A Decidable Very Expressive Description Logic for Databases (Extended Version)

Alessandro Artale, Enrico Franconi, Rafael Peñaloza, Francesco Sportelli

KRDB Research Centre, Free University of Bozen-Bolzano, Italy
{artale,franconi,penaloza,sportelli}@inf.unibz.it

Abstract. We introduce $DLR^+$, an extension of the $n$-ary propositionally closed description logic $DLR$ to deal with attribute-labelled tuples (generalising the positional notation), projections of relations, and global and local objectification of relations, able to express inclusion, functional, key, and external uniqueness dependencies. The logic is equipped with both TBox and ABox axioms. We show how a simple syntactic restriction on the appearance of projections sharing common attributes in a $DLR^+$ knowledge base makes reasoning in the language decidable with the same computational complexity as $DLR$. The obtained $DLR^*$ $n$-ary description logic is able to encode more thoroughly conceptual data models such as EER, UML, and ORM.

1 Introduction

We introduce the description logic (DL) $DLR^+$ extending the $n$-ary DL $DLR$ [6], in order to capture database oriented constraints. While $DLR$ is a rather expressive logic, it lacks a number of expressive means that can be added without increasing the complexity of reasoning—when used in a carefully controlled way. The added expressivity is motivated by the increasing use of DLs as an abstract conceptual layer (an ontology) over relational databases.

A $DLR$ knowledge base can express axioms with (i) propositional combinations of concepts and (compatible) $n$-ary relations, (ii) concepts as unary projections of $n$-ary relations, and (iii) relations with a selected typed component. For example, if Pilot and RacingCar are concepts and DrivesCar, DrivesMotorbike, DrivesVehicle are binary relations, the knowledge base:

$$\text{Pilot} \sqsubseteq \exists[1] \sigma_{2} \text{RacingCar} \text{DrivesCar}$$
$$\text{DrivesCar} \sqcap \text{DrivesMotorbike} \sqsubseteq \text{DrivesVehicle}$$

asserts that a pilot drives a racing car and that driving a car or a motorbike implies driving a vehicle.

The language we propose here, $DLR^+$, extends $DLR$ in the following ways.

– While $DLR$ instances of $n$-ary relations are $n$-tuples of objects—whose components are identified by their position in the tuple—instances of relations
in $\mathcal{DLR}^+$ are attribute-labelled tuples of objects, i.e., tuples where each component is identified by an attribute and not by its position in the tuple (see, e.g., [11]). For example, the relation Employee may have the signature:

$$\text{Employee}(\text{firstname}, \text{lastname}, \text{dept}, \text{deptAddr}),$$

and an instance of Employee could be the tuple:

$$\langle \text{firstname}: \text{John}, \text{lastname}: \text{Doe}, \text{dept}: \text{Purchase}, \text{deptAddr}: \text{London} \rangle.$$

- Attributes can be renamed, for example to recover the positional attributes:

$$\text{firstname}, \text{lastname}, \text{dept}, \text{deptAddr} \mapsto 1, 2, 3, 4.$$

- Relation projections allow to form new relations by projecting a given relation on some of its attributes. For example, if Person is a relation with signature Person($\text{name}, \text{surname}$), it could be related to Employee as follows:

$$\pi[\text{firstname}, \text{lastname}]\text{Employee} \sqsubseteq \text{Person}, \quad \text{firstname}, \text{lastname} \mapsto \text{name}, \text{surname}.$$  

- The objectification of a relation (also known as reification) is a concept whose instances are unique identifiers of the tuples instantiating the relation. Those identifiers could be unique only within an objectified relation (local objectification), or they could be uniquely identifying tuples independently on the relation they are instance of (global objectification). For example, the concept EmployeeC could be the global objectification of the relation Employee, assuming that there is a global 1-to-1 correspondence between pairs of values of the attributes firstname, lastname and EmployeeC instances:

$$\text{EmployeeC} = \bigcirc \exists[\text{firstname}, \text{lastname}]\text{Employee}.$$  

Consider the relations with the following signatures:

$$\text{DrivesCar}($$name, surname, car$$), \quad \text{ OwnsCar}($$name, surname, car$$),$$

and assume that anybody driving a car also owns it: \text{DrivesCar} \sqsubseteq \text{OwnsCar}. The locally objectified events of driving and owning, defined as

$$\text{CarDrivingEvent} \equiv \bigcirc \text{DrivesCar}, \quad \text{CarOwningEvent} \equiv \bigcirc \text{OwnsCar},$$

do not imply that a driving event by a person of a car is the owning event by the same person and the same car: \text{CarDrivingEvent} \not\equiv \text{CarOwningEvent}. Indeed, they are even disjoint: \text{CarDrivingEvent} \cap \text{CarOwningEvent} \sqsubseteq \bot.

It turns out that $\mathcal{DLR}^+$ is an expressive description logic able to assert relevant constraints typical of relational databases. In Section 3 we will consider inclusion dependencies, functional and key dependencies, external uniqueness and identification axioms. For example, $\mathcal{DLR}^+$ can express the fact that the
The syntax of $\mathcal{DLR}^+$.

Fig. 1. The syntax of $\mathcal{DLR}^+$.

attributes $\text{firstname}, \text{lastname}$ play the role of a multi-attribute key for the relation $\text{Employee}$:

$$\pi[\text{firstname}, \text{lastname}]\text{Employee} \subseteq \pi[\text{firstname}, \text{lastname}]\text{Employee},$$

and that the attribute $\text{deptAddr}$ functionally depends on the attribute $\text{dept}$ within the relation $\text{Employee}$:

$$\exists[\text{dept}]\text{Employee} \subseteq \exists[\text{dept}]\pi[\text{dept}, \text{deptAddr}]\text{Employee}.$$

While $\mathcal{DLR}^+$ turns out to be undecidable, we show how a simple syntactic condition on the appearance of projections sharing common attributes in a knowledge base makes the language decidable. The result of this restriction is a new language called $\mathcal{DLR}^\ddagger$. We prove that $\mathcal{DLR}^\ddagger$, while preserving most of the $\mathcal{DLR}^+$ expressivity, has a reasoning problem whose complexity does not increase w.r.t. the computational complexity of the basic $\mathcal{DLR}$ language. We also present in Section 6 the implementation of an API for the reasoning services in $\mathcal{DLR}^\ddagger$.

2 The Description Logic $\mathcal{DLR}^+$

We start by introducing the syntax of $\mathcal{DLR}^+$. A $\mathcal{DLR}^+$ signature is a tuple $\mathcal{L} = (\mathcal{C}, \mathcal{R}, \mathcal{O}, \mathcal{U}, \tau)$ where $\mathcal{C}$, $\mathcal{R}$, $\mathcal{O}$ and $\mathcal{U}$ are finite, mutually disjoint sets of concept names, relation names, individual names, and attributes, respectively, and $\tau$ is a relation signature function, associating a set of attributes to each relation name $\tau(RN) = \{U_1, \ldots, U_n\} \subseteq \mathcal{U}$, with $n \geq 2$.

The syntax of concepts $C$, relations $R$, formulas $\varphi$, and attribute renaming axioms $\vartheta$ is given in Figure 1 where $CN \in \mathcal{C}$, $RN \in \mathcal{R}$, $U \in \mathcal{U}$, $o \in \mathcal{O}$, $q$ is a positive integer and $2 \leq k < \text{arity}(R)$. The arity of a relation $R$ is the number of the attributes in its signature; i.e., $\text{arity}(R) = |\tau(R)|$, with the relation signature function $\tau$ extended to complex relations as in Figure 2. Note that it is possible that the same attribute appears in the signature of different relations.

