A Decidable Very Expressive *n*-ary Description Logic for Database Applications

Abstract

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

1 Introduction

We introduce the new description logic $\mathcal{DLR}^+$extending the n-ary description logics $\mathcal{DLR}_{ifd}$ [Calvanese *et al.*, 2001], itself based on the description logic $\mathcal{DLR}$ [Horrocks *et al.*, 2000; Calvanese *et al.*, 2008], in order to capture more database oriented constraints. While $\mathcal{DLR}_{ifd}$ 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.

uld we cite the paper?

The added expressivity is motivated by the increasingly use of description logics as an abstract conceptual layer over relational databases, both to reason over such conceptual models during the database design phase (see, e.g., works as [Calvanese *et al.*, 1998; Berardi *et al.*, 2005; Artale *et al.*, 2007; Artale and Franconi, 2009; Toman and Weddell, 2009; Artale *et al.*, 2010; Franconi *et al.*, 2012]) and to answer ontology-mediated queries over databases (see, e.g, works as [Artale *et al.*, 2009; Franconi *et al.*, 2013]).

To describe the added expressivity of $\mathcal{DLR}^+$ we consider the scenario where the following verbs are modelled: a generic `TransitiveVerb` and two specific verbs as `Love` and `Kill`. Then, in $\mathcal{DLR}^+$ the following constraints that are common both in databases and in data model can be captured:

- Instances of relations are so called *attribute-labelled tuples*, i.e., tuples such that their components are identified by an attribute and not by a position in the tuple. Thus, the three verbs can be captured in $\mathcal{DLR}^+$ with the following relations:

  $$\texttt{TransitiveVerb}(\texttt{Agent}, \texttt{Patient}, \texttt{When}),$$
  $$\texttt{Love}(\texttt{Lover}, \texttt{Loved}, \texttt{When}, \texttt{How}),$$
  $$\texttt{Kill}(\texttt{Killer}, \texttt{Killed}, \texttt{When}, \texttt{Location}, \texttt{Tool}).$$

  A possible instance of the `Kill` event has the form:

  $$\langle \texttt{Killer}{:}\textit{John}, \texttt{Killed}{:}\textit{Mary}, \texttt{When} : \textit{23.11.1974}\ \texttt{Location}{:}\textit{NY}, \texttt{Tool}{:}\textit{gun} \rangle.$$

- *Renaming of attributes* is possible, e.g., to recover the positional semantics. In our example, we can rename the `TransitiveVerb` attributes into the positional attributes with the following:

  $$\texttt{Agent}, \texttt{Patient}, \texttt{When} \rightleftarrows \texttt{1}, \texttt{2}, \texttt{3}.$$

- *Relation projections* (with the same meaning of projections in databases) allow to form new relations by projecting a given relation on some of its attributes. For example, to capture that every triple of the form `Lover`, `Loved`, `When` is also a `TransitiveVerb` we can project the `Love` relation and add the axiom (inclusion dependency):

  $$\exists[\texttt{Lover}, \texttt{Loved}, \texttt{When}]\texttt{Love} \sqsubseteq \texttt{TransitiveVerb}.$$

- *Functional dependencies* and, in particular, *multiple-attribute keys* can be expressed via the multiple-attribute cardinalities construct. Therefore, e.g., to express that the pair `Killer`, `Killed` is a key for the `Kill` relation can be captured by the following axiom:

  $$\exists[\texttt{Killer}, \texttt{Killed}]\texttt{Kill} \sqsubseteq \exists^{\leqslant 1}[\texttt{Killer}, \texttt{Killed}]\texttt{Kill}.$$

- *Local and global objectification* (the last one is also known as reification) allow to create a new concept starting from a relation where relation tuples are identified either by a unique global identifier or by an identifier which is unique only within the interpretation of

$$
\begin{array}{rcl}
C & \rightarrow & \top \mid \bot \mid CN \mid \neg C \mid C_1 \sqcap C_2 \mid C_1 \sqcup C_2 \mid \exists^{\leqslant q}[U_i]R \mid \circledcirc R \mid \odot RN \\
R & \rightarrow & RN \mid R_1 \backslash R_2 \mid R_1 \sqcap R_2 \mid R_1 \sqcup R_2 \mid \sigma_{U_i:C} R \mid \exists^{\leqslant q}[U_1, \ldots, U_k]R \\
\varphi & \rightarrow & C_1 \sqsubseteq C_2 \mid R_1 \sqsubseteq R_2 \mid CN(o) \mid RN(U_1{:}o_1, \ldots, U_n{:}o_n) \mid o_1 = o_2 \mid o_1 \neq o_2 \\
\vartheta & \rightarrow & U_1 \rightleftarrows U_2
\end{array}
$$

Figure 1: Syntax of $\mathcal{DLR}^+$.

a relation, depending whether global or local objectification, respectively, is used. While global objectification corresponds to the well known reification construction, we show an example of the local objectification. Consider the relations $\texttt{OwnsCar}(\texttt{Name}, \texttt{Car})$ and $\texttt{DrivesCar}(\texttt{Name}, \texttt{Car})$ and assume that to drive a car you should also own it, i.e., $\texttt{DrivesCar} \sqsubseteq \texttt{OwnsCar}$. By using classical reification to capture the events of driving and owing a car we logically imply a subsumption between the two reified events. On the other hand, if we use local objectification to generate the two events the subsumption is no more implied and actually the two events can also be disjoint (as they should be considered).

Besides a TBox, $\mathcal{DLR}^+$ also expresses factual knowledge via an ABox where objects and labelled-tuples are asserted to belong to concepts and relations, respectively.

While $\mathcal{DLR}^+$ is undecidable due to the possibility to express arbitrary functional and inclusion dependencies [Mitchell, 1983; Chandra and Vardi, 1985] we show how a simple syntactic condition on the appearance of projections in the knowledge base makes the language decidable. The result of this restriction is a new language called $\mathcal{DLR}^{\pm}$. We prove that $\mathcal{DLR}^{\pm}$, 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. Furthermore, $\mathcal{DLR}^{\pm}$ is able to correctly express the UML/EER conceptual data models as introduced in [Berardi *et al.*, 2005; Artale *et al.*, 2007] and the ORM conceptual data model as introduced in [Franconi and Mosca, 2013].

