer $\mathfrak{a}''$ and homomorphism $\pi'$ in $\mathcal{I}_{\mathcal{K}}^{(\ell)}$ are of the form $\pi^{-1}(\Pi(v_1, \ldots, v_m))$. If such an atom does not contain bound variables, then it can simply be removed. Otherwise, it is mapped by $\pi'$ and $\mathfrak{a}''$ to the atom $\Pi(v_1, \ldots, v_m)$ in $\Gamma_e^{(\ell)}$, which means that at least one of the involved variables is bound to an object variable $x$ that is mapped to $e$. Hence, for each such atom we can apply a rewriting step w.r.t. the attribute range constraint $B \sqsubseteq \forall U_1, \ldots, U_m.\Pi$, thereby replacing $\pi^{-1}(\Pi(v_1, \ldots, v_m))$ with $B(x)$ and all associated atoms $U_i(x, \pi^{-1}(v_i))$, which are all satisfied by $\pi'$ since $x$ is mapped to $e$ and each term

$\pi^{-1}(v_i)$ is mapped to a term equivalent to $v_i$. This results in an element of $\Phi_{\mathcal{T}}$ that is satisfied in $\mathcal{I}_{\mathcal{K}}^{(\ell-1)}$, which again yields a contradiction.

The remaining rules can be handled by similar, but simpler, arguments. □

For other *DL-Lite* formalisms, usually at this point combined rewritability has been shown and the CQs in $\Phi_{\mathcal{T}}$ can simply be evaluated over a finite relational database [10, 35]. However, in our setting we need to take another step.

## 6.4 Evaluating the Rewriting

In order to obtain a combined rewriting as described in Definition 4.2, the last step is to replace the abstract interpretation $\mathcal{I}_{\mathcal{A}}^*$ by an ordinary finite interpretation, i.e., a database. We will use the canonical solution $f_{\mathcal{K}}$ from Section 5.4 to instantiate the variables in $\mathcal{I}_{\mathcal{A}}^*$. We assume without loss of generality that the individual names in $\mathsf{N_I}(\mathcal{K})$ are the first elements occurring in the enumeration of $\Delta^{\mathcal{I}_{\mathcal{K}}}$ that was used in the construction of $f_{\mathcal{K}}$. Since $\mathcal{I}_{\mathcal{A}}^*$ is equal to $\mathcal{I}_{\mathcal{K}}$ on $\mathsf{N_I}(\mathcal{K})$, this means that we only have to compute the partial solutions $f_{\mathcal{K}}^{(1)}, \ldots, f_{\mathcal{K}}^{(|\mathsf{N_I}(\mathcal{K})|)}$ in order to obtain $f_{\mathcal{K}}(\mathcal{I}_{\mathcal{A}}^*)$. For each $f_{\mathcal{K}}^{(j)}$, we have to find a variable assignment in $\mathsf{sol}(\bigwedge \mathsf{Pos}_j) \setminus \mathsf{sol}(\bigvee \mathsf{Neg}_j^-)$, which is possible in polynomial time since $\mathcal{D}$ is polynomial and constructive. Observe that $\mathsf{Pos}_j$ and $\mathsf{Neg}_j^-$ are of size polynomial in the number of value terms from $\mathcal{I}_{\mathcal{A}}^*$ and $\Delta^{\mathcal{D}}(\mathcal{K})$ and exponential in the maximum arity of the involved predicates. The latter is not problematic since all predicates in $\mathfrak{R} = \mathfrak{R}_{\mathcal{K}} \cup \mathfrak{R}_{\mathcal{K},2} \cup \{=\}$ whose arity is larger than 2 must occur in the (fixed) TBox $\mathcal{T}$. This shows that we can construct $f_{\mathcal{K}}(\mathcal{I}_{\mathcal{A}}^*)$ in polynomial time in the size of $\mathcal{A}$. Moreover, $f_{\mathcal{K}}$ actually is a solution of $\mathcal{I}_{\mathcal{A}}^*$ since $\mathcal{K}$ is consistent (see Lemmas 5.6 and 5.8). The correctness of this approach can be shown by similar arguments as in the proofs of Lemmas 5.8 and 5.9.