As mentioned in the introduction, the $\mathcal{DLR}^+$ constructors added to $\mathcal{DLR}$ are the local and global objectification ($\bowtie\mathcal{R}$ and $\widehat{\bowtie}\mathcal{R}$, respectively); relation projections with the possibility to count the projected tuples ($\pi^{\leq q}[U_1, \ldots, U_k]R$), and renaming axioms over attributes ($U_1 \bowtie U_2$). Note that local objectification ($\bowtie\mathcal{R}$) can be applied to relation names, while global objectification ($\widehat{\bowtie}\mathcal{R}$) can be applied to complex relations. We use the standard abbreviations:

$$\bot = C \sqcap \neg C, \quad \top = \bot, \quad C_1 \sqcup C_2 = \neg(C_1 \sqcap \neg C_2), \quad \exists[U_i]R = \exists^{\geq 1}[U_i]R,$$
The semantics of $\mathcal{DLR}^+$ uses the notion of labelled tuples over a potentially infinite domain $\Delta$. Given a set of labels $\mathcal{X} \subseteq \mathcal{U}$ an $\mathcal{X}$-labelled tuple over

$$\exists \rho [U_1]R = \neg(\exists^{\rho+1}[U_1]R), \quad \pi[U_1, \ldots, U_k]R = \pi^{\rho+1}[U_1, \ldots, U_k]R.$$
\((-C)^2 = T \setminus C^T\)
\((C_1 \cap C_2)^2 = C_1^2 \cap C_2^2\)
\((\exists^*[U_i]R)^2 = \{d \in \Delta \mid | \{t \in R^T \mid t[U_i] = d \}| \geq q\}\)
\((\bar{\bigcirc} R)^2 = \{d \in \Delta \mid d = \iota(t) \land t \in R^T\}\)
\((\bar{\bigcirc} R N)^2 = \{d \in \Delta \mid d = \ell_R(t) \land t \in R N^T\}\)
\((R_1 \setminus R_2)^2 = R_1^2 \setminus R_2^2\)
\((R_1 \cap R_2)^2 = R_1^2 \cap R_2^2\)
\((R_1 \cup R_2)^2 = \{t \in R_1^2 \cup R_2^2 \mid \tau(R_1) = \tau(R_2)\}\)
\((\sigma_{U_i \, c}(R))^2 = \{t \in R^T \mid t[U_i] \in C^T\}\)
\((\pi \exists^*[U_1, \ldots, U_k]R)^2 = \langle(U_1 : d_1, \ldots, U_k : d_k) \in T_\Delta([U_1, \ldots, U_k]) \mid 1 \leq |\{t \in R^T \mid t[U_1] = d_1, \ldots, t[U_k] = d_k\}| \leq q\}\)

Fig. 3. The semantics of DLR⁺ expressions.

\(\Delta\) (or tuple for short) is a total function \(t : \mathcal{X} \rightarrow \Delta\). For \(U \in \mathcal{X}\), we write \(t[U]\) to refer to the domain element \(d \in \Delta\) labelled by \(U\). Given \(d_1, \ldots, d_n \in \Delta\), the expression \(\langle U_1 : d_1, \ldots, U_n : d_n \rangle\) stands for the tuple \(t\) defined on the set of labels \(\{U_1, \ldots, U_n\}\) such that \(t[U_i] = d_i\), for \(1 \leq i \leq n\). The projection of the tuple \(t\) over the attributes \(U_1, \ldots, U_k\) is the function \(t\) restricted to be undefined for the labels not in \(U_1, \ldots, U_k\), and it is denoted by \(t[U_1, \ldots, U_k]\). The relation signature function \(\tau\) is extended to labelled tuples to obtain the set of labels on which a tuple is defined. \(T_\Delta(\mathcal{X})\) denotes the set of all \(\mathcal{X}\)-labelled tuples over \(\Delta\), for \(\mathcal{X} \subseteq \mathcal{U}\), and we overload this notation by denoting with \(T_\Delta(\mathcal{U})\) the set of all possible tuples with labels within the whole set of attributes \(\mathcal{U}\).

A DLR⁺ interpretation is a tuple \(I = (\Delta, \mathcal{T}, \iota, \mathcal{L})\) consisting of a nonempty domain \(\Delta\), an interpretation function \(\mathcal{T}\), a global objectification function \(\iota\), and a family \(\mathcal{L}\) containing one local objectification function \(\ell_{RN}\), for each named relation \(RN \in \mathcal{R}\). The global objectification function is an injective function, \(\iota : T_\Delta(\mathcal{U}) \rightarrow \Delta\), associating a unique global identifier to each tuple. The local objectification functions, \(\ell_{RN} : T_\Delta(\mathcal{U}) \rightarrow \Delta\), are associated to each relation name in the signature, and as the global objectification function they are injective: they associate an identifier—which is guaranteed to be unique only within the interpretation of a relation name—to each tuple.

The interpretation function \(\mathcal{T}\) assigns a domain element to each individual, \(o^T \in \Delta\), a set of domain elements to each concept name, \(CN^T \subseteq \Delta\), and a set of \(\tau(RN)\)-labelled tuples over \(\Delta\) to each relation name \(RN\), \(RN^T \subseteq T_\Delta(\tau(RN))\). Note that the unique name assumption is not enforced. The interpretation function \(\mathcal{T}\) is unambiguously extended over concept and relation expressions as specified in Figure 3. Notice that the construct \(\pi \exists^*[U_1, \ldots, U_k]R\) is interpreted as a classical projection over a relation, thus including only tuples belonging to the relation.

The interpretation \(I\) satisfies the concept inclusion axiom \(C_1 \subseteq C_2\) if \(C_1^T \subseteq C_2^T\), and the relation inclusion axiom \(R_1 \subseteq R_2\) if \(R_1^T \subseteq R_2^T\). It satisfies the concept instance axiom \(CN(o)\) if \(o^T \in CN^T\), the relation instance axiom \(RN(U_1 : o_1, \ldots, U_n : o_n)\) if \(\langle U_1 : o_1^T, \ldots, U_n : o_n^T \rangle \in RN^T\), and the axioms \(o_1 = o_2\) and \(o_1 \neq o_2\) if \(o_1^T = o_2^T\).
and \( \alpha_{T} \neq \alpha_{T^{'}} \), respectively. \( \mathcal{I} \) is a model of the knowledge base \((\mathcal{T}, \mathcal{A}, \mathcal{R})\) if it satisfies all the axioms in the TBox \( \mathcal{T} \) and in the ABox \( \mathcal{A} \), once the knowledge base has been rewritten according to the renaming schema.

**Example 1.** Consider the relation names \( R_1, R_2 \) with \( \tau(R_1) = \{W_1, W_2, W_3, W_4\} \), \( \tau(R_2) = \{V_1, V_2, V_3, V_4, V_5\} \), and a knowledge base with the renaming axiom \( W_1 W_2 W_3 \models V_3 V_4 V_5 \) and a TBox \( \mathcal{T}_{\text{exa}} \):

\[
\pi[W_1, W_2] R_1 \subseteq \pi \equiv [W_1, W_2] R_1 \tag{1}
\]

\[
\pi[V_3, V_4] R_2 \subseteq \pi \equiv [V_3, V_4] (\pi[V_3, V_4] R_2) \tag{2}
\]

\[
\pi[W_1, W_2, W_3] R_1 \subseteq \pi[V_3, V_4, V_5] R_2 \tag{3}
\]

The axiom (1) expresses that \( W_1, W_2 \) form a multi-attribute key for \( R_1 \); (2) introduces a functional dependency in the relation \( R_2 \) where the attribute \( V_5 \) is functionally dependent from attributes \( V_3, V_4 \), and (3) states an inclusion between two projections of the relation names \( R_1, R_2 \) based on the renaming schema axiom.

**KB satisfiability** refers to the problem of deciding the existence of a model of a given knowledge base; **concept satisfiability** (resp. **relation satisfiability**) is the problem of deciding whether there is a model of the knowledge base with a non-empty interpretation of a given concept (resp. relation). A knowledge base **entails** (or **logically implies**) an axiom if all models of the knowledge base are also models of the axiom. For instance, it is easy to see that the TBox in Example 1 entails that \( V_3, V_4 \) are a key for \( R_2 \):

\[
\mathcal{T}_{\text{exa}} \models \pi[V_3, V_4] R_2 \subseteq \pi \equiv [V_3, V_4] R_2,
\]

and that axiom (2) is redundant in \( \mathcal{T}_{\text{exa}} \). The decision problems in \( \mathcal{DLR}^{+} \) can be all reduced to KB satisfiability.

**Lemma 2.** In \( \mathcal{DLR}^{+} \), concept and relation satisfiability and entailment are reducible to KB satisfiability.

### 3 Expressiveness of \( \mathcal{DLR}^{+} \)

\( \mathcal{DLR}^{+} \) is an expressive description logic able to assert relevant constraints in the context of relational databases, such as **inclusion dependencies** (namely inclusion axioms among arbitrary projections of relations), **equijoins**, **functional dependency** axioms, **key and foreign key** axioms, **external uniqueness** axioms, **identification** axioms, and **path functional dependencies**.