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

We first define 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 τ 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 $n \geqslant 2$.

The syntax of concepts C, relations R, formulas φ, and attribute renaming axioms ϑ 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 $2 \leqslant k < \text{ARITY}(R)$. The *arity* of a relation R is the number of the attributes in its signature; i.e., $\text{ARITY}(R) = |\tau(R)|$, where we extend the signature function τ to arbitrary relations as specified in Figure 2. Notice that, while global objectification ($\circledcirc R$) can be applied to arbitrary relations, local ones ($\odot RN$) can be applied just to relation names. We use the shortcut $\exists[U_1, \ldots, U_k]R$ for $\exists^{\geqslant 1}[U_1, \ldots, U_k]R$ for $k \geqslant 1$.

A $\mathcal{DLR}^+$ *TBox* $\mathcal{T}$ is a finite set of *concept inclusion* axioms of the form $C_1 \sqsubseteq C_2$ and *relation inclusion* axioms of the form $R_1 \sqsubseteq R_2$. We use $X_1 \equiv X_2$ as a shortcut for the two axioms $X_1 \sqsubseteq X_2$ and $X_2 \sqsubseteq X_1$. A $\mathcal{DLR}^+$ *ABox* $\mathcal{A}$ is a finite set of *concept instance* axioms of the form $CN(o)$, *relation instance* axioms of the form $RN(U_1{:}o_1, \ldots, U_n{:}o_n)$, and *same/distinct individual* axioms of the form $o_1 = o_2$ and $o_1 \neq o_2$, with $o_i \in \mathcal{O}$. Restricting ABox axioms to concept and relation names only does not affect the expressivity of $\mathcal{DLR}^+$ due to the availability of TBox axioms.

$$
\begin{array}{rll}
\tau(R_1 \backslash R_2) = & \tau(R_1) & \text{if } \tau(R_1) = \tau(R_2) \\
\tau(R_1 \sqcap R_2) = & \tau(R_1) & \text{if } \tau(R_1) = \tau(R_2) \\
\tau(R_1 \sqcup R_2) = & \tau(R_1) & \text{if } \tau(R_1) = \tau(R_2) \\
\tau(\sigma_{U_i:C} R) = & \tau(R) & \text{if } U_i \in \tau(R) \\
\tau(\exists^{\leqslant q}[U_1, \ldots, U_k]R) = & \{U_1, \ldots, U_k\} & \text{if } \{U_1, \ldots, U_k\} \subset \tau(R) \\
\tau(R) = & \varnothing & \text{otherwise}
\end{array}
$$

Figure 2: The signature of $\mathcal{DLR}^+$ relations.

A set of renaming axioms forms a *renaming schema*, which induces an equivalence relation $(\rightleftarrows, \mathcal{U})$ over the attributes $\mathcal{U}$, providing a partition of $\mathcal{U}$ into equivalence classes each one representing the alternative ways to name attributes. We write $[U]_{\Re}$ to denote the equivalence class of the attribute U w.r.t. the equivalence relation $(\rightleftarrows, \mathcal{U})$. We allow only *well founded* renaming schemas; that is, schemas such that each equivalence class $[U]_{\Re}$ in the induced equivalence relation never contains two attributes from the same relation signature. We use the shortcut $U_1 \ldots U_n \rightleftarrows U'_1 \ldots U'_n$ to group many renaming axioms with the obvious meaning that $U_i \rightleftarrows U'_i$, for all $i = 1, \ldots, n$.

The renaming schema reconciles the named attribute and the positional perspectives on relations (see, e.g., [Kanellakis, 1990]). They are crucial when expressing both inclusion axioms and set operators ($\sqcap$, $\sqcup$, $\backslash$) between relations, which make sense only over *union compatible* relations. Two relations R_1, R_2 are union compatible if their signatures are equal up to the attribute renaming induced by the renaming schema $\Re$, namely, $\tau(R_1) = \{U_1, \ldots, U_n\}$ and $\tau(R_2) = \{V_1, \ldots, V_n\}$ have the same arity n and $[U_i]_{\Re} = [V_i]_{\Re}$ for each $1 \leqslant i \leqslant n$. Notice that through the renaming schema, relations can use just local attribute names that can then be renamed when composing relations. Also note that it is obviously possible for the same attribute to appear in the signature of different relations.

A $\mathcal{DLR}^+$ knowledge base (KB) $\mathcal{KB} = (\mathcal{T}, \mathcal{A}, \Re)$ is composed by a TBox $\mathcal{T}$, an ABox $\mathcal{A}$, and a renaming schema $\Re$.

Example 1. *Consider the relation names R_1, R_2 where $\tau(R_1) = \{W_1, W_2, W_3, W_4\}$, $\tau(R_2) = \{V_1, V_2, V_3, V_4, V_5\}$, and the renaming axiom $W_1W_2W_3 \rightleftarrows V_3V_4V_5$. The TBox*

$\mathcal{T}_{\mathsf{exa}}$ *consists of the axioms:*

$$R_2 \sqsubseteq R_1 \tag{1}$$

$$\exists[W_1, W_2]R_1 \sqsubseteq \exists^{\leqslant 1}[W_1, W_2]R_1 \tag{2}$$

$$\exists[V_3, V_4]R_2 \sqsubseteq \exists^{\leqslant 1}[V_3, V_4](\exists[V_3, V_4, V_5]R_2) \tag{3}$$

$$\exists[W_1, W_2, W_3]R_1 \sqsubseteq \exists[V_3, V_4, V_5]R_2. \tag{4}$$

Intuitively, the axiom (2) *expresses that* W_1, W_2 *form a multi-attribute key for* R_1*;* (3) *introduces a functional dependency in the relation* R_2 *where the attribute* V_5 *is functionally dependent from attributes* V_3, V_4*, and* (4) *states an inclusion between two projections of the relation names* R_1, R_2 *based on the renaming schema axiom.*