**Lemma 6.5.** *If $\mathcal{K}$ is consistent, then we have $\mathsf{ans_a}(\phi', \mathcal{I}_{\mathcal{A}}^*) = \mathsf{ans}(\phi', f_{\mathcal{K}}(\mathcal{I}_{\mathcal{A}}^*))$ for all $\phi' \in \Phi_{\mathcal{T}}$.*

*Proof.* Given a homomorphism of $\phi'$ into $f_{\mathcal{K}}(\mathcal{I}_{\mathcal{A}}^*)$, we can construct a homomorphism of $\phi'$ into $\mathcal{I}_{\mathcal{A}}^*$ exactly as in the proof of Lemma 5.9. Conversely, let $\mathfrak{a} \in \mathsf{ans_a}(\phi', \mathcal{I}_{\mathcal{A}}^*)$ and $\pi$ be an abstract homomorphism of $\phi'$ into $\mathcal{I}_{\mathcal{A}}^*$. We construct a homomorphism $\pi'$ of $\phi'$ into $f_{\mathcal{K}}(\mathcal{I}_{\mathcal{A}}^*)$ by setting $\pi'(x) \coloneqq \pi(x)$ for all object terms $x$ and $\pi'(v) \coloneqq f_{\mathcal{K}}(\pi(v))$ for all value terms $v$ in $\phi'$. All atoms involving only object variables are obviously still satisfied. For an attribute atom $U(x, v) \in \phi'$, there must be a $w \in \mathsf{terms}(\mathcal{I}_{\mathcal{A}}^*)$ such that $(\pi(x), w) \in U^{\mathcal{I}_{\mathcal{A}}^*}$ and $=(w, \pi(v))$ is implied by $\mathcal{I}_{\mathcal{A}}^*$. Since $\pi(v)$ is either a constant or a variable from $\mathcal{I}_{\mathcal{A}}^*$ and $f_{\mathcal{K}}$ satisfies all constraint sets of $\mathcal{I}_{\mathcal{A}}^*$, we know that $f_{\mathcal{K}}(w) = f_{\mathcal{K}}(\pi(v)) = \pi'(v)$, and hence $(\pi'(x), \pi'(v) = (\pi(x), f_{\mathcal{K}}(w)) \in U^{f_{\mathcal{K}}(\mathcal{I}_{\mathcal{A}}^*)}$, i.e., the atom is also satisfied by $\pi'$. Consider now a value comparison atom $\Pi(v_1, \ldots, v_m) \in \phi$, which must be satisfied by $\pi$, i.e., $\Pi(\pi(v_1), \ldots, \pi(v_m))$ is implied by $\mathcal{I}_{\mathcal{A}}^*$. Again, since $f_{\mathcal{K}}$ satisfies all constraint sets in $\mathcal{I}_{\mathcal{A}}^*$, we know that the atom remains satisfied. □

Hence, we obtain our main result.

**Theorem 6.6.** *If $\mathcal{D}$ is cr-admissible, then safe and $\mathcal{T}$-restricted CQs are combined rewritable w.r.t. DL-Lite$_{core}^{(\mathcal{HF})}(\mathcal{D})$ TBoxes $\mathcal{T}$, and the rewritings are computable.*

*Proof.* From Lemmas 5.9, 6.4, and 6.5 we obtain the requirements of Definition 4.2 if we set $\mathcal{I}(\mathcal{K}) \coloneqq f_{\mathcal{K}}(\mathcal{I}_{\mathcal{A}}^*)$. □

This shows that the entailment problem for safe Boolean CQs in *DL-Lite*$_{core}^{(\mathcal{HF})}(\mathcal{D})$ is in P in data complexity. From a practical point of view, this result allows us to combine an off-line (polynomial) computation of the database $f_{\mathcal{K}}(\mathcal{I}_{\mathcal{A}}^*)$ with an on-line rewriting of incoming queries.

## 6.5 Checking Consistency

So far, we have ignored the omnipresent side condition that $\mathcal{K} = \langle \mathcal{A}, \mathcal{T} \rangle$ should be consistent, since query answering over an inconsistent KB is meaningless. However, by Lemma 5.8, this can be checked by testing whether (i) $\mathcal{I}_\mathcal{K}$ has a solution and (ii) $\mathcal{I}_\mathcal{K} \models_\mathsf{a} \mathcal{K}$ holds. For (i), observe that there can be only finitely many constraint sets $\Gamma_e$ in $\mathcal{I}_\mathcal{K}$ (modulo variable renaming), and hence we can use the following modified construction of $\mathcal{I}_\mathcal{K}$:

- After each application of a completion rule, we check whether all constraint sets are satisfiable, which is possible due to the cr-admissibility of $\mathcal{D}$.
- Each new domain element $e$ that is created in the construction initially satisfies exactly one basic concept of the form $\exists R^-$, and all other concepts, role successors, and attribute values must follow from this concept. This also means that the obtained constraint set $\Gamma_e$ will be isomorphic to the constraint set of any other such anonymous $R$-successor. Hence, it suffices to construct only one such element for each role $R$, and saturate it according to the completion rules (cf. Section 5.3), in order to verify the existence of a solution for $\mathcal{I}_\mathcal{K}$.

