Path-Based Identification Constraints in Description Logics

Diego Calvanese[1], Giuseppe De Giacomo[2], Domenico Lembo[2], Maurizio Lenzerini[2], Riccardo Rosati[2]

[1] Faculty of Computer Science
Free University of Bozen-Bolzano
Piazza Domenicani 3
I-39100 Bolzano, Italy
calvanese@inf.unibz.it

[2] Dip. di Informatica e Sistemistica
SAPIENZA Università di Roma
Via Ariosto 25
I-00185 Roma, Italy
lastname@dis.uniroma1.it

Abstract

In spite of the importance of identification mechanisms in ontology engineering, the Description Logics at the basis of current reasoners do not include modeling features for expressing identification constraints. In this paper, we consider a powerful class of identification constraints, which allow for using roles, inverses, and paths, thus capturing sophisticated forms of identifications often needed in real-world applications. We show that, when used with no limitations, such path-based identification constraints are problematic with respect to effectiveness/efficiency of reasoning. We then propose a restricted form of these constraints, called *local*, requiring that at least one of the component paths of the concept identifier is a direct property of the concept. We argue that such a restriction is not a severe limitation in practice, and we show that local path-based identification constraints do not increase the complexity of reasoning both in very expressive Description Logics and in the tractable *DL-Lite* family.

Introduction

Description Logics (DLs) are presently the most popular formalisms for expressing ontologies, and are at the basis of Ontology Languages for the Semantic Web, such as OWL (Bechhofer et al. 2004). Identity is one of the main principles in ontology engineering (Welty and Guarino 2001), and methods for explicitly asserting identity properties are provided in some modeling languages. For example, mechanisms for identifying objects using attribute values or their participation to relationships are present in most conceptual modeling formalisms used in software engineering, databases, and information systems (Fowler and Scott 1997; Chen 1976; Hull and King 1987).

In spite of the importance of identity, the DLs at the basis of current DL reasoners do not include specific mechanisms for expressing identification constraints. For example, in OWL[1], the only way to specify identification is through the use of one-to-one relationships, and this corresponds to a very limited form of identification. More powerful identification mechanisms have been already studied in the context of very expressive Description Logics, see, e.g., (Calvanese, De Giacomo, and Lenzerini 2001; Lutz et al. 2005; Toman and Weddell 2008), but they have not been incorporated in DL reasoners yet.

Following the above mentioned papers, we argue for the usefulness of identification constraints in modeling a domain of interest through an ontology. For example, in the context of geographic information systems, the fact that a location is identified by its coordinates should be considered as part of the very definition of location. The lack of identification mechanisms is even more serious if one considers that in most DLs both n-*ary relations* and *attributes of roles* are missing. Indeed, the only way to represent an arbitrary n-ary relation in the most popular DLs is through the well-known *reification*[2] technique (Baader et al. 2003). Reification, as well as modeling attributes of roles, actually need identification constraints, as we briefly illustrate in the following simple example.

Consider the notion of enrolment, where each instance relates a student s, a program p (e.g., Computer Science), and the year of enrolment of s in p, in such a way that no two instances exist for the same pair (s, p). This notion can be modeled in FOL with a ternary relation, but in DLs with binary relations only, it must be represented by the reified concept *enrolment*, two functional binary roles *HAS-STUDENT* (from *enrolment* to *student*) and *HAS-PROGRAM* (from *enrolment* to *program*), and one functional attribute $\mathbf{year}$ for *enrolment*. However, in order for this reified representation to be correct, we must also impose that no two distinct instances of *enrolment* exist that are connected to the same pair of fillers for *HAS-STUDENT* and *HAS-PROGRAM*. This is exactly what an identification constraint can be used for: in this case, *HAS-STUDENT* and *HAS-PROGRAM* form an identifier for *enrolment*.

To continue the example, suppose that every program is offered by one or more universities (forming a sort of consortium), represented by the binary role *OFFERED-BY* from *program* to the concept *university*, and that every student has a code, represented by the functional attribute $\mathbf{code}$ for *student*, that is unique for all universities offering the programs in which she is enrolled. To model this situation,

[1] Although we generically use the term "OWL", in this paper we focus on OWL-DL.