The semantics of $\mathcal{DLR}^+$ uses of the notion of *labelled tuples* over a domain Δ: a *$\mathcal{U}$-labelled tuple over Δ* (or *tuple* for short) is a function $t\colon \mathcal{U} \to \Delta$. For $U \in \mathcal{U}$, we write $t[U]$ to refer to the domain element $d \in \Delta$ labelled by U, if the function t is defined for U—that is, if the attribute U is a label of the tuple t. Given $d_1, \ldots, d_n \in \Delta$, the expression $\langle U_1\colon d_1, \ldots, U_n\colon d_n\rangle$ stands for the tuple t such that $t[U_i] = d_i$, for $1 \leqslant 1 \leqslant n$. The *projection* of the tuple t over the attributes $U_1, \ldots, U_k$ (i.e., the function t restricted to be undefined for the labels not in $U_1, \ldots, U_k$) is denoted by $t[U_1, \ldots, U_k]$. The relation signature function τ can be applied also to labelled tuples to obtain the set of labels on which the tuple is defined. $T_\Delta(\mathcal{U})$ denotes the set of all $\mathcal{U}$-labelled tuples over Δ.

A $\mathcal{DLR}^+$ *interpretation* is a tuple $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}}, \rho, \imath, L)$ consisting of a nonempty *domain* Δ, an *interpretation function* $\cdot^{\mathcal{I}}$, a *renaming function* ρ, a *global objectification function* $\imath$, and a family L containing one *local objectification function* ℓ_{RN_i} for each named relation $RN_i \in \mathcal{R}$.

The renaming function ρ is a total function $\rho : \mathcal{U} \to \mathcal{U}$ representing a canonical renaming for all attributes. We use $\rho(\{U_1, \ldots, U_k\})$ to denote $\{\rho(U_1), \ldots, \rho(U_k)\}$. The global objectification function is an injective function, $\imath : T_\Delta(\mathcal{U}) \to \Delta$, associating a *unique* global identifier to each tuple. The local objectification functions, $\ell_{RN_i} : T_\Delta(\mathcal{U}) \to \Delta$, 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 $\cdot^{\mathcal{I}}$ assigns a domain element to each individual $o^{\mathcal{I}} \in \Delta$, a set of domain elements to each concept name $CN^{\mathcal{I}} \subseteq \Delta$, and a set of $\mathcal{U}$-labelled tuples over Δ to each relation name conforming with its signature and to the renaming function $RN^{\mathcal{I}} \subseteq T_\Delta(\{\rho(U) \mid U \in \tau(RN)\})$. Note that the unique name assumption is not enforced. The interpretation function $\cdot^{\mathcal{I}}$ is unambiguously extended over concept and relation expressions as specified in Figure 3.

The interpretation $\mathcal{I}$ satisfies the concept inclusion axiom $C_1 \sqsubseteq C_2$ if $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$, and the relation inclusion axiom $R_1 \sqsubseteq R_2$ if $R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$. It satisfies the concept instance axiom $CN(o)$ if $o^{\mathcal{I}} \in CN^{\mathcal{I}}$, the relation instance axiom $RN(U_1{:}o_1, \ldots, U_n{:}o_n)$ if $\langle \rho(U_1)\colon o_1^{\mathcal{I}}, \ldots, \rho(U_n)\colon o_n^{\mathcal{I}}\rangle \in RN^{\mathcal{I}}$, and the axioms $o_1 = o_2$ and $o_1 \neq o_2$ if $o_1^{\mathcal{I}} = o_2^{\mathcal{I}}$, and $o_1^{\mathcal{I}} \neq o_2^{\mathcal{I}}$, respectively. $\mathcal{I}$ satisfies a renaming schema $\Re$ if for every $U, V \in \mathcal{U}$, (i) $\rho(U) \in [U]_\Re$, and (ii) if $V \in [U]_\Re$, then $\rho(U) = \rho(V)$. $\mathcal{I}$ is a *model* of the KB $(\mathcal{T}, \mathcal{A}, \Re)$ if it satisfies all the axioms in the TBox $\mathcal{T}$ and in the ABox $\mathcal{A}$, and the renaming schema $\Re$.

$$\begin{aligned}
\top^{\mathcal{I}} &= \Delta \\
\bot^{\mathcal{I}} &= \varnothing \\
(\neg C)^{\mathcal{I}} &= \top^{\mathcal{I}} \backslash C^{\mathcal{I}} \\
(C_1 \sqcap C_2)^{\mathcal{I}} &= C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}} \\
(C_1 \sqcup C_2)^{\mathcal{I}} &= C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}} \\
(\exists^{\leqslant q}[U_i]R)^{\mathcal{I}} &= \{d \in \Delta \mid \left|\{t \in R^{\mathcal{I}} \mid t[\rho(U_i)] = d\}\right| \leqslant q\} \\
(\circledcirc R)^{\mathcal{I}} &= \{d \in \Delta \mid d = \imath(t) \wedge t \in R^{\mathcal{I}}\} \\
(\odot RN)^{\mathcal{I}} &= \{d \in \Delta \mid d = \ell_{RN}(t) \wedge t \in RN^{\mathcal{I}}\} \\
(R_1 \backslash R_2)^{\mathcal{I}} &= R_1^{\mathcal{I}} \backslash R_2^{\mathcal{I}} \\
(R_1 \sqcap R_2)^{\mathcal{I}} &= R_1^{\mathcal{I}} \cap R_2^{\mathcal{I}} \\
(R_1 \sqcup R_2)^{\mathcal{I}} &= \{t \in R_1^{\mathcal{I}} \cup R_2^{\mathcal{I}} \mid \rho(\tau(R_1)) = \rho(\tau(R_2))\} \\
(\sigma_{U_i:C} R)^{\mathcal{I}} &= \{t \in R^{\mathcal{I}} \mid t[\rho(U_i)] \in C^{\mathcal{I}}\} \\
(\exists^{\leqslant q}[U_1, \ldots, U_k]R)^{\mathcal{I}} &= \{\langle \rho(U_1) : d_1, \ldots, \rho(U_k) : d_k\rangle \in T_\Delta(\{\rho(U_1), \ldots, \rho(U_k)\}) \mid \\
&\quad \left|\{t \in R^{\mathcal{I}} \mid t[\rho(U_1)] = d_1, \ldots, t[\rho(U_k)] = d_k\}\right| \leqslant q\}
\end{aligned}$$

Figure 3: Semantics of $\mathcal{DLR}^+$ expressions.