If this check is successful, it remains to verify (ii). The proof of Lemma 5.8 suggests the following standard procedure [10, 35] for checking this: If functionality constraints are violated by $\mathcal{I}_\mathcal{K}$, then this must be the case already in $\mathcal{I}^{(0)}_\mathcal{K}$, which can be checked in polynomial time. For the disjointness constraints, we restrict $\mathcal{K}$ to the KB $\mathcal{K}' \coloneqq \langle \mathcal{A}, \mathcal{T}' \rangle$, where $\mathcal{T}'$ is obtained from $\mathcal{T}$ by dropping all functionality and disjointness constraints. We know that $\mathcal{K}'$ is consistent since its canonical model is the same as that of $\mathcal{K}$, and hence has a solution, and the only other sources of inconsistencies have been removed (see the proof of Lemma 5.8). We then ask a set of Boolean CQs $\Psi$ over $\mathcal{K}'$: For each $\mathsf{disj}(X_1, X_2) \in \mathcal{T}$, we include the CQ $() \leftarrow X_1(\vec{x}) \wedge X_2(\vec{x})$ in $\Psi$, where $\vec{x}$ contains variables of the appropriate types. By Theorem 6.6, we can answer these CQs, and we know that $\mathcal{K}$ is consistent iff none of the CQs in $\Psi$ is entailed by $\mathcal{K}'$.

## 6.6 The Special Case of Unary Predicates

We consider again the special case of a unary, decidable concrete domain $\mathcal{D}$ satisfying *(infinitediff)*, as in [38]. We show that in this case the ABox completion is not necessary, i.e., we can extend the rewriting of 6.4 directly to the substructure $\mathcal{I}^{(0)}_\mathcal{K}$ of $\mathcal{I}^*_\mathcal{A}$, which does not contain any variables. Before we come to the proof, we make several critical observations (see also Section 2.2):

- By Lemma 2.7, we can add the equality predicates to $\mathcal{D}$ without affecting *(infinitediff)*. Moreover, since the binary equality predicate $=$ cannot occur in $\mathcal{T}$ or $\phi$, for $\rightarrow_\mathcal{D}$ it suffices to decide implications that do not contain $=$. It is easy to see that the unary predicates $=_d$ ($d \in \Delta^\mathcal{D}$) do not affect decidability.
- The rewriting $\Phi_\mathcal{T}$ cannot contain $=$, and any atom $=_d(v)$ can be eliminated by replacing $v$ with $d$. Hence, the rewritten CQs are formulated over $\mathcal{D}$.
- By Lemma 2.8, $\mathcal{D}$ is convex due to the predicates $=_d$. Moreover, $\mathcal{D}$ is trivially functional.
- The only places where polynomiality and constructivity of $\mathcal{D}$ are needed are in the constructions of $\mathfrak{R}_{\mathcal{T},2}$ (for condition (B3) of boundedness) and $f_\mathcal{K}$ (to obtain $f_\mathcal{K}(\mathcal{I}^*_\mathcal{A})$). We now define a weaker notion of boundedness without $\mathfrak{R}_{\mathcal{T},2}$ that suffices for unary concrete domains, and subsequently show that one can directly use $\mathcal{I}(\mathcal{A}) = \mathcal{I}^{(0)}_\mathcal{K}$ instead of $f_\mathcal{K}(\mathcal{I}^*_\mathcal{A})$. Hence, $\mathcal{D}$ does not need to be polynomial or constructive, and our results apply to any unary, decidable concrete domain $\mathcal{D}$ that satisfies *(infinitediff)*.

To get rid of $\mathfrak{R}_{\mathcal{T},2}$, observe that, since $\mathcal{D}$ is unary, all constraint sets can be partitioned into independent components for each involved concrete domain variable. We will consider each separately, and hence assume in the following that all $\Gamma \in \mathbf{\Gamma}_\mathcal{T}$ contain at most one variable. Note that all constraint sets $\Gamma_a$, $a \in \mathsf{N_I}$, are also of this form, i.e., for each variable or constant occurring in them they contain an element of $\mathbf{\Gamma}_\mathcal{T}$. We adapt the notion of boundedness and say that a CQ is *weakly bounded* if condition (B1) is modified such that each object variable $x$ may have an arbitrary number of bound concrete domain variables $v$, but the value comparison atoms involving $v$ must form an element of $\mathbf{\Gamma}_\mathcal{T}$. The overall size and number of CQs in the rewriting is still bounded since two nondistinguished concrete domain variables satisfying the same attribute and value comparison atoms can clearly be merged into one variable without affecting the semantics. Hence, Lemmas 6.2 and 6.3 still hold in this setting. As a consequence of these changes, case (B2) in the definition of boundedness is now subsumed by (B1). This also means that the rewriting cannot produce atoms of the form (B3), and hence the only remaining case is (B1). Another consequence of this is that we do not need to compute the set $\mathfrak{R}_{\mathcal{T},2}$.