An **equijoin** among two relations with disjoint signatures is the set of all combinations of tuples in the relations that are equal on their selected attribute names. Let \( R_1, R_2 \) be relations with signatures \( \tau(R_1) = \{U, U_1, \ldots, U_n\} \) and \( \tau(R_2) = \{V, V_1, \ldots, V_m\} \); their equijoin over \( U \) and \( V \) is the relation \( R = R_{1 \bowtie U=V} R_2 \).
with signature $\tau(R) = \tau(R_1) \cup \tau(R_2) \setminus \{V\}$, which is expressed by the $\mathcal{DLR}^+$ axioms:

$$\pi[U, U_1, \ldots, U_n] R \equiv \sigma_{U [\exists U | R_1 \land \exists U | R_2]} R_1$$
$$\pi[V, V_1, \ldots, V_n] R \equiv \sigma_{V [\exists U | R_1 \land \exists U | R_2]} R_2$$
$$U \subseteq V .$$

A functional dependency axiom $(R: U_1 \ldots U_j \rightarrow U)$ (also called internal uniqueness axiom [9]) states that the values of the attributes $U_1 \ldots U_j$ uniquely determine the value of the attribute $U$ in the relation $R$. Formally, the interpretation $I$ satisfies this functional dependency axiom if, for all tuples $s, t \in \mathbb{R}^Z$, $s[U_1] = t[U_1], \ldots, s[U_j] = t[U_j]$ imply $s[U] = t[U]$. Functional dependencies can be expressed in $\mathcal{DLR}^+$, assuming that $\{U_1, \ldots, U_j, U\} \equiv \tau(R)$, with the axiom:

$$\pi[U_1, \ldots, U_j] R \equiv \pi^{\leq 1}[U_1, \ldots, U_j](\pi[U_1, \ldots, U_j, U] R).$$

A special case of a functional dependency are key axioms $(R: U_1 \ldots U_j \rightarrow R)$, which state that the values of the key attributes $U_1 \ldots U_j$ of a relation $R$ uniquely identify tuples in $R$. A key axiom can be expressed in $\mathcal{DLR}^+$, assuming that $\{U_1 \ldots U_j\} \equiv \tau(R)$, with the axiom:

$$\pi[U_1, \ldots, U_j] R \equiv \pi^{\leq 1}[U_1, \ldots, U_j] R.$$

A foreign key is the obvious result of an inclusion dependency together with a key constraint involving the foreign key attributes.

The external uniqueness axiom $([U_1] R_1 \downarrow \ldots \downarrow [U_h] R_h)$ states that the join $R$ of the relations $R_1, \ldots, R_h$ via the attributes $U^1, \ldots, U^h$ has the joined attribute functionally dependent on all the others [9]. This can be expressed in $\mathcal{DLR}^+$ with the axioms:

$$R \equiv R_1 \bowtie \ldots \bowtie R_h$$
$$R : U_1^1 \downarrow \ldots \downarrow U_h^1 \rightarrow U$$

where $\tau(R_i) = \{U_i^1, U_i^2, \ldots, U_i^h\}, 1 \leq i \leq h$, and $R$ is a new relation name with $\tau(R) = \{U_1^1, U_2^1, \ldots, U_h^1\}$.

Identification axioms as defined in $\mathcal{DLR}_{ifid}$ [4] (an extension of $\mathcal{DLR}$ with functional dependencies and identification axioms) are a variant of external uniqueness axioms, constraining only the elements of a concept $C$; they can be expressed in $\mathcal{DLR}^+$ with the axiom:

$$[U^1] \sigma_{U_i^1 : C} R_1 \downarrow \ldots \downarrow [U^h] \sigma_{U_h^1 : C} R_h.$$

Path functional dependencies—as defined in the DL family $\mathcal{CFD}$ [14]—can be expressed in $\mathcal{DLR}^+$ as identification axioms involving joined sequences of functional binary relations. $\mathcal{DLR}^+$ also captures the tree-based identification constraints (tid) introduced in [5] to express functional dependencies in $\mathcal{DL-LiteR}_{RDFS,tid}$. The rich set of constructors in $\mathcal{DLR}^+$ allows us to extend the known mappings in description logics of popular conceptual data models. The EER mapping as introduced in [1] can be extended to deal with multi-attribute keys (by using
identification axioms) and named roles in relations; the ORM mapping as introduced in [13] can be extended to deal with arbitrary subset and exclusive relation constructs (by using inclusions among global objectifications of projections of relations), arbitrary internal and external uniqueness constraints, arbitrary frequency constraints (by using projections), local objectification, named roles in relations, and fact type readings (by using renaming axioms); the UML mapping as introduced in [3] can be fixed to deal properly with association classes (by using local objectification) and named roles in associations.

4 The $\mathcal{DLR}^\pm$ fragment of $\mathcal{DLR}^+$

Since a $\mathcal{DLR}^+$ knowledge base can express inclusions and functional dependencies, the entailment problem is undecidable [7]. Thus, in this section we present $\mathcal{DLR}^\pm$, a decidable syntactic fragment of $\mathcal{DLR}^+$ limiting the coexistence of relation projections in a knowledge base.

Given a $\mathcal{DLR}^+$ knowledge base $KB = (T, A, R)$, we define the projection signature of $KB$ as the set $\tau(RN) \subseteq U$ of all relation RN $\in R$, the singleton sets associated with each attribute name $U \in U$, and the relation signatures that appear explicitly in projection constructs in some axiom from $T$, together with their implicit occurrences due to the renaming schema. Formally, $T$ is the smallest set such that (i) $\tau(RN) \subseteq U$ for all $RN \in R$; (ii) $\{U\} \in T$ for all $U \in U$; and (iii) $\{U_1, \ldots, U_k\} \in T$ for all $\pi \{V_1, \ldots, V_k\} \subseteq T$ appearing as sub-formulas in $T$ and $V_i \in \{U_i\}$ for $1 \leq i \leq k$.

The projection signature graph of $KB$ is the directed acyclic graph corresponding to the Hasse diagram of $T$ ordered by the proper subset relation $\triangleright$, whose sinks are the attribute singletons $\{U\}$. We call this graph $(\triangleright, T)$. Given a set of attributes $\tau = \{U_1, \ldots, U_k\} \subseteq U$, the projection signature graph dominated by $\tau$, denoted as $T_\tau$, is the sub-graph of $(\triangleright, T)$ with $\tau$ as root and containing all the nodes reachable from $\tau$. Given two sets of attributes $\tau_1, \tau_2 \subseteq U$, $\text{PATH}_T(\tau_1, \tau_2)$ denotes the set of paths in $(\triangleright, T)$ between $\tau_1$ and $\tau_2$. Note that, $\text{PATH}_T(\tau_1, \tau_2) = \emptyset$ both when a path does not exist and when $\tau_1 \subseteq \tau_2$. The notation $\text{CHILD}_T(\tau_1, \tau_2)$ means that $\tau_2$ is a child (i.e., a direct descendant) of $\tau_1$ in $(\triangleright, T)$. We now introduce $\mathcal{DLR}^\pm$ as follows.

**Definition 3.** A $\mathcal{DLR}^\pm$ knowledge base is a $\mathcal{DLR}^+$ knowledge base that satisfies the following syntactic conditions:

1. the projection signature graph $(\triangleright, T)$ is a multitree: i.e., for every node $\tau \in T$, the graph $T_\tau$ is a tree; and
2. for every projection construct $\pi \{U_1, \ldots, U_k\} \subseteq T$ and every concept expression of the form $\exists q[U]R$ appearing in $T$, if $q > 1$ then the length of the path $\text{PATH}_T(\tau(R), \{U_1, \ldots, U_k\})$ is 1.

The first condition in $\mathcal{DLR}^\pm$ restrict $\mathcal{DLR}^+$ in the way that multiple projections of relations may appear in a knowledge base: intuitively, there cannot be different projections sharing a common attribute. Moreover, observe that in $\mathcal{DLR}^\pm$...
Fig. 4. The projection signature graph of Example 1.

$\mathcal{DLR}$ is necessarily functional, due to the multitree restriction. By relaxing the first condition the language becomes undecidable, as we mentioned at the beginning of this Section. The second condition is also necessary to prove decidability of $\mathcal{DLR}$ (see the proof in the next Section); however, we do not know whether this condition could be relaxed while preserving decidability.

Figure 4 shows that the projection signature graph of the knowledge base from Example 1 is indeed a multitree. Note that in the figure we have collapsed equivalent attributes in a unique equivalence class, according to the renaming schema. Furthermore, since all its projection constructs have $q = 1$, this knowledge base belongs to $\mathcal{DLR}$.

$\mathcal{DLR}$ is included in $\mathcal{DLR}$, since the projection signature graph of any $\mathcal{DLR}$ knowledge base is always a degenerate multitree with maximum depth equal to 1. Not all the database constraints as introduced in Section 3 can be directly expressed in $\mathcal{DLR}$. While functional dependency and key axioms can be expressed directly in $\mathcal{DLR}$, equijoins, external uniqueness axioms, and identification axioms introduce projections of a relation which share common attributes, thus violating the multitree restriction. For example, the axioms for capturing an equijoin between two relations, $R_1, R_2$ would generate a projection signature graph with the signatures of $R_1, R_2$ as projections of the signature of the join relation $R$ sharing the attribute on which the join is performed, thus violating condition 1.