KB satisfiability refers to the problem of deciding the existence of a model of a given KB; *concept satisfiability* (resp. *relation satisfiability*) is the problem of deciding whether there is a model of the KB that assigns a non-empty extension to a given concept (resp. relation); and *entailment* is to check whether the KB logically implies an axiom, that is, whether all models of the KB are also models of the axiom. For instance, from the TBox in Example 1 we can entail that V_3, V_4 are a key for the relation R_2:

$$\mathcal{T}_{\mathsf{exa}} \models \exists[V_3, V_4]R_2 \sqsubseteq \exists^{\leqslant 1}[V_3, V_4]R_2.$$

All the decision problems can be reduced to KB satisfiability.

Lemma 1. *In $\mathcal{DLR}^+$, concept and relation satisfiability and entailment are reducible to KB satisfiability.*

3 Expressiveness of $\mathcal{DLR}^+$

$\mathcal{DLR}^+$ is a very expressive description logic capable of asserting several kinds of constraints that are important in the context of relational databases, such as *inclusion dependencies* (inclusion axioms among arbitrary projections of relations), *equijoins*, *functional dependency* axioms, *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 $\tau(R_1) = \{U, U_1, \ldots, U_{n_1}\}$ and $\tau(R_2) = \{V, V_1, \ldots, V_{n_2}\}$; their equijoin over U and V is the relation $R = R_1 \underset{U=V}{\bowtie} R_2$ with signature $\tau(R) = \tau(R_1) \cup \tau(R_2) \backslash \{V\}$, which is expressed by the $\mathcal{DLR}^+$ axioms:

$$\begin{aligned}
&\exists[U, U_1, \ldots, U_{n_1}]R \equiv \sigma_{U:(\exists[U]R_1 \sqcap \exists[V]R_2)} R_1 \\
&\exists[V, V_1, \ldots, V_{n_2}]R \equiv \sigma_{V:(\exists[U]R_1 \sqcap \exists[V]R_2)} R_2 \\
&U \rightleftarrows V\ .
\end{aligned}$$

A *functional dependency* axiom $(R\colon U_1 \ldots U_j \to U)$ 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 $\mathcal{I}$ satisfies this functional dependency axiom if, for all tuples $s, t \in R^{\mathcal{I}}$, $s[U_1] = t[U_1], \ldots, s[U_j] = t[U_j]$ imply $s[U] = t[U]$. Functional dependency axioms are

also called *internal uniqueness* axioms [Halpin and Morgan, 2008]. Functional dependencies can be expressed in $\mathcal{DLR}^+$, assuming that $\{U_1, \ldots, U_j, U\} \sqsubseteq \tau(R)$, with the axiom:

$$\exists[U_1, \ldots, U_j]R \sqsubseteq \exists^{\leqslant 1}[U_1, \ldots, U_j](\exists[U_1, \ldots, U_j, U]R).$$

A special case of functional dependencies 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 $\{U_1 \ldots U_j\} \sqsubseteq \tau(R)$, with the axiom:

$$\exists[U_1, \ldots, U_j]R \sqsubseteq \exists^{\leqslant 1}[U_1, \ldots, U_j]R.$$

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 [Halpin and Morgan, 2008]. This can be expressed in $\mathcal{DLR}^+$ with the axioms:

$$R \equiv R_1 \underset{U^1=U^2}{\bowtie} \cdots \underset{U^{h-1}=U^h}{\bowtie} R_h$$
$$R : U_1^1, \ldots, U_{n_1}^1, \ldots, U_1^h, \ldots, U_{n_h}^h \rightarrow U^1$$

where $\tau(R_i) = \{U^i, U_1^i, \ldots, U_{n_i}^i\}, 1 \leqslant i \leqslant h$, and R with $\tau(R) = \{U^1, U_1^1, \ldots, U_{n_1}^1, \ldots, U_1^h, \ldots, U_{n_h}^h\}$ is a new relation name.

Identification axioms as defined in $\mathcal{DLR}_{ifd}$ [Calvanese *et al.*, 2001] (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:

$$[U^1]\sigma_{U_1:C}R_1 \downarrow \ldots \downarrow [U^h]\sigma_{U_h:C}R_h.$$

Path functional dependencies—as defined in the $\mathcal{CFD}$ family of description logics [Toman and Weddell, 2009]—can be expressed in $\mathcal{DLR}^+$ as identification axioms involving joined sequences of functional binary relations.

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 [Artale *et al.*, 2007] can be extended to deal with multi-attribute keys (by using identification axioms) and named roles in relations; the ORM mapping as introduced in [Franconi *et al.*, 2012; Franconi and Mosca, 2013; Sportelli and Franconi, 2016] 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 [Berardi *et al.*, 2005] can be fixed to deal properly with association classes (by using local objectification) and named roles in associations.

to say without nding?

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

Since a $\mathcal{DLR}^+$ KB can express both inclusion and functional dependencies, the entailment problem is undecidable [Chandra and Vardi, 1985]. 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.

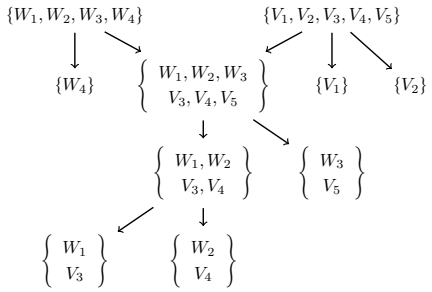

Figure 4: The projection signature graph of Example 1.