We now provide the final missing piece for our arguments above, by showing that the rewriting can be directly evaluated over $\mathcal{I}_\mathcal{K}^{(0)}$ instead of $f_\mathcal{K}(\mathcal{I}_\mathcal{A}^*)$ (cf. Lemmas 6.4 and 6.5).

**Lemma 6.7.** *If $\mathcal{D}$ is unary, decidable, and satisfies (infinitediff), and $\mathcal{K}$ is consistent, then we have*

$$\mathsf{ans}_\mathsf{a}(\phi, \mathcal{I}_\mathcal{K}) = \bigcup_{\phi' \in \Phi_\mathcal{T}} \mathsf{ans}(\phi', \mathcal{I}_\mathcal{K}^{(0)}).$$

*Proof.* Since $\mathcal{I}_\mathcal{K}^{(0)}$ is also a substructure of $\mathcal{I}_\mathcal{K}$, soundness of the rewrite rules can be shown as in Lemma 6.4. For the other direction, we already know from Lemma 6.4 that for each $\mathfrak{a}$ with $\mathcal{I}_\mathcal{K} \models_\mathsf{a} \mathfrak{a}(\phi)$ there exists a $\phi' \in \Phi_\mathcal{T}$ such that $\mathcal{I}_\mathcal{A}^* \models_\mathsf{a} \mathfrak{a}'(\phi')$ and $\mathfrak{a}(\phi)$ and $\mathfrak{a}'(\phi')$ yield the same answer tuple. We can extend the arguments from the proof of that lemma to find a rewriting of $\phi$ that obtains the same answer from $\mathcal{I}_\mathcal{K}^{(0)}$. Hence, assume that $\ell > 0$ is minimal such that $\mathcal{I}_\mathcal{K}^{(\ell)} \models_\mathsf{a} \mathfrak{a}'(\phi')$ holds for some $\phi' \in \Phi_\mathcal{T}$ and a potential answer $\mathfrak{a}'$ for which $\mathfrak{a}(\phi)$ and $\mathfrak{a}'(\phi')$ yield the same answer tuple. Let $\pi$ be a homomorphism of $\mathfrak{a}'(\phi')$ into $\mathcal{I}_\mathcal{K}^{(\ell)}$. Most of the arguments in the proof of Lemma 6.4 still apply here, and many are not necessary (i.e., those for predicates of higher arity), with a few exceptions.

- For the case that **(CR1)** was applied to an inclusion $\exists U.\Pi \sqsubseteq X_2$ to obtain $\mathcal{I}_\mathcal{K}^{(\ell)}$, it suffices to note that, if the application of this inclusion to $\phi'$ results in two nondistinguished concrete domain variables having isomorphic sets of attribute and value comparison atoms, then one of these variables can be removed without affecting the semantics of the query.
- For the case of **(CR4)**, assume that the inclusion $B \sqsubseteq \exists U.\Pi$ was applied to $e \in B^{\mathcal{I}_\mathcal{K}^{(\ell-1)}}$, resulting in $(e, v) \in U^{\mathcal{I}_\mathcal{K}^{(\ell)}}$ and the new atom $\Pi(v)$ in $\Gamma_e^{(\ell)}$. As in the proof of Lemma 6.4, the first step is to rewrite atoms $\Pi'(v')$ that are implied by $\Gamma_e^{(\ell)}$, but not already by $\Gamma_e^{(\ell-1)}$. This can only be the case if $\pi(\mathfrak{a}'(v')) = v$ since otherwise its satisfaction would not depend on the new atom $\Pi(v)$. Hence, we can simply replace $v$ in $\Gamma_e^{(\ell)}$ by $v'$ to obtain an element of $\mathbf{\Gamma}_\mathcal{T}$ that implies $\Pi'(v')$, which can be used by $\mathsf{infer}$. The property (B1) can be preserved by a suitable substitution, as in the case of **(CR1)** above. The atoms $\Pi(v')$ and $U(x, v')$ in $\phi'$ can then be merged as in the proof of Lemma 6.4. Assume now that the resulting two atoms $\Pi(v')$, $U(x, v')$ of $\phi''$ are such that $v'$ is mapped by $\pi$ and $\mathfrak{a}'$ to a constant $d$. By the satisfaction condition $U(x, v')$, it then must be the case that $=(d, v)$ is implied by $\Gamma_e^{(\ell)}$. Since the only atom involving $v$ in $\Gamma_e^{(\ell)}$ is $\Pi(v)$, the predicate $\Pi$ must be of the form $=_d$. Hence, we can split the variable $v'$, and obtain a fresh nondistinguished variable $v''$ and the atoms $=_d(v')$, $=_d(v'')$, and $U(x, v'')$. The first atom is satisfied by $\pi$ and $\mathfrak{a}'$ in $\mathcal{I}_\mathcal{K}^{(0)}$, while the other two can be satisfied by mapping $v''$ to $v$. This finally allows to apply a rewriting step for $B \sqsubseteq \exists U.\Pi$ via $\mathsf{infer}_\mathcal{T}$, obtaining an element of $\Phi_\mathcal{T}$ that is satisfied in $\mathcal{I}_\mathcal{K}^{(\ell-1)}$.