However, in $\mathcal{DLR}$ it is still possible to reason over both external uniqueness and identification axioms by encoding them into a set of saturated ABoxes (as originally proposed in [4]) and check whether there is a saturation that satisfies the constraints. Therefore, we can conclude that $\mathcal{DLR}_{ifd}$ extended with unary functional dependencies is included in $\mathcal{DLR}$, provided that projections of relations in the knowledge base form a multitree projection signature graph. Since (unary) functional dependencies are expressed via the inclusions of projections...
of relations, by constraining the projection signature graph to be a multitree, the possibility to build combinations of functional dependencies as the ones in \[4\] leading to undecidability is ruled out.

Note that the non-conflicting keys sufficient condition guaranteeing the decidability of inclusion dependencies and keys of \[12\] is in conflict with our more restrictive requirement: indeed \[12\] allow for overlapping projections, but the considered datalog language is not comparable to DLR\(^+\).

Concerning the ability of DLR\(^\pm\) to capture conceptual data models, only the mapping of ORM schemas is affected by the DLR\(^\pm\) restrictions: DLR\(^\pm\) is able to correctly express an ORM schema if the projections involved in the schema satisfy the DLR\(^\pm\) multitree restriction.

5 Mapping DLR\(^\pm\) to ALCQI

This section shows constructively the main technical result of this paper, i.e., that reasoning in DLR\(^\pm\) is an ExpTime-complete problem. The lower bound is clear by observing that DLR is a sublanguage of DLR\(^\pm\). More challenging is the upper bound obtained by providing a mapping from DLR\(^\pm\) KBs to ALCQI KBs—a Boolean complete DL with qualified number restrictions of the form \(\exists q R C\), and inverse roles of the form \(R^-\) (see [2] for more details). We adapt and extend the mapping presented for DLR in [6], with the modifications proposed by [10] to deal with ABoxes without the unique name assumption.

We recall that the renaming schema, \(\mathcal{R}\), does not play any role since we assumed that a DLR\(^\pm\) KB is rewritten by choosing a single canonical representative, \([U]_R\), for each \(V \in [U]_R\). Thus, we consider DLR\(^\pm\) KBs as pairs of TBox and ABox axioms.

We first introduce a mapping function \(\downarrow\) from DLR\(^\pm\) concepts and relations to ALCQI concepts. The function \(\downarrow\) maps each concept name \(CN\) and each relation name \(RN\) appearing in the DLR\(^\pm\) KB to an ALCQI concept names \(CN\) and \(A_{RN}\), respectively. The latter is the global reification of \(RN\). For each relation name \(RN\), the ALCQI signature also includes a concept name \(A_{CN}\) and a role name \(Q_{RN}\) to capture local objectification. The mapping \(\downarrow\) is extended to concept and relation expressions as illustrated in Figure 5, where the notation \(\exists^1 \exists q R C\) is a shortcut for the conjunction \(\exists R C \land \exists q R C\).

The mapping crucially uses the projection signature graph to map projections and selections, by accessing paths in the projection signature graph (\(\tau, \mathcal{T}\)) associated to the DLR\(^\pm\) KB. If there is a path \(\text{PATH}_{\mathcal{T}}(\tau, \tau') = \tau, \tau_1, \ldots, \tau_n, \tau'\) from \(\tau\) to \(\tau'\) in \(\mathcal{T}\), then the ALCQI signature contains role names \(Q_{\tau_i}, Q_{\tau_n}\), for \(i = 1, \ldots, n\), and the following role chain expression is generated by the mapping:

\[
\text{PATH}_{\mathcal{T}}(\tau, \tau') \downarrow = Q_{\tau_1} \circ \ldots \circ Q_{\tau_n} \circ Q_{\tau'},
\]

In particular, the mapping uses the following notation: the inverse role chain \((R_1 \circ \ldots \circ R_n)^-\), for \(R_i\) a role name, stands for the chain \(R_1^- \circ \ldots \circ R_n^-\), with \(R_i^-\) an inverse role, the expression \(\exists^1 R_1 \circ \ldots \circ R_n C\) stands for the ALCQI
by the projection signature graph it dominates. Thus, to access, e.g., an at-
that each

DLR

labelled by the roles (Q for A

ALCQI

tions. The target

KB denotes a relation projection and the mapping reifies each of the se projec-
τ

R N

projections of

ing global reification concept (A

A

syntax,
U

|}

p

τ

R N

pro.

τ

R N

for each projected signature

path

ij

in the projection signature graph dominated by

τ

R N

. . . , U

k

resulting from mapping the

C

concept expression

@

Def. 3), the above notation shows that we remain within the

ALCQI

concept expression

˘

when the mapping applies to cardinalities. If, e.g., we need to map the

concept expression

@,

finally, notice that the mapping introduces a concept

name

A

for concept and relation expressions.

Fig. 5. The mapping to

ALCQI

for concept and relation expressions.

concept expression \(\exists^{\aleph_1}R_1, \ldots, \exists^{\aleph_1}R_n, C\) and \(\forall R_1 \circ \ldots \circ R_n, C\) for the

ALCQI

concept expression \(\forall R_1, \ldots, \forall R_n, C\). Thus, since

DLR\(^\pm\)

restricts to \(q = 1\) the cardinalities on any path of length strictly greater than 1 (see condition 2 in Def. 3), the above notation shows that we remain within the

ALCQI

syntax when the mapping applies to cardinalities. If, e.g., we need to map the

DLR\(^\pm\)

cardinality constraint \(\exists^{\aleph_q}[U_i]R\) with \(q > 1\), then, to stay within the

ALCQI

syntax, \(U_i\) must not be mentioned in any other projection in such a way that \(\text{card}([U_i]) = 1\). Finally, notice that the mapping introduces a concept

name

A\(^{\sigma}(RN)\)

for each projected signature \(\tau_i\) in the projection signature graph

dominated by \(\tau(RN)\), i.e., \(\tau_i \in \mathcal{T}_{\tau(RN)}\), to capture global reifications of the various

paths of \(RN\) in the given KB. We also use the shortcut \(A_{RN}\) which stands

for \(A\(^{\sigma}(RN)\)\).

Intuitively, each node in the projection signature graph associated to a

DLR\(^\pm\)

KB denotes a relation projection and the mapping reifies each of these projections. The target

ALCQI

signature resulting from mapping the

DLR\(^\pm\)

KB of Example [1] is partially presented in Fig. 6 together with the projection signature graph (showed in Fig. 3). Each node of the graph is labelled with the corresponding global reification concept \(A\(^{\tau}(R_i)\)\), for each \(R_i \in \mathcal{R}\) and each projected signature \(\tau_j\) in the projection signature graph dominated by \(\tau(R_i)\), while the edges are labelled by the roles \((Q_{\tau_j})\) needed for the reification.

To better clarify the need for the path function in the mapping, notice that each

DLR\(^\pm\)

relation is reified according to the decomposition dictated by the projection signature graph it dominates. Thus, to access, e.g., an at-
tribute $U_j$ of a $\mathcal{DLR}^\pm$ relation $R_j$ it is necessary to follow the path through the projections that use that attribute. Such a path, from the node denoting the whole signature of the relation, $\tau(R_j)$, to the node denoting the attribute $U_j$ is returned by the $\text{PATH}_\tau(\tau(R_j), U_j)$ function. For instance, considering the example from Figure 6, to access the attribute $W_1$ of the relation $R_2$ in the expression $(\sigma_{W_1:C}R_2)$, the mapping of the path $\text{PATH}_{\tau}(\tau(R_2),\{W_1\})^\dagger$ is equal to the role chain $Q(W_1,W_2,W_3) \circ Q(W_1,W_2) \circ Q(W_1)$. This means that $(\sigma_{W_1:C}R_2)^\dagger = A_{R_2} \cap \forall Q(W_1,W_2,W_3), \forall Q(W_1,W_2), \forall Q(W_1), C$. Similar considerations can be done when mapping cardinalities over relation projections.