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

The *projection signature graph* is the directed acyclic graph $(\supset, \mathscr{T})$ whose sinks are the attribute singletons $\{U\}$. Given a set of attributes $\tau = \{U_1, \ldots, U_k\} \subseteq \mathcal{U}$, the *projection signature graph dominated by* τ, denoted as $\mathscr{T}_\tau$, is the sub-graph of $(\supset, \mathscr{T})$ containing all the nodes reachable from τ. Given two sets of attributes $\tau_1, \tau_2 \subseteq \mathcal{U}$, $\textsc{path}_{\mathscr{T}}(\tau_1, \tau_2)$ denotes the set of paths in $(\supset, \mathscr{T})$ between τ_1 and τ_2. Note that, $\textsc{path}_{\mathscr{T}}(\tau_1, \tau_2) = \varnothing$ both when a path does not exist and when $\tau_1 \subseteq \tau_2$. The notation $\textsc{child}_{\mathscr{T}}(\tau_1, \tau_2)$ means that τ_2 is a child of τ_1 in $(\supset, \mathscr{T})$. We now introduce $\mathcal{DLR}^{\pm}$ as follows.

Definition 1. *A $\mathcal{DLR}^{\pm}$ knowledge base is a $\mathcal{DLR}^+$ knowledge base that satisfies the following syntactic conditions:*

1. *the projection signature graph $(\supset, \mathscr{T})$ is a multitree: i.e., for every node $\tau \in \mathscr{T}$, the graph $\mathscr{T}_\tau$ is a tree; and*
2. *for every projection construct $\exists^{\leqslant q}[U_1, \ldots, U_k]R$ appearing in $\mathcal{T}$, if $q > 1$ then the length of the path $\textsc{path}_{\mathscr{T}}(\tau(R), \{U_1, \ldots, U_k\})$ is 1.*

Essentially, the conditions in $\mathcal{DLR}^{\pm}$ restrict $\mathcal{DLR}^+$ in the way that multiple projections of relations may appear in a knowledge base. In particular, observe that in $\mathcal{DLR}^{\pm}$ $\textsc{path}_{\mathscr{T}}$ is necessarily functional, due to the multitree restriction.

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 $\mathcal{DLR}^{\pm}$.

$\mathcal{DLR}$ is included in $\mathcal{DLR}^{\pm}$, 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 $\mathcal{DLR}^{\pm}$. Equijoins, external unique-

$$
\begin{aligned}
(\neg C)^\dagger &= \neg C^\dagger \\
(C_1 \sqcap C_2)^\dagger &= C_1^\dagger \sqcap C_2^\dagger \\
(C_1 \sqcup C_2)^\dagger &= C_1^\dagger \sqcup C_2^\dagger \\
(\exists^{\leqslant q}[U_i]R)^\dagger &= \exists^{\leqslant q} \left(\text{PATH}_{\mathscr{T}}(\tau(R), \{U_i\})^\dagger\right)^- . R^\dagger \\
(\circledcirc R)^\dagger &= R^\dagger \\
(\odot RN)^\dagger &= A^l_{RN} \\
\\
(R_1 \backslash R_2)^\dagger &= R_1^\dagger \sqcap \neg R_2^\dagger \\
(R_1 \sqcap R_2)^\dagger &= R_1^\dagger \sqcap R_2^\dagger \\
(R_1 \sqcup R_2)^\dagger &= R_1^\dagger \sqcup R_2^\dagger \\
(\sigma_{U_i:C} R)^\dagger &= R^\dagger \sqcap \forall \text{PATH}_{\mathscr{T}}(\tau(R), \{U_i\})^\dagger . C^\dagger \\
(\exists^{\leqslant q}[U_1, \ldots, U_k]R)^\dagger &= \exists^{\leqslant q} \left(\text{PATH}_{\mathscr{T}}(\tau(R), \{U_1, \ldots, U_k\})^\dagger\right)^- . R^\dagger
\end{aligned}
$$

Figure 5: The mapping for concept and relation expressions.

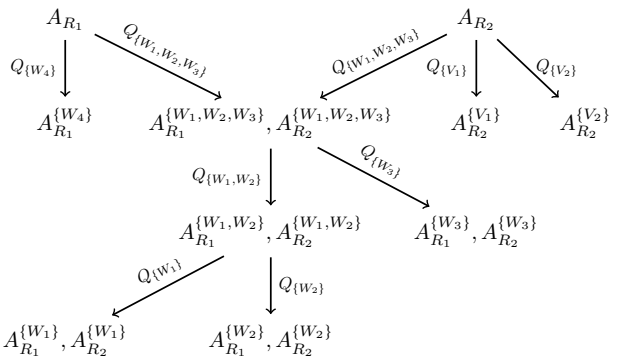

Figure 6: The $\mathcal{ALCQI}$ signature generated by $\mathcal{T}_{\mathsf{exa}}$.

ness axioms, and identification axioms introduce projections which share common attributes, thus violating the multitree restriction. However, it is still possible to express in $\mathcal{DLR}^{\pm}$ external uniqueness and identification axioms by encoding them as a special set of ABox axioms, as originally proposed in [Calvanese *et al.*, 2001]. Therefore, we can conclude that $\mathcal{DLR}_{ifd}$ [Calvanese *et al.*, 2001] extended with unary functional dependencies is included in $\mathcal{DLR}^{\pm}$, 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 [Calvanese *et al.*, 2001] leading to undecidability is ruled out.

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

5 Mapping $\mathcal{DLR}^{\pm}$ to $\mathcal{ALCQI}$

We show that reasoning in $\mathcal{DLR}^{\pm}$ is EXPTIME-complete, using the EXPTIME-completeness of $\mathcal{ALCQI}$ [Baader *et al.*, 2003], by providing a mapping from $\mathcal{DLR}^{\pm}$ KBs to $\mathcal{ALCQI}$ KBs, and by observing that $\mathcal{ALCQI}$ is directly expressible in $\mathcal{DLR}$. We adapt and extend the mapping presented for $\mathcal{DLR}$ in [Calvanese *et al.*, 2008], with the modifications proposed by [Horrocks *et al.*, 2000] to deal with ABoxes without the unique name assumption.