- The case of **(CR6)** can again be treated in the same way. □

We obtain the claimed FO rewriting for unary concrete domains.

**Theorem 6.8.** *If $\mathcal{D}$ is unary, decidable, and satisfies (infinitediff), then safe and $\mathcal{T}$-restricted CQs are FO rewritable w.r.t. DL-Lite$_{core}^{(\mathcal{HF})}(\mathcal{D})$ TBoxes $\mathcal{T}$, and the rewritings are computable.*

*Proof.* This follows directly from Lemmas 5.9 and 6.7. □

Further note that consistency of KBs is still decidable since $\mathcal{D}$ is decidable. Thus, we have extended the results of [38] to a more expressive ontology language, and added the missing condition of $\mathcal{T}$-restrictedness.

# 7 Related Work

The roots of our research lie in the built-in predicates of relational database systems [1]. A lot of effort has been spent on analyzing *query containment* for CQs with built-in predicates in this context, with the usual goal of finding equivalent queries that are easier to evaluate. As expected, built-in predicates increase the complexity from NP to $\Pi_2^{\mathrm{P}}$, PSPACE, or even make the problem undecidable [2, 9, 22].

More recently, concrete domains have been investigated as an extension of classical DL reasoning problem, and found to cause similar problems [26, 28, 30]. Starting with [34, 35], the problem of CQ answering has been investigated in DLs capable of referring to concrete values. Under similar restrictions as for roles, attributes were found to not increase the complexity of this reasoning task. However, it was only in [4, 37, 38] that also the queries were allowed to refer to concrete values, and several known techniques were extended to deal with this case. However, most of this work is restricted to unary concrete domain predicates, which is justified by the OWL 2 standard [31].

Only very recently were CQs with the binary predicate $\leq$ over the rational numbers considered [18, 19]. These papers classify a restricted form of CQs according to their data complexity over the concrete domain $(\mathbb{Q}, \leq)$; the authors obtain a P/CO-NP dichotomy result based on a classification of the patterns of value comparison atoms occurring in the query and TBox. In [19], the abstract canonical model is called *universal pre-model*, and solutions are called *completion functions*. There are several differences to our approach. On the one hand, $(\mathbb{Q}, \leq)$ is not convex, and [19] does not need our safety restriction since the authors are interested also in the CO-NP-hard cases (cf. Lemma 4.4). On the other hand, our queries are not restricted to satisfy the so-called *bounded match depth property*, which in particular holds if the CQ is *rooted*, i.e., all object variables are connected via atoms to a constant or an answer variable, or if the TBox has a finite canonical model w.r.t. all ABoxes (see [19, Lemma 4.3] for details). Moreover, our ontology language is incomparable to the one in [19], which is an extension of *DL-Lite*$_{\mathcal{A}}$ with qualified attribute restrictions of the form $\exists U.\phi$ or $\forall U.\phi$ on the *right-hand side* of concept inclusions, where $\phi$ is a $\mathcal{D}$-conjunction in which all variables except one are existentially quantified. Hence, such restrictions are essentially unary in that they can refer only to one attribute value explicitly; however, they allow to refer to the existence of concrete domain values that are not involved in an attribute. It is easy to see that our logic *DL-Lite*$_{core}^{(\mathcal{HF})}(\mathcal{D})$ can simulate $\exists U.\phi$, and we conjecture that our results could be extended to cover also $\forall U.\phi$. While Lemma 4.4 and results in [4, 37] show that (combined) rewritability cannot hold for the more general setting of non-convex concrete domains with non-safe CQs, it is still open whether the results of [19] can be extended to arbitrary CQs and TBoxes with $\exists U_1, \ldots, U_m.\Pi$ or $\exists U.\phi$ on the *left*-hand side of concept inclusions.

The technique we employ for obtaining the query rewriting is based on the first algorithms for *DL-Lite* [10] and not very sophisticated. There is definitely room for improvement, e.g.,

obtaining a combined rewriting that is (nearly) polynomial in the size of the query and the TBox [23], or rewriting into a different query language like (non-recursive) Datalog [36].