We now present in details the mapping of a $\mathcal{DLR}^\pm$ KB into a KB in $\mathcal{ALCQI}$. Let $KB = (T, A)$ be a $\mathcal{DLR}^\pm$ KB with signature $(\mathcal{C}, \mathcal{R}, \mathcal{O}, \mathcal{U}, \tau)$. The mapping $\gamma(KB)$ is assumed to be unsatisfiable (i.e., it contains the axiom $\top \sqsubseteq \bot$) if the ABox contains the relation assertion $RN(t)$ with $\tau(RN) \neq \tau(t)$, for some relation $RN \in \mathcal{R}$ and some tuple $t$. Otherwise, $\gamma(KB) = (\gamma(T), \gamma(A))$ defines an $\mathcal{ALCQI}$ KB as follows:

\[
\gamma(T) = \gamma_{daj} \cup \bigcup_{RN \in \mathcal{R}} \gamma_{rel}(RN) \cup \bigcup_{RN \in \mathcal{R}} \gamma_{obj}(RN) \cup \bigcup_{C_1 \subseteq C_2 \in KKB} C_1^t \subseteq C_2^t \cup \bigcup_{R_1 \subseteq R_2 \in KKB} R_1^t \subseteq R_2^t
\]

\[
\gamma_{daj} = \{ \lnot A_{RN_1}^{\tau(RN_1)} \cup \lnot A_{RN_2}^{\tau(RN_2)} \mid RN_1, RN_2 \in \mathcal{R},
\tau_1 \in \mathcal{F}_{\tau}(RN_1), \tau_2 \in \mathcal{F}_{\tau}(RN_2), \lvert \tau_1 \rvert \geq 2, \lvert \tau_2 \rvert \geq 2, \tau_1 \neq \tau_2 \}
\]

\[
\gamma_{rel}(RN) = \bigcup_{\tau_i \in \mathcal{F}_{\tau}(RN) \text{ CHILD}_{\tau}(\tau_i, \tau_j)} \{ \lnot A_{RN}^\tau \mid \exists Q_{\tau_1} \cup \exists Q_{\tau_2} \mid \top \sqsubseteq \bot \}
\]
Following [10], the mapping \( \gamma(A) = \{ CN^\dagger(o) \mid CN(o) \in A \} \cup \)
\[
\{ a_1 \neq a_2 \mid a_1 \neq a_2 \in A \} \cup \{ a_1 = a_2 \mid a_1 = a_2 \in A \} \cup \\
\{ A_{RN}^\dagger(\xi(t[\tau_1])) \mid RN(t) \in A \text{ and } \tau_1 \in \mathcal{F}(RN) \} \cup \\
\{ Q_{\tau_1}(\xi(t[\tau_1]), \xi(t[\tau_1])) \mid RN(t) \in A, \tau_1 \in \mathcal{F}(RN) \text{ and } \text{CHILD}(\tau_1, \tau_2) \} \cup \\
\{ Q_{\xi}(o) \mid o \in O \} \cup \\
\{ Q_{t}(a_1) \mid t = \langle U_1 : a_1, \ldots, U_n : a_n \rangle \text{ occurs in } A \}. \\
\]
(5)

Fig. 7. The mapping \( \gamma(A) \)

\[
\gamma_{\text{obj}}(RN) = \{ A_{RN} \subseteq \exists Q_{RN} \cdot A_{RN}^1, \exists^2 Q_{RN} \cdot T \subseteq \bot, \\
A_{RN}^1 \subseteq \exists Q_{RN} \cdot A_{RN}, \exists^2 Q_{RN} \cdot T \subseteq \bot \}. \\
\]

Intuitively, \( \gamma_{\text{obj}} \) ensures that relations with different signatures are disjoint, thus, e.g., enforcing the union compatibility. The axioms in \( \gamma_{\text{rel}} \) introduce classical reification axioms for each relation and its relevant projections. The axioms in \( \gamma_{\text{obj}} \) make sure that each local objectification differs from the global one while each role \( Q_{RN} \) defines a bijection.

To translate the ABox, we first map each individual \( o \in O \) in the \( \mathcal{DLR}^\pm \) ABox \( A \) to an \( ALCQI \) individual \( o \). Each relation instance occurring in \( A \) is mapped via an injective function \( \xi \) to a distinct individual. That is, \( \xi : T_O(U) \to O_{ALCQI} \), with \( O_{ALCQI} = O \cup O^\prime \) being the set of individual names in \( \gamma(KB) \), \( O \cap O^\prime = \emptyset \) and
\[
\xi(t) = \begin{cases} 
  o \in O, & \text{if } t = \langle U \cdot o \rangle \\
  o \in O^\prime, & \text{otherwise.}
\end{cases}
\]

Following [10], the mapping \( \gamma(A) \) in Fig. 7 introduces a new concept name \( Q_o \) for each individual \( o \in O \) and a new concept name \( Q_t \) for each relation instance \( t \) occurring in \( A \), with each \( Q_t \) restricted as follows:
\[
Q_t \subseteq \exists \forall \langle \text{PATH}_\mathcal{F}(\tau(t), \{ U_1 \})^\dagger \rangle. \\
\exists \langle \text{PATH}_\mathcal{F}(\tau(t), \{ U_2 \})^\dagger \rangle Q_{a_1} \cap \ldots \cap \exists \langle \text{PATH}_\mathcal{F}(\tau(t), \{ U_n \})^\dagger \rangle Q_{a_n}. \\
\]
(4)

Intuitively, (7) and (8) reify each relation instance occurring in \( A \) using the projection signature of the relation instance itself. The formulas (9)-(10) together with the axioms for concepts \( Q_t \) guarantee that there is exactly one \( ALCQI \) individual reifying a given relation instance. Clearly, the size of \( \gamma(KB) \) is polynomial in the size of \( KB \) under the same coding of the numerical parameters.

We are now able to state our main results.

**Theorem 4.** A \( \mathcal{DLR}^\pm \) knowledge base \( KB \) is satisfiable iff the \( ALCQI \) knowledge base \( \gamma(KB) \) is satisfiable.

**Proof.** We assume that the \( KB \) is consistently rewritten by substituting each attribute with its canonical representative, thus, we do not have to deal with
We first show that

\[
\ell \gamma(\mathcal{T}) = \gamma(\mathcal{T})_i
\]

and

\[
\tau_{\mathcal{T}}(\mathcal{R}_N) = \gamma(\mathcal{T}) \text{ for the } \mathcal{ALCQI} \text{ knowledge base } \gamma(\mathcal{KB}) \text{ we set } \Delta^J = \Delta^J, \sigma^J = \sigma^J \text{ for all } o \in \mathcal{O} \text{ and }
\]

\[
[\ell(\{U_1 \cdot o_1, \ldots, U_n \cdot o_n\})]^J = \ell(\{U_1 \cdot o_1^J, \ldots, U_n \cdot o_n^J\}). \tag{11}
\]

Furthermore, we set: \((CN^1)^J = (CN)^J\), for every atomic concept \(CN \in \mathcal{C}\), while for every \(RN \in \mathcal{R}\) and \(\tau_i \in \mathcal{F}(\mathcal{R}_N)\) we set

\[
(A^J_{\mathcal{R}_N})^J = \{i(\{U_1 : d_1, \ldots, U_k : d_k\}) \mid \{U_1, \ldots, U_k\} = \tau_i \text{ and } \exists t \in \mathcal{R}_N^J. t[U_1] = d_1, \ldots, t[U_k] = d_k\}. \tag{12}
\]

For each role name \(Q_{\tau_i}, \tau_i \in \mathcal{F}\), we set

\[
(Q_{\tau_i})^J = \{(d_1, d_2) \in \Delta^J \times \Delta^J \mid \exists t \in \mathcal{R}_N^J \text{ s.t. } d_1 = \ell(\tau_i), d_2 = \ell(\tau_i) \text{ and } \text{CHILD}_J(\tau_j, \tau_i), \text{ for some } RN \in \mathcal{R}\}. \tag{13}
\]

For every \(RN \in \mathcal{R}\) we set

\[
Q^J_{\mathcal{R}_N} = \{(d_1, d_2) \in \Delta^J \times \Delta^J \mid \exists t \in \mathcal{R}_N^J \text{ s.t. } d_1 = \ell(t) \text{ and } d_2 = \ell(\mathcal{R}_N(t))\}, \tag{14}
\]

and

\[
(A^J_{\mathcal{R}_N})^J = \{\ell(\mathcal{R}_N(t)) \mid t \in \mathcal{R}_N^J\}. \tag{15}
\]

We first show that \(\mathcal{J}\) is indeed a model of \(\gamma(\mathcal{T})\).

1. \(\mathcal{J} \models \gamma_{\text{def}}\). This is a direct consequence of the fact that \(\ell\) is an injective function and that tuples with different signatures are different tuples.
2. \(\mathcal{J} \models \gamma_{\text{red}}(\mathcal{R}_N)\), for every \(RN \in \mathcal{R}\). We show that, for each \(\tau_i, \tau_j\) such that \(\text{CHILD}_J(\tau_i, \tau_j)\) and \(\tau_i \in \mathcal{F}(\mathcal{R}_N)\), it holds that \(\mathcal{J} \models A^J_{\mathcal{R}_N} \subseteq \exists Q_{\tau_j}, A^J_{\mathcal{R}_N}\) and

   \[
   \mathcal{J} \models \exists \exists^2 Q_{\tau_j}, T \subseteq \mathcal{T}.
   \]

   - \(\mathcal{J} \models A^J_{\mathcal{R}_N} \subseteq \exists Q_{\tau_j}, A^J_{\mathcal{R}_N}\). Let \(d \in (A^J_{\mathcal{R}_N})^J\), by \(\text{(12)}\), \(\exists t \in \mathcal{R}_N^J \text{ s.t. } d = \ell(\tau_i)\).