In the following $(S_1 \circ \ldots \circ S_n)^-$ stands for $S_n^- \circ \ldots \circ S_1^-$, $\exists^{\leqslant 1} S_1 \circ \ldots \circ S_n . C$ stands for $\exists^{\leqslant 1} S_1 . \ldots . \exists^{\leqslant 1} S_n . C$, and $\forall S_1 \circ \ldots \circ S_n . C$ for $\forall S_1 . \ldots . \forall S_n . C$. The concept expressions $\exists^{\leqslant 1} \bot . C$ and $\forall \bot . C$ stand for the $\bot$ concept. For a relation instance axiom of the form $RN(U_1 : o_1, \ldots, U_n : o_n)$ we use the shortcut $RN(t)$, with $t = \langle U_1 : o_1, \ldots, U_n : o_n \rangle$ a *relation instance*, i.e., a $\mathcal{U}$-labelled tuple over $\mathcal{O}$.

Let $\mathcal{KB} = (\mathcal{T}, \mathcal{A}, \Re)$ be a $\mathcal{DLR}^{\pm}$ KB. We first pre-process $\mathcal{KB}$ by transforming it into a logically equivalent one as follows: for each equivalence class $[U]_{\Re}$ a single *canonical* representative of the class is chosen, and $\mathcal{KB}$ is consistently rewritten by substituting each attribute with its canonical representative. After this rewriting, the renaming schema does not play any role in the mapping.

We first introduce a mapping function $\cdot^\dagger$ from $\mathcal{DLR}^{\pm}$ concepts and relations to $\mathcal{ALCQI}$ concepts. The function $\cdot^\dagger$ maps each concept name CN and each relation name RN appearing in the $\mathcal{DLR}^{\pm}$ KB to an $\mathcal{ALCQI}$ concept name CN and A_{RN}, respectively. The latter is the global reification of RN. For each relation name RN, the $\mathcal{ALCQI}$ signature also includes a concept name A^l_{RN} and a role name Q_{RN} that captures local objectification. The mapping $\cdot^\dagger$ is extended to concept and relation expressions as in Figure 5.

The mapping crucially uses the projection signature graph structure to map projections and selections, by accessing paths in the projection signature $\mathscr{T}$ associated to the $\mathcal{DLR}^{\pm}$ KB. If there is a path $\text{PATH}_{\mathscr{T}}(\tau, \tau') = \tau, \tau_1, \ldots, \tau_n, \tau'$ from τ to τ' in $\mathscr{T}$, then its mapping is the $\mathcal{ALCQI}$ role chain expression using role names Q_{τ_i}:

$$\text{PATH}_{\mathscr{T}}(\tau, \tau')^\dagger = Q_{\tau_1} \circ \ldots \circ Q_{\tau_n} \circ Q_{\tau'}.$$

with $\text{PATH}_{\mathscr{T}}(\tau, \tau')^\dagger = \bot$ if $\text{PATH}_{\mathscr{T}}(\tau, \tau') = \varnothing$.

We also include a concept name $A^{\tau_i}_{RN}$ for each projected signature τ_i in the projection signature graph dominated by $\tau(RN)$, $\tau_i \in \mathscr{T}_{\tau(RN)}$ to capture global reifications of the projections of RN. Note that $A^{\tau(RN)}_{RN}$ coincides with A_{RN}.

Intuitively, the mapping reifies each node in the projection signature graph: the target $\mathcal{ALCQI}$ signature of Example 1 is partially presented in Fig. 6, together with the projection signature graph. Each node is labelled with the corresponding global reification concept ($A^{\tau_j}_{R_i}$), for each $R_i \in \mathcal{R}$ and each projected signature τ_j in the projection signature graph dominated by $\tau(R_i)$, while the edges are labelled by the roles (Q_{τ_i}) needed for the reification.

Recall that $\mathcal{DLR}^{\pm}$ restricts to $q = 1$ the cardinalities on any path of length greater than 1. Thus, we remain within $\mathcal{ALCQI}$ when the mapping applies to cardinalities. If we need to express e.g. the cardinality constraint $\exists^{\leqslant q}[U_i]R$ with $q > 1$, then U_i should not be mentioned in any other projection of the relation R in such a way that $|\text{PATH}_{\mathscr{T}}(\tau(R), \{U_i\})| = 1$.

In order to explain the need for the path function in the mapping, notice that a relation is reified according to the decomposition dictated by the projection signature graph it dominates. Thus, to access an attribute U_j of a relation R_i it is necessary to follow the path through the projections that

$$\gamma(\mathcal{A}) = \{CN^\dagger(o) \mid CN(o) \in \mathcal{A}\} \cup \quad (5)$$
$$\{o_1 \neq o_2 \mid o_1 \neq o_2 \in \mathcal{A}\} \cup \{o_1 = o_2 \mid o_1 = o_2 \in \mathcal{A}\} \cup \quad (6)$$
$$\{A^{\tau_i}_{RN}(\xi(t[\tau_i])) \mid RN(t) \in \mathcal{A} \text{ and } \tau_i \in \mathscr{T}_{\tau(RN)}\} \cup \quad (7)$$
$$\{Q_{\tau_j}(\xi(t[\tau_i]), \xi(t[\tau_j])) \mid \text{CHILD}_{\mathscr{T}}(\tau_i, \tau_j)\} \quad (8)$$
$$\{Q_o(o) \mid o \in \mathcal{O}\} \cup \quad (9)$$
$$\{Q_t(o_1), Q_t \sqsubseteq C_t \mid t = \langle U_1{:}o_1, \ldots, U_n{:}o_n\rangle \text{ occurs in } \mathcal{A}\}, \quad (10)$$

Figure 7: The mapping $\gamma(\mathcal{A})$

use that attribute. This path is a role chain from the signature of the relation (the root) to the attribute as returned by the $\text{PATH}_{\mathscr{T}}(\tau(R_i), U_i)$ function. For example, in Figure 6 in order to access the attribute U_4 of the relation R_3 in the expression $(\sigma_{U_4:C} R_3)$, the path $\text{PATH}_{\mathscr{T}}(\tau(R_3), \{U_4\})^\dagger$ is equal to the role chain $Q_{\{U_3,U_4,U_5\}} \circ Q_{\{U_3,U_4\}} \circ Q_{\{U_4\}}$, so that $(\sigma_{U_4:C} R_3)^\dagger = A_{R_3} \sqcap \forall Q_{\{U_3,U_4,U_5\}} \circ Q_{\{U_3,U_4\}} \circ Q_{\{U_4\}}.C$. Similar considerations can be done when mapping cardinalities over relation projections.