In less closely related research, very expressive *aggregates* over concrete values are considered, but only under so-called *closed world* or *epistemic* semantics, i.e., decoupled from the logical reasoning [16, 20].

# 8 Conclusion

Our combined rewritability result for CQs with built-in predicates over *DL-Lite*$_{core}^{(\mathcal{HF})}(\mathcal{D})$ ontologies establishes for the first time a polynomial data complexity for query answering w.r.t. ontologies formulated in an ontology language with $n$-ary concrete domains. These results subsume the ones of [38] for the case of unary concrete domains, and they are orthogonal to the results in [19]. In the latter work, the data complexity is in general CO-NP, and the authors investigate for which queries this goes down to P.

Until now, our focus was on showing rewritability and complexity results. To be useful in practice, the size of the rewriting needs to be reduced, e.g., by investigating whether more concise rewritings [23] or alternative target languages [36] can be employed in our setting. Instead of considering all possible implications in the concrete domain, it may also be possible to realize the operator $\mathsf{infer}_{\mathcal{D}}$ by a dedicated solving engine for the concrete domain. In addition to considering minor extensions, like allowing for concrete domain variables and predicates in the ABox as in [26], we will also try to extend the language by local identification constraints (keys) [11] and functional roles on the right-hand side of inclusions, and investigate whether FO rewritability holds in the general case.

# References


[1] Serge Abiteboul, Richard Hull, and Victor Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[2] Foto Afrati, Chen Li, and Prasenjit Mitra. Rewriting queries using views in the presence of arithmetic comparisons. *Theoretical Computer Science*, 368(1-2):88–123, 2006. doi:10.1016/j.tcs.2006.08.020.

[3] Alessandro Artale, Diego Calvanese, Roman Kontchakov, and Michael Zakharyaschev. The *DL-Lite* family and relations. *Journal of Artificial Intelligence Research*, 36:1–69, 2009. doi:10.1613/jair.2820.

[4] Alessandro Artale, Vladislav Ryzhikov, and Roman Kontchakov. *DL-Lite* with attributes and datatypes. In Luc De Raedt, Christian Bessiere, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter Lucas, editors, *Proc. of the 20th Eur. Conf. on Artificial Intelligence (ECAI'12)*, volume 242 of *Frontiers in Artificial Intelligence and Applications*, pages 61–66. IOS Press, 2012. doi:10.3233/978-1-61499-098-7-61.

[5] Franz Baader and Philipp Hanschke. A scheme for integrating concrete domains into concept languages. In John Mylopoulos and Raymond Reiter, editors, *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)*, pages 452–457. Morgan Kaufmann, 1991. URL `http://ijcai.org/Proceedings/91-1/Papers/070.pdf`.

[6] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ envelope. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI'05)*, pages 364–369. Professional Book Center, 2005. URL `http://www.ijcai.org/papers/0372.pdf`.

[7] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ envelope. LTCS-Report 05-01, Chair for Automata Theory, TU Dresden, Germany, 2005. https://lat.inf.tu-dresden.de/research/reports-abs.html#BaaderBrandtLutz-LTCS-05-01.

[8] Meghyn Bienvenu, Balder ten Cate, Carsten Lutz, and Frank Wolter. Ontology-based data access: A study through disjunctive datalog, CSP, and MMSNP. In Richard Hull and Wenfei Fan, editors, *Proc. of the 32nd Symp. on Principles of Database Systems (PODS'13)*, pages 213–224. ACM, 2013. doi:10.1145/2463664.2465223.

[9] Nieves R. Brisaboa, Héctor J. Hernández, José R. Paramá, and Miguel R. Penabad. Containment of conjunctive queries with built-in predicates with variables and constants over any ordered domain. In Witold Litwin, Tadeusz Morzy, and Gottfried Vossen, editors, *Proc. of the 2nd East European Symp. on Advances in Databases and Information Systems (ADBIS'98)*, volume 1475 of *Lecture Notes in Computer Science*, pages 46–57. Springer, 1998. doi:10.1007/bfb0057716.

[10] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. *Journal of Automated Reasoning*, 39(3):385–429, 2007. doi:10.1007/s10817-007-9078-x.

[11] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Path-based identification constraints in description logics. In Gerhard Brewka and Jérôme Lang, editors, *Proc. of the 11th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'08)*, pages 231–241. AAAI Press, 2008. URL http://www.aaai.org/Library/KR/2008/kr08-023.php.

[12] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Mariano Rodriguez-Muro, and Riccardo Rosati. Ontologies and databases: The *DL-Lite* approach. In Sergio Tessaris and Enrico Franconi, editors, *Reasoning Web. Semantic Technologies for Informations Systems. 5th Int. Summer School, Tutorial Lectures*, volume 5689 of *Lecture Notes in Computer Science*, pages 255–356. Springer-Verlag, 2009. doi:10.1007/978-3-642-03754-2_7.