   Since \(\text{CHILD}_J(\tau_i, \tau_j)\), then \(\exists d' = \ell(\tau_j)\) and, by \(\text{(13)}\), \((d, d') \in Q^J_{\mathcal{R}_N}\), while by \(\text{(12)}\), \(d' \in (A^J_{\mathcal{R}_N})^J\). Thus, \(d \in (\exists Q_{\tau_j}, A^J_{\mathcal{R}_N})^J\).

   - \(\mathcal{J} \models \exists \exists^2 Q_{\tau_j}, T \subseteq \mathcal{T}\). The fact that each \(Q_{\tau_j}\) is interpreted as a functional role is a direct consequence of the construction \(\text{(13)}\) and the fact that \(\ell\) is an injective function.
3. \(\mathcal{J} \models \gamma_{\text{lop}}(\mathcal{R}_N)\), for every \(RN \in \mathcal{R}\). Similar as above, considering the fact that each \(\ell(\mathcal{R}_N)\) is an injective function and equations \(\text{(14)-(15)}\).
4. \(\mathcal{J} \models C^J_1 \subseteq C^J_2\) and \(\mathcal{J} \models R^J_1 \subseteq R^J_2\). Since \(\mathcal{I} \models C_1 \subseteq C_2\) and \(\mathcal{I} \models R_1 \subseteq R_2\), it is enough to show the following:

   - \(d \in C^J\) iff \(d \in (C^\dagger)^J\), for all \(\mathcal{DLCR}^\pm\) concepts;

   - \(t \in R^J\) iff \(\ell(t) \in (R^\dagger)^J\), for all \(\mathcal{DLCR}^\pm\) relations.
Before we proceed with the proof, it is easy to show by structural induction that the following property holds:

If $i(t) = R^1$ then $\exists i(t') \in RN^{1,J}$ s.t. $t = t'[^{\tau(R)}]$, for some $RN \in R$. \hspace{1cm} (16)

We now proceed with the proof by structural induction. The base cases, for atomic concepts and roles, are immediate form the definition of both $CN^J$ and $RN^J$. The cases where complex concepts and relations are constructed using either boolean operators, relation difference or global reification are easy to show. We thus show only the following cases.

Let $d \in (C \circ RN)^J$. Then, $d = \ell_{RN}(t)$ with $t \in RN^T$. By induction, $i(t) \in A^J_{RN}$ and, by $\gamma_{obj}(RN)$, there is a $d' \in \Delta^J$ s.t. $(i(t), d') \in Q^J_{RN}$ and $d' \in (A^J_{RN})^J$. By (14), $d' = \ell_{RN}(t)$ and, since $\ell_{RN}$ is injective, $d = d'$. Thus, $d \in (C \circ RN)^J$.

Let $d \in (3 \circ [U_i] R)^J$. Then, there are different $t_1, \ldots, t_q \in R^2$ s.t. $t_i[U_i] = d$, for all $i = 1, \ldots, q$. By induction, $i(t_i) \in R^1$ while, by (16), $i(t_i') \in RN^J$, for some atomic relation $RN \in R$ and a tuple $t_i'$ s.t. $t_i = t_i'[\tau(R)]$. By $\gamma_{rel}(RN)$ and (13), $(i(t_i'), i(t_i)) \in (pen \circ (\tau(R), \tau(R)))^J$ and $(i(t_i)), d) \in (pen \circ (\tau(R), \{U_i\}))^J$.

Since $i$ is injective, $i(t_i') \neq i(t_i)$ when $i \neq j$, thus, $d \in (3 \circ [U_i] R)^J$.

Let $t \in (\exists [U_i \circ C] R)^J$. Then, $t \in R^2$ and $t[U_i] \in C^J$ and, by induction, $i(t) \in R^1$ and $t[U_i] \in \bar{C}$. As before, by $\gamma_{rel}(RN)$ and by (13) and (16), we have $(i(t), t[U_i]) \in (pen \circ (\tau(R), \{U_i\}))^J$. Since $pen \circ (\tau(R), \{U_i\})$ is functional, then we have that $i(t) \in (\exists [U_i \circ C] R)^J$.

Let $t \in (3 [U_i, \ldots, U_k] R)^J$. Then, there is a tuple $t' \in R^2$ s.t. $t'[U_i, \ldots, U_k] = t$ and, by induction, $i(t') \in R^1$. As before, by $\gamma_{rel}(RN)$ and by (13) and (16), we can show that $(i(t'), i(t)) \in pen \circ (\tau(R), \{U_i, \ldots, U_k\})^J$ and thus it follows that $i(t) \in (\exists [U_i, \ldots, U_k] R)^J$.

All the other cases can be proved in a similar way. We now show the vice versa.

Let $d \in (C \circ RN)^J$. Then, $d \in (A^J_{RN})^J$ and $d = \ell_{RN}(t)$, for some $t \in RN^T$, i.e., $d \in (C \circ RN)^J$.

Let $d \in (3 \circ [U_i] R)^J$. Then, there are different $d_1, \ldots, d_q \in \Delta^J$ such that $(d_i, d) \in (pen \circ (\tau(R), \{U_i\}))^J$ and $d_i \in R^1$, for $l = 1, \ldots, q$. By induction, each $d_i = i(t_i)$ and $t_i \in R^2$. Since $i$ is injective, then $t_i \neq t_j$ for all $l, j = 1, \ldots, q$, $i \neq j$. We need to show that $t_i[U_i] = d$, for all $i = 1, \ldots, q$. By (14) and the fact that $(d_i, d) \in (pen \circ (\tau(R), \{U_i\}))^J$, then $d = i(t_i[U_i]) = t_i[U_i]$.

Let $i(t) \in (\exists [U_i \circ C] R)^J$. Then, $i(t) \in R^1$ and, by induction, $t \in R^2$. Let $t[U_i] = d$. We need to show that $d \in C^J$. By $\gamma_{rel}(RN)$ and by (13) and (16), it follows that $(i(t), d) \in (pen \circ (\tau(R), \{U_i\}))^J$, then $d \in C^J$ and, by induction, $d \in C^T$.

Let $i(t) \in (\exists [U_i, \ldots, U_k] R)^J$. Then, there is $d \in \Delta^J$ s.t.

$$(d, i(t)) \in (pen \circ (\tau(R), \{U_i, \ldots, U_k\}))^J$$

and $d \in R^1$. By induction, $d = i(t')$ and $t' \in R^2$. By the definition of the mapping of paths and (13), $i(t) = i(t'[U_i, \ldots, U_k])$, i.e., $t = t'[U_i, \ldots, U_k]$. Thus, $t \in (\exists [U_i, \ldots, U_k] R)^2$.

We now show that $J$ is a model of $\gamma(A)$. 
Concerning axioms in (6) and (10) they are satisfied by construction. $J$ also satisfies axioms in (7) and in (8) due to (12) and (13), respectively, and the interpretation of $\xi$ in (14). Concerning axioms in (9)-(10), we set $Q^J_{\alpha} = \{\alpha^J\}$, for each $\alpha \in \mathcal{O}$, and $Q^J_1 = \{1^J\}$, for each tuple $t = \langle U_1; \alpha_1, \ldots, U_n; \alpha_n \rangle$ occurring in $A$. We finally show that $J$ satisfies axiom (4) by considering, w.l.o.g., the case of binary tuples, $t = \langle U_1; \alpha_1, U_2; \alpha_2 \rangle$. Then, $\text{path}_J(\tau(t), \{U_1\}) = Q_U^J$, and $\text{path}_J(\tau(t), \{U_2\}) = Q_{U_2}^J$. Assume that $\alpha_1^J \in Q_{U_1}^J$ and that there are objects $d_1, d_2, d_3, d_4 \in \Delta^J$ such that $(d_1, \alpha_1^J), (d_2, \alpha_1^J) \in Q_{U_1}^J$, $(d_1, d_3), (d_2, d_4) \in Q_{U_2}^J$ and $d_3, d_4 \in Q_{\alpha_2}$. We need to show that $d_1 = d_2$. We first notice that, since concepts $Q_\alpha$ are interpreted as singleton, $d_3, d_4 \in 0^J$. Furthermore, by (13), $d_1 = \iota(t_1)$ and $d_2 = \iota(t_2)$, with $t_1 = \langle U_1; \alpha_1^J, U_2; d_3 \rangle$ and $t_2 = \langle U_1; \alpha_1^J, U_2; d_4 \rangle$ and thus $t_1 = t_2$. Since $\iota$ is injective, then $d_1 = d_2$.