Let $\mathcal{KB} = (\mathcal{T}, \mathcal{A})$ be a $\mathcal{DLR}^\pm$ KB with signature $(\mathcal{C}, \mathcal{R}, \mathcal{U}, \tau)$. The mapping $\gamma(\mathcal{KB}) = (\gamma(\mathcal{T}), \gamma(\mathcal{A}))$ defines the $\mathcal{ALCQI}$ KB:

$$\gamma(\mathcal{T}) = \gamma_{dsj} \cup \bigcup_{RN \in \mathcal{R}} \gamma_{rel}(RN) \cup \bigcup_{RN \in \mathcal{R}} \gamma_{lobj}(RN) \cup \bigcup_{C_1 \sqsubseteq C_2 \in \mathcal{KB}} C_1^\dagger \sqsubseteq C_2^\dagger \cup \bigcup_{R_1 \sqsubseteq R_2 \in \mathcal{KB}} R_1^\dagger \sqsubseteq R_2^\dagger,$$

where

$$\gamma_{dsj} = \{A^{\tau_i}_{RN_1} \sqsubseteq \neg A^{\tau_j}_{RN_2} \mid RN_1, RN_2 \in \mathcal{R}, \tau_i, \tau_j \in \mathscr{T}, |\tau_i| \geqslant 2, |\tau_j| \geqslant 2, \tau_i \neq \tau_j\}$$

$$\gamma_{rel}(RN) = \bigcup_{\tau_i \in \mathscr{T}_{\tau(RN)}} \bigcup_{\text{CHILD}_{\mathscr{T}}(\tau_i, \tau_j)} \{A^{\tau_i}_{RN} \sqsubseteq \exists Q_{\tau_j}.A^{\tau_j}_{RN},\ \exists^{\geqslant 2} Q_{\tau_j}.\top \sqsubseteq \bot\}$$

$$\gamma_{lobj}(RN) = \{A_{RN} \sqsubseteq \exists Q_{RN}.A^l_{RN},\ \exists^{\geqslant 2} Q_{RN}.\top \sqsubseteq \bot, A^l_{RN} \sqsubseteq \exists Q^-_{RN}.A_{RN},\ \exists^{\geqslant 2} Q^-_{RN}.\top \sqsubseteq \bot\}.$$

Intuitively, γ_{dsj} ensures that relations with different signatures are disjoint, thus, e.g., enforcing the union compatibility. The axioms in γ_{rel} introduce classical reification axioms for each relation and its relevant projections. The axioms in γ_{lobj} make sure that each local objectification differs form the global one.

To translate the ABox, we first map each individual o in the $\mathcal{DLR}^\pm$ ABox to an $\mathcal{ALCQI}$ individual o. Each relation instance occurring in $\mathcal{A}$ is mapped via an injective function ξ to a distinct individual. That is, $\xi : T_{\mathcal{O}}(\mathcal{U}) \to \mathcal{O}_{\mathcal{ALCQI}}$, with $\mathcal{O}_{\mathcal{ALCQI}} = \mathcal{O} \cup \mathcal{O}^t$ being the set of individuals in $\gamma(\mathcal{KB})$, $\mathcal{O} \cap \mathcal{O}^t = \varnothing$ and

$$\xi(t) = \begin{cases} o \in \mathcal{O}, & \text{if } t = \langle U{:}o\rangle \\ o \in \mathcal{O}^t, & \text{otherwise.} \end{cases}$$

Following [Horrocks *et al.*, 2000], the mapping $\gamma(\mathcal{A})$ in Figure 7 introduces a new concept name Q_o for each individual in $o \in \mathcal{O}$ and a new concept name Q_t for each relation instance occurring in $\mathcal{A}$, where C_t stands for the concept

$$\exists^{\leqslant 1}\big(\text{PATH}_{\mathscr{T}}(\tau(t), \{U_1\})^\dagger\big)^-.\ \exists\big(\text{PATH}_{\mathscr{T}}(\tau(t), \{U_2\})^\dagger\big).Q_{o_2} \sqcap \ldots \sqcap \exists\big(\text{PATH}_{\mathscr{T}}(\tau(t), \{U_n\})^\dagger\big).Q_{o_n}\ .$$

Intuitively, (7) and (8) reify each relation instance occurring in $\mathcal{A}$ using the projection signature of the relation instance itself. The formulas (9)-(10) guarantee that there is exactly one $\mathcal{ALCQI}$ individual reifying a given relation instance. Clearly, the size of $\gamma(\mathcal{KB})$ is polynomial in the size of $\mathcal{KB}$ under the same coding of the numerical parameters.
We are able to state our main results.

Theorem 1. *A $\mathcal{DLR}^\pm$ knowledge base $\mathcal{KB}$ is satisfiable iff the $\mathcal{ALCQI}$ knowledge base $\gamma(\mathcal{KB})$ is satisfiable.*

As a direct consequence of this theorem and the fact that $\mathcal{DLR}$ is a sublanguage of $\mathcal{DLR}^\pm$, we obtain the following.