[13] Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, Antonella Poggi, Mariano Rodriguez-Muro, Riccardo Rosati, Marco Ruzzi, and Domenico Fabio Savo. The MASTRO system for ontology-based data access. *Semantic Web*, 2:43–53, 2011. doi:10.3233/SW-2011-0029.

[14] Richard Cyganiak, David Wood, and Markus Lanthaler. RDF 1.1 concepts and abstract syntax. W3C recommendation, W3C, February 2014. http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/.

[15] Francesco M. Donini, Maurizio Lenzerini, Daniele Nardi, and Andrea Schaerf. Deduction in concept languages: from subsumption to instance checking. *Journal of Logic and Computation*, 4(4):423–452, 1994. doi:10.1093/logcom/4.4.423.

[16] Birte Glimm and Chimezie Ogbuji. SPARQL 1.1 entailment regimes. W3C recommendation, W3C, March 2013. http://www.w3.org/TR/2013/REC-sparql11-entailment-20130321/.

[17] Georg Gottlob, Marco Manna, and Andreas Pieris. Polynomial combined rewritings for existential rules. In Chitta Baral, Giuseppe De Giacomo, and Thomas Eiter, editors, *Proc. of the 14th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'14)*, pages 268–277. AAAI Press, 2014. URL http://www.aaai.org/ocs/index.php/KR/KR14/paper/view/7973.

[18] André Hernich, Julio Lemos, and Frank Wolter. Constraint patterns for tractable ontology-mediated queries with datatypes. In Maurizio Lenzerini and Rafael Peñaloza, editors, *Proc. of the 29th Int. Workshop on Description Logics (DL'16)*, volume 1577 of *CEUR Workshop Proceedings*, 2016. URL `http://ceur-ws.org/Vol-1577/paper_9.pdf`.

[19] André Hernich, Julio Lemos, and Frank Wolter. Query answering in DL-Lite with datatypes: A non-uniform approach. In Satinder Singh and Shaul Markovitch, editors, *Proc. of the 31st AAAI Conf. on Artificial Intelligence (AAAI'17)*, pages 1142–1148. AAAI Press, 2017. URL `http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14660`.

[20] Evgeny Kharlamov, Yannis Kotidis, Theofilos Mailis, Christian Neuenstadt, Charalampos Nikolaou, Özgür Özçep, Christoforos Svingos, Dmitriy Zheleznyakov, Sebastian Brandt, Ian Horrocks, Yannis Ioannidis, Steffen Lamparter, and Ralf Möller. Towards analytics aware ontology based access to static and streaming data. In Paul Groth, Elena Simperl, Alasdair Gray, Marta Sabou, Markus Krötzsch, Freddy Lecue, Fabian Flöck, and Yolanda Gil, editors, *Proc. of the 15th Int. Semantic Web Conf. (ISWC'16)*, volume 9982 of *Lecture Notes in Computer Science*, pages 344–362. Springer-Verlag, 2016. doi:10.1007/978-3-319-46547-0_31.

[21] Stanislav Kikot, Roman Kontchakov, Vladimir Podolskii, and Michael Zakharyaschev. Exponential lower bounds and separation for query rewriting. In Artur Czumaj, Kurt Mehlhorn, Andrew Pitts, and Roger Wattenhofer, editors, *Proc. of the 39th Int. Coll. on Automata, Languages and Programming (ICALP'12)*, volume 7392 of *Lecture Notes in Computer Science*, pages 263–274. Springer-Verlag, 2012. doi:10.1007/978-3-642-31585-5_26.

[22] Anthony Klug. On conjunctive queries containing inequalities. *Journal of the ACM*, 35(1): 146–160, 1988. doi:10.1145/42267.42273.

[23] Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, and Michael Zakharyaschev. The combined approach to query answering in *DL-Lite*. In Fangzhen Lin, Ulrike Sattler, and Miroslaw Truszczynski, editors, *Proc. of the 12th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'10)*, pages 247–257. AAAI Press, 2010. URL `http://aaai.org/ocs/index.php/KR/KR2010/paper/view/1282`.

[24] Roman Kontchakov, Carsten Lutz, David Toman, Frank Wolter, and Michael Zakharyaschev. The combined approach to ontology-based data access. In Toby Walsh, editor, *Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI'11)*, pages 2656–2661. AAAI Press, 2011. doi:10.5591/978-1-57735-516-8/IJCAI11-442.