$J = (\Delta^J, \cdot^J)$ be a model for the knowledge base $\gamma(\mathcal{KB})$. Without loss of generality, we can assume that $J$ is a forest model. We then construct a model $I = (\Delta^I, \cdot^I, \rho, i, \ell_{\mathcal{RN}}, \ldots,)$ for a $\mathcal{ALCQI}$ knowledge base $\mathcal{KB}$. We set: $\Delta^I = \Delta^J$, $\cdot^I = \cdot^J$ for all $\alpha \in \mathcal{O}$, $CN^I = (CN^J)^J$, for every atomic concept $CN \in \mathcal{C}$, while, for every $\mathcal{RN} \in \mathcal{R}$, we set:

$$R_{\mathcal{RN}}^I = \{t = \langle U_1; d_1, \ldots, U_n; d_n \rangle \in T_{\Delta^I}(\tau(\mathcal{RN})) \mid \exists d \in A_{\mathcal{RN}}^J \text{ s.t.} (\ell(t[U_1]) = \text{path}_J(\tau(\mathcal{RN}), \{U_1\})^J \text{ for } i = 1, \ldots, n\}. \quad (17)$$

Notice that (17) defines a bijection between objects in $\mathcal{ALCQI}$ reifying tuples and tuples themselves. Indeed, since $J$ satisfies $\gamma_{rel}(\mathcal{RN})$, for every $d \in A_{\mathcal{RN}}^J$ there is a unique tuple $\langle U_1; d_1, \ldots, U_n; d_n \rangle \in R_{\mathcal{RN}}^I$—thus we say that $d$ generates $\langle U_1; d_1, \ldots, U_n; d_n \rangle$ and, in symbols, $d \rightarrow \langle U_1; d_1, \ldots, U_n; d_n \rangle$. Furthermore, since $J$ is forest shaped, to each tuple whose components are not in the ABox corresponds a unique $d$ that generates it. On the other hand, since $J$ satisfies axiom (4), then also for tuples occurring in the ABox there is a unique $d$ that generates them. Thus, let $d \rightarrow \langle U_1; d_1, \ldots, U_n; d_n \rangle$, by setting $\iota(\langle U_1; d_1, \ldots, U_n; d_n \rangle) = d$ and

$$\iota(\langle U_1; d_1, \ldots, U_n; d_n \rangle[\tau_1]) = d_{\tau_1}, \text{ s.t.} \quad (d, d_{\tau_1}) \in \text{path}_J(\langle U_1, \ldots, U_n \rangle, \tau_1)^J, \quad (18)$$

for all $\tau_1 \in J$ s.t. $\tau_1 \subseteq \{U_1, \ldots, U_n\}$, then, the function $\iota$ is as required.

By setting

$$\ell_{\mathcal{RN}}(\langle U_1; d_1, \ldots, U_n; d_n \rangle) = d, \text{ s.t.} \quad (\iota(\langle U_1; d_1, \ldots, U_n; d_n \rangle), d) \in Q_{\mathcal{RN}}^J, \quad (19)$$

then, by $\gamma_{lob}(\mathcal{RN})$, both $Q_{\mathcal{RN}}$ and its inverse are interpreted as a functional roles by $J$, thus the function $\ell_{\mathcal{RN}}$ is as required.

It is easy to show by structural induction that the following property holds:

If $t \in R^J$ then $\exists t' \in R_{\mathcal{RN}}^I$ s.t. $t = t'^{[\tau(R)]}$, for some $\mathcal{RN} \in \mathcal{R}$. \quad (20)
We now show that $I$ is indeed a model of $KB$. We first show that $I \models T$, i.e., $I \models C_1 \subseteq C_2$ and $I \models R_1 \subseteq R_2$. As before, since $J \models C'_1 \subseteq C'_2$ and $J \models R'_1 \subseteq R'_2$, it is enough to show the following:

- $d \in C^J_2$ iff $d \in (C^J_1)^J$, for all $\mathcal{DLR}^\pm$ concepts;
- $t \in R^J_2$ iff $t(\bar{t}) \in (R^J_1)^J$, for all $\mathcal{DLR}^\pm$ relations.

The proof is by structural induction. The base cases are trivially true. Similarly for the boolean operators, difference between relations and global, we thus show only the following cases.

Let $d \in (\bigcirc \mathcal{RN})^J$. Then, $d = t_{\mathcal{RN}}(t)$ with $t \in \mathcal{RN}^J$. By induction, $\forall \bar{t} \in A_{\mathcal{RN}}^J$ and by $\gamma_{lobj}(\mathcal{RN})$, there is a $d' \in \Delta^J$ s.t. $(\bar{t}(\bar{t}), d') \in Q_{\mathcal{RN}}^J$ and $d' \in (A_{\mathcal{RN}}^J)^J$. By (19), $d = d'$ and thus, $d \in (\bigcirc \mathcal{RN})^J$.

Let $d \in (\exists \mathcal{RN}[U])^J$. Then, $U_i \in \tau(R)$ and there are different $t_1, \ldots, t_q \in R^J$ with $t_j[U_i] = d$, for all $i = 1, \ldots, q$. For each $t_j$, by (20), there must exist some element $t'_j \in \mathcal{RN}^J$ such that $t_j = t'_j[\tau(R)]$, for some $\forall \mathcal{RN} \subseteq \mathcal{R}$, while, by induction, $t_j[t'_j] \in \tau^J_1$ and $t'_j \in \tau^J_1$. Thus, $t'_j[U_i] = t_j[U_i] = d$ and, by (17), it then follows that $(t'_j, \bar{t}) \in (\mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1))^J$ while, by (13), we have $(t'_j, \bar{t}) \in (\mathcal{PATH}_{\mathcal{J}}(\tau(R), \tau(R)))^J$. Since $\mathcal{DLR}^\pm$ allows only for knowledge bases with a projection signature graph being a multtree, then,

$$\mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1) = \mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1)^J \circ \mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1)^J.$$

Thus, $(\bar{t}(\bar{t}), d) \in (\mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1))^J$ and, since $\bar{t}$ is injective, then $t_j \neq t_i$ when $l \neq j$. Thus, $d \in (\exists \mathcal{RN}[U])^J$.

Let $d \in (\exists \mathcal{U}_C \mathcal{R})^J$. Then, $t \in R^J_2$, $U_i \in \tau(R)$ and $\forall \bar{t} \in \tau(R)$, $d \in C^J_1$. By induction, $\forall \bar{t} \in \tau(R)$ and $d \in C^J_1$. As before, by (17), (18) and (20), we can show that $(\bar{t}(\bar{t}), d) \in (\mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1))^J$ and, since $\mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1)$ is functional, then $t \in (\sigma_{\mathcal{U}_C \mathcal{R}})^J$.

Let $d \in (\exists \mathcal{RN}[U])^J$. Then, $U_i \in \mathcal{R}$ and $\forall \bar{t} \in \mathcal{R}$, $d \in C^J_1$. By induction, $\forall \bar{t} \in \mathcal{R}$ and $d \in C^J_1$. As before, by (18) and (20), we can show that $(\bar{t}(\bar{t}), d) \in (\mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1))^J$ and thus $d \in (\exists \mathcal{U}_C \mathcal{R})^J$.

All the other cases can be proved in a similar way. We now show the converse direction.

Let $d \in (\bigcirc \mathcal{RN})^J$. Then, $d \in (A_{\mathcal{RN}}^J)^J$ and, by $\gamma_{lobj}(\mathcal{RN})$, there is a $d' \in \Delta^J$ s.t. $(d', \bar{t}) \in Q_{\mathcal{RN}}^J$ and $d' \in A_{\mathcal{RN}}^J$. By induction, $d' = t'(t')$ with $t' \in \mathcal{RN}^J$ and thus, $(d', \bar{t}) \in Q_{\mathcal{RN}}^J$ and, by (19), $t_{\mathcal{RN}}(t') = d$, i.e., $d \in (\bigcirc \mathcal{RN})^J$.

Let $d \in (\exists \mathcal{RN}[U])^J$. Then, $U_i \in \tau(\mathcal{R})$ and there are different $d_1, \ldots, d_q \in \Delta^J$ s.t. $(d_i, d) \in (\mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1))^J$ and $d_i \in \tau^J_1$, for $l = 1, \ldots, q$. By induction, each $d_i = t_i[t_i]$ and $t_i \in R^J_2$. Since $\bar{t}$ is injective, then $t_i \neq t_j$ for all $j, l = 1, \ldots, q$, $l \neq j$. We need to show that $t_i[U_i] = d$, for all $i = 1, \ldots, q$. By (20), there exists a $t'_i \in \mathcal{RN}^J$ such that $t_i = t'_i[\tau(R)]$, for some $\forall \mathcal{RN} \subseteq \mathcal{R}$ and, by (18), it holds that $(t'_i, \bar{t}_i) \in (\mathcal{PATH}_{\mathcal{J}}(\tau(R), \tau(R)))^J$. Since $(t'_i, \bar{t}_i) \in (\mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1))^J$ and $\mathcal{PATH}_{\mathcal{J}}$ is functional in $\mathcal{DLR}^\pm$, then $(t'_i, \bar{t}_i) \in (\mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1))^J$ and, by (17), $t'_i[U_i] = t_i[U_i] = d$.