Corollary 1. *Reasoning in $\mathcal{DLR}^\pm$ is* EXPTIME*-complete.*

6 Implementation of a $\mathcal{DLR}^\pm$ API

We have implemented the framework discussed in this paper. DLRtoOWL is a Java-based library which fully implements $\mathcal{DLR}^\pm$ language services. The library is based on the tool ANTLR4 to parse serialised input provided by the user, and on OWLAPI4 for the OWL2 encoding. The final user could define its own knowledge base in a text file, or via a TELL interface, on which an algorithm constructing the multitree is applied. After that, the tool performs the encoding into OWL. The system includes JFact, the Java-based version of the popular Fact++ reasoner. DLRtoOWL provides the most used reasoning services (hierarchy, satisfiability, entailment, etc.) via a ASK interface. In DLRtoOWL we also provide a Java DLRApi package to allow developers to create, manipulate and serialise $\mathcal{DLR}^\pm$ knowledge bases in their Java-based application, extending in a compatible way the standard OWL API.

During development we strongly focused on performance. Since the OWL encoding is only possible if we have already built the multitree, ideally the program should performs two parsing rounds: one for the creating of the multitree and the other one for 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 tree, 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 description logic $\mathcal{DLR}^+$, which extends $\mathcal{DLR}$ with constructors capable of dealing with database oriented constraints. In particular, this logic is expressive enough to cover directly and more thoroughly the EER, UML, and ORM conceptual data models, among others.

Although reasoning in $\mathcal{DLR}^+$ is undecidable, we show that a simple syntactic constraint on KBs restores decidability. In fact, the resulting logic $\mathcal{DLR}^\pm$ has the same complexity (EXPTIME-complete) as the basic $\mathcal{DLR}$ language. In other words, handling database constraints does not increase the complexity of reasoning in these logics.

To enhance the use and adoption of $\mathcal{DLR}^\pm$, we have developed an API that fully implements this language, and maps input KBs into OWL. Using a standard OWL reasoner, we are able to provide a variety of $\mathcal{DLR}^\pm$ reasoning services as well.

Something about future work?

References

[Artale and Franconi, 2009] Alessandro Artale and Enrico Franconi. Foundations of temporal conceptual data models. In *Conceptual Modeling: Foundations and Applications*, volume 5600 of *Lecture Notes in Computer Science*, pages 10–35. Springer, 2009.

[Artale *et al.*, 2007] A. Artale, D. Calvanese, R. Kontchakov, V. Ryzhikov, and M. Zakharyaschev. Reasoning over extended ER models. In *Proc. of the 26th Int. Conf. on Conceptual Modeling (ER'07)*, volume 4801 of *Lecture Notes in Computer Science*, pages 277–292. Springer, 2007.

[Artale *et al.*, 2009] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev. The DL-Lite family and relations. *Journal of Artificial Intelligence Research*, 36:1–69, 2009.

[Artale *et al.*, 2010] A. Artale, D. Calvanese, and A. Ibáñez-García. Full satisfiability of UML class diagrams. In *Proc. of the 29th Int. Conf. on Conceptual Modeling (ER'10)*, volume 4801 of *Lecture Notes in Computer Science*, 2010.

[Baader *et al.*, 2003] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2003.

[Berardi *et al.*, 2005] D. Berardi, D. Calvanese, and G. De Giacomo. Reasoning on UML class diagrams. *Artificial Intelligence*, 168(1–2):70–118, 2005.

[Calvanese *et al.*, 1998] D. Calvanese, M. Lenzerini, and D. Nardi. Description logics for conceptual data modeling. In J. Chomicki and G. Saake, editors, *Logics for Databases and Information Systems*. Kluwer, 1998.

[Calvanese *et al.*, 2001] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Identification constraints and functional dependencies in description logics. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI-01*, pages 155–160. Morgan Kaufmann, 2001.

[Calvanese *et al.*, 2008] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. Conjunctive query containment and answering under description logic constraints. *ACM Trans. Comput. Logic*, 9(3):22:1–22:31, June 2008.

[Chandra and Vardi, 1985] Ashok K. Chandra and Moshe Y. Vardi. The implication problem for functional and inclusion dependencies is undecidable. *SIAM Journal on Compututing*, 14(3):671–677, 1985.

[Franconi and Mosca, 2013] Enrico Franconi and Alessandro Mosca. Towards a core ORM2 language (research note). In *International Workshop on Fact-Oriented Modeling (ORM 2013)*, volume 8186 of *Lecture Notes in Computer Science*, pages 448–456. Springer, 2013.

[Franconi *et al.*, 2012] Enrico Franconi, Alessandro Mosca, and Dmitry Solomakhin. ORM2: formalisation and encoding in OWL2. In *International Workshop on Fact-Oriented Modeling (ORM 2012)*, pages 368–378, 2012.

[Franconi *et al.*, 2013] Enrico Franconi, Volha Kerhet, and Nhung Ngo. Exact query reformulation over databases with first-order and description logics ontologies. *J. Artif. Intell. Res. (JAIR)*, 48:885–922, 2013.

[Halpin and Morgan, 2008] Terry Halpin and Tony Morgan. *Information Modeling and Relational Databases*. Morgan Kaufmann, 2nd edition, 2008.

[Horrocks *et al.*, 2000] Ian Horrocks, Ulrike Sattler, Sergio Tessaris, and Stephan Tobies. How to decide query containment under constraints using a description logic. In *7th International Conference on Logic for Programming and Automated Reasoning (LPAR00), 2000*, pages 326–343, 2000.

[Kanellakis, 1990] Paris C. Kanellakis. Elements of relational database theory. In A.R. Meyer, M. Nivat, M.S. Paterson, D. Perrin, and J. van Leeuwen, editors, *The Handbook of Theoretical Computer Science*, volume B, chapter 17, pages 1075–1144. North Holland, 1990.

[Mitchell, 1983] John C. Mitchell. The implication problem for functional and inclusion dependencies. *Information and Control*, 56(3):154–173, 1983.

[Sportelli and Franconi, 2016] Francesco Sportelli and Enrico Franconi. Formalisation of ORM derivation rules and their mapping into OWL. In *ODBASE Conference 2016*, pages 827–843, 2016.

[Toman and Weddell, 2009] David Toman and Grant E. Weddell. Applications and extensions of PTIME description logics with functional constraints. In *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 948–954, 2009.