[25] Roman Kontchakov, Martin Rezk, Mariano Rodriguez-Muro, Guohui Xiao, and Michael Zakharyaschev. Answering SPARQL queries over databases under OWL 2 QL entailment regime. In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig Knoblock, Denny Vrandečić, Paul Groth, Natasha Noy, Krzysztof Janowicz, and Carole Goble, editors, *Proc. of the 13th Int. Semantic Web Conf. (ISWC'14)*, volume 8796 of *Lecture Notes in Computer Science*, pages 552–567. Springer, 2014. doi:10.1007/978-3-319-11964-9_35.

[26] Carsten Lutz. *The Complexity of Description Logics with Concrete Domains*. PhD thesis, RWTH Aachen, Germany, 2002. URL `https://lat.inf.tu-dresden.de/research/phd/Lutz-PhD-2002.pdf`.

[27] Carsten Lutz. Description logics with concrete domains - a survey. In Philippe Balbiani, Nobu-Yuki Suzuki, Frank Wolter, and Michael Zakharyaschev, editors, *Advances in Modal Logic 4 (AiML'02)*, pages 265–296. King's College Publications, 2003. URL `http://www.aiml.net/volumes/volume4/Lutz.ps`.

[28] Carsten Lutz and Maja Miličić. A tableau algorithm for description logics with concrete domains and general tboxes. *Journal of Automated Reasoning*, 38(1–3):227–259, 2007. doi:10.1007/s10817-006-9049-7.

[29] Carsten Lutz, David Toman, and Frank Wolter. Conjunctive query answering in the description logic $\mathcal{EL}$ using a relational database system. In Craig Boutilier, editor, *Proc. of the 21st Int. Joint Conf. on Artificial Intelligence (IJCAI'09)*, pages 2070–2075. AAAI Press, 2009. URL `http://ijcai.org/papers09/Papers/IJCAI09-341.pdf`.

[30] Despoina Magka, Yevgeny Kazakov, and Ian Horrocks. Tractable extensions of the description logic $\mathcal{EL}$ with numerical datatypes. *Journal of Automated Reasoning*, 47(4):427–450, 2011. doi:10.1007/s10817-011-9235-0.

[31] Boris Motik and Ian Horrocks. OWL datatypes: Design and implementation. In Amit Sheth, Steffen Staab, Mike Dean, Massimo Paolucci, Diana Maynard, Timothy Finin, and Krishnaprasad Thirunarayan, editors, *Proc. of the 7th Int. Semantic Web Conf. (ISWC'08)*, volume 5318 of *Lecture Notes in Computer Science*, pages 307–322. Springer-Verlag, 2008. doi:10.1007/978-3-540-88564-1_20.

[32] Magdalena Ortiz. Ontology based data access: The story so far. In Loreto Bravo and Maurizio Lenzerini, editors, *Proc. of the 7th Alberto Mendelzon Int. Workshop on Foundations of Data Management*, volume 1087 of *CEUR Workshop Proceedings*, 2013. URL `http://ceur-ws.org/Vol-1087/keynote3.pdf`.

[33] Peter Patel-Schneider, Bijan Parsia, and Boris Motik. OWL 2 web ontology language structural specification and functional-style syntax (second edition). W3C recommendation, W3C, 2012. `http://www.w3.org/TR/2012/REC-owl2-syntax-20121211/`.

[34] Antonella Poggi. *Structured and Semi-Structured Data Integration*. PhD thesis, Università degli Studi di Roma "La Sapienza" and Université de Paris Sud, Italy/France, 2006. URL `http://www.dis.uniroma1.it/~poggi/publi/thesis.pdf`.

[35] Antonella Poggi, Domenico Lembo, Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Riccardo Rosati. Linking data to ontologies. *Journal on Data Semantics*, X: 133–173, 2008. doi:10.1007/978-3-540-77688-8_5.

[36] Riccardo Rosati and Alessandro Almatelli. Improving query answering over *DL-Lite* ontologies. In Fangzhen Lin, Ulrike Sattler, and Miroslaw Truszczynski, editors, *Proc. of the 12th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'10)*, pages 290–300. AAAI Press, 2010. URL `http://aaai.org/ocs/index.php/KR/KR2010/paper/view/1400`.

[37] Ognjen Savković. Managing data types in ontology-based data access. Master's thesis, Free University of Bozen-Bolzano, Italy, 2011. URL `http://www.inf.unibz.it/~savkovic/publications/savkovic-msc-thesis-2011.pdf`.

[38] Ognjen Savković and Diego Calvanese. Introducing datatypes in *DL-Lite*. In Luc De Raedt, Christian Bessiere, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter Lucas, editors, *Proc. of the 20th Eur. Conf. on Artificial Intelligence (ECAI'12)*, volume 242 of *Frontiers in Artificial Intelligence and Applications*, pages 720–725. IOS Press, 2012. doi:10.3233/978-1-61499-098-7-720.