Let $d \in (\exists \mathcal{U}_C \mathcal{R})^J$. Then, $d \in R^J_1$ and, by induction, $t \in R^J_2$. Let $t[U_i] = d$. We need to show that $d \in C^J_2$. As before, by (20) and (13), we have that $(d, \bar{t}) \in (\mathcal{PATH}_{\mathcal{J}}(\tau(R), \{U_i\}_1^1))^J$. Then $d \in C^J_1$ and, by induction, $d \in C^J_2$. 


Let \( \nu(t) \in (\mathbb{3}[U_1, \ldots, U_k]R)^{\mathcal{J}}. \) Then, there is \( d \in \Delta^\mathcal{J} \) s.t.
\[
(d, \nu(t)) \in (\text{path}_{\mathcal{J}}(\tau(R), \{U_1, \ldots, U_k\}))^{\mathcal{J}}
\]
and \( d \in R^{\mathcal{J}}. \) By induction, \( d = \nu(t') \) and \( t' \in R^\mathcal{J}. \) By (20), there is a tuple \( t'' \in RN^\mathcal{J} \) s.t. \( t' = t''[\tau(R)] \) and, by (18), \( (\nu(t''), \nu(t)) \in (\text{path}_{\mathcal{J}}(\tau(RN), \tau(R)))^{\mathcal{J}} \)
and thus, by (21), \( (\nu(t''), \nu(t)) \in (\text{path}_{\mathcal{J}}(\tau(RN), \{U_1, \ldots, U_k\}))^{\mathcal{J}} \) and thus \( t = t''[\{U_1, \ldots, U_k\}]. \) Since \( \{U_1, \ldots, U_k\} \subseteq \tau(R) \subseteq \tau(RN) \), then, \( t = t''[\{U_1, \ldots, U_k\}] = (t''[\tau(R)])[U_1, \ldots, U_k] = t'[U_1, \ldots, U_k] \), i.e., \( t \in (\mathbb{3}[U_1, \ldots, U_k]R)^\mathcal{J}. \)

To show that \( \mathcal{I} \models \mathcal{A}, \) notice that \( \mathcal{I} \) satisfies both concept assertions and individual assertions by construction. We need to show that \( \mathcal{I} \) satisfies also relation assertions. Let \( RN(t) \in \mathcal{A}, \) with \( t = \langle U_1:o_1, \ldots, U_n:o_n \rangle, \) then, since \( \mathcal{J} \)

satisfies \( \gamma(\mathcal{A}), \) and in particular axiom (7), then there exists \( d = \xi(t) \in A_{RN}^\mathcal{J}. \)

By (8), \( (d, o^\mathcal{J}) \in (\text{path}_{\mathcal{J}}(\tau(RN), \{U_1\}))^{\mathcal{J}} \) and, by (17), \( t^\mathcal{J} \in RN^\mathcal{J}. \) \( \square \)

As a direct consequence of the above theorem and the fact that \( \mathcal{DLR} \)
is a sublanguage of \( \mathcal{DLR}^\pm, \) we have that

**Corollary 5.** *Reasoning in \( \mathcal{DLR}^\pm \) is an \( \text{ExpTime}-\text{complete} \) problem.*

### 6 Implementation of a \( \mathcal{DLR}^\pm \) API

We have implemented the framework discussed in this paper. DLRtoOWL is a Java library fully implementing \( \mathcal{DLR}^\pm \) reasoning services. The library is based on the tool ANTLR4 to parse serialised input, and on OWLAPI4 for the OWL2 encoding. The system includes JFact, the Java version of the popular Fact++ reasoner. DLRtoOWL provides a Java \( \mathcal{DLR} \) API package to allow developers to create, manipulate, serialise, and reason with \( \mathcal{DLR}^\pm \) knowledge bases in their Java-based application, extending in a compatible way the standard OWL API with the \( \mathcal{DLR}^\pm \) TELL and ASK services.

During the development of this new library we strongly focused on performance. Since the OWL encoding is only possible if we have already built the \( \mathcal{ALCQI} \) projection signature multitree, in principle the program should perform two parsing rounds: one to create the multitree and the other one to generate the OWL mapping. We faced this issue using dynamic programming: during the first (and only) parsing round we store in a data structure each axiom that we want to translate in OWL and, after building the multitree, by the dynamic programming technique we build on-the-fly a Java class which generates the required axioms.

### 7 Conclusions

We have introduced the very expressive \( \mathcal{DLR}^\pm \) description logic, which extends \( \mathcal{DLR} \) with database oriented constraints. \( \mathcal{DLR}^\pm \) is expressive enough to cover directly and more thoroughly the EER, UML, and ORM conceptual data models,
among others. Although reasoning in $DLR^+$ is undecidable, we show that a simple syntactic constraint on KBs restores decidability. In fact, the resulting logic $DLR^\bar{\circ}$ has the same complexity (ExpTime-complete) as the basic $DLR$ language. In other words, handling database constraints does not increase the complexity of reasoning in the logic. To enhance the use and adoption of $DLR^\bar{\circ}$, we have developed an API that fully implements reasoning for this language, and maps input knowledge bases into OWL. Using a standard OWL reasoner, we are able to provide a variety of $DLR^\bar{\circ}$ reasoning services.

We plan to investigate the problem of query answering under $DLR^\bar{\circ}$ ontologies and to check whether the complexity for this problem can be lifted from known results in $DLR$ to $DLR^\bar{\circ}$.

References
1. Artale, A., Calvanese, D., Kontchakov, R., Ryzhikov, V., Zakharyaschev, M.: Reasoning over extended ER models. In: Proc. of the 26th Int. Conf. on Conceptual Modeling (ER’07). Lecture Notes in Computer Science, vol. 4801, pp. 277–292. Springer (2007)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press (2003)
3. Berardi, D., Calvanese, D., De Giacomo, G.: Reasoning on UML class diagrams. Artificial Intelligence 168(1–2), 70–118 (2005)
4. Calvanese, D., De Giacomo, G., Lenzerini, M.: Identification constraints and functional dependencies in description logics. In: Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI-01. pp. 155–160. Morgan Kaufmann (2001)
5. Calvanese, D., Fischl, W., Pichler, R., Sallinger, E., Simkus, M.: Capturing relational schemas and functional dependencies in RDFS. In: Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI). pp. 1003–1011. AAAI Press (2014)
6. Calvanese, D., Giacomo, G.D., Lenzerini, M.: Conjunctive query containment and answering under description logic constraints. ACM Trans. Comput. Logic 9(3), 22:1–22:31 (Jun 2008)
7. Chandra, A.K., Vardi, M.Y.: The implication problem for functional and inclusion dependencies is undecidable. SIAM Journal on Computing 14(3), 671–677 (1985)
8. Franconi, E., Mosca, A., Solomatkin, D.: ORM2: formalisation and encoding in OWL2. In: International Workshop on Fact-Oriented Modeling (ORM 2012). pp. 368–378 (2012)
9. Halpin, T., Morgan, T.: Information Modeling and Relational Databases. Morgan Kaufmann, 2nd edn. (2008)
10. Horrocks, I., Sattler, U., Tessaris, S., Tobies, S.: How to decide query containment under constraints using a description logic. In: 7th International Conference on Logic for Programming and Automated Reasoning (LPAR00), 2000. pp. 326–343 (2000)
11. Kanellakis, P.C.: Elements of relational database theory. In: Meyer, A., Nivat, M., Paterson, M., Perrin, D., van Leeuwen, J. (eds.) The Handbook of Theoretical Computer Science, vol. B, chap. 17, pp. 1075–1144. North Holland (1990)
12. Lukasiewicz, T., Cali, A., Gottlob, G.: A general datalog-based framework for tractable query answering over ontologies. Web Semantics: Science, Services and Agents on the World Wide Web 14(0) (2012)

13. Sportelli, F., Franconi, E.: Formalisation of ORM derivation rules and their mapping into OWL. In: ODBASE Conference 2016. pp. 827–843 (2016)

14. Toman, D., Weddell, G.E.: Applications and extensions of PTIME description logics with functional constraints. In: IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence. pp. 948–954 (2009)