[2] By reification we mean here the use of an object to denote a tuple, and it should not be confused with reification where an object is used to denote a predicate, like, for instance, in RDF.

we need to define an identifier for *student* constituted by the combination of **code** and the path connecting the student to the universities through the program she is enrolled in. Note that this path is constituted by the composition of the inverse of the role *HAS-STUDENT*, the role *HAS-PROGRAM* and the role *OFFERED-BY*.

The above example shows the need of sophisticated forms of identification constraints to accurately model the domain of interest. In particular, the example calls for identification constraints including paths with (non-functional) roles and inverse roles, called path-based identification constraints. According to our experience (see, for instance, (Amoroso et al. 2008)), this example represents a typical situation, as such constraints are of vital importance in modeling real-world applications. Therefore, the lack of identification mechanisms in ontology languages is one of the most relevant gaps between requirements from the real world, and current semantic technologies.

In this paper, we carry out an extensive study of path-based identification constraints in the context of two DLs, the expressive DL $\mathcal{ALCQIb}_{reg}$ (Calvanese, Eiter, and Ortiz 2007), and the tractable DL *DL-Lite*$_{\mathcal{A}}$ (Poggi et al. 2008). Our contributions can be summarised as follows:

- We show that path-based identification constraints with no limitations are problematic with respect to the effectiveness/efficiency of reasoning. In particular, adding such constraints to $\mathcal{ALCI}$ leads to undecidability, whereas extending *DL-Lite*$_{\mathcal{A}}$ with path-based identifiers makes query answering NLOGSPACE-hard with respect to ABox complexity, and therefore computationally harder than in *DL-Lite*$_{\mathcal{A}}$.
- We propose a restricted form of path-based identification constraints, which we call *local*, requiring that at least one of the component paths of the identifier has length 1. Intuitively, this means that every identifier of a concept C must include at least one direct (or "local") property of C. We show that adding identification constraints of this restricted form does not increase the complexity of reasoning both in $\mathcal{ALCQIb}_{reg}$, and in *DL-Lite*$_{\mathcal{A}}$. Conversely, our results imply that going beyond the locality restriction leads to undecidability/increase in complexity.

Note that all identification constraints discussed in the above examples are indeed local, and so are virtually all identification mechanisms introduced in conceptual modeling (Hull and King 1987) (see, for example, the notion of "weak entity" in the Entity-Relationship model). More generally, we argue that local path-based identification constraints capture most of the identifiers needed in real applications, and therefore locality does not constitute a severe limitation in practice.

Although this is not the first investigation on this subject, previous work concentrated on special cases of identification constraints, such as paths with only functional roles, paths with no inverses, or paths of length 1 (related work is discussed in detail in Section 3). More precisely, to the best of our knowledge, the results presented in this paper are the first decidability/tractability results on path-based identification constraints with roles and inverse roles. Our work can provide the basis for incorporating powerful forms of identification constraints in DLs, including OWL, and for extending current DL reasoners with suitable capabilities for dealing with such constraints, thus filling the gap between semantic technologies and one of the most important modeling requirements coming from real world applications.

The rest of the paper is organized as follows. In the next section we describe the two DLs studied in our work. In the following section we provide the formal definitions of the identification constraints proposed in the paper, we present examples of ontologies with such constraints, and we discuss related work. In the two following sections we investigate the issue of adding path-based identification constraints to expressive DLs and to tractable DLs, respectively. Finally, the last section provides our conclusions.

Preliminaries

We briefly discuss the two main DLs considered in this paper, namely $\mathcal{ALCQIb}_{reg}$ (Calvanese, Eiter, and Ortiz 2007), and *DL-Lite*$_{\mathcal{A}}$ (Poggi et al. 2008).

$\mathcal{ALCQIb}_{reg}$ is one of the most expressive DLs proposed in the literature. It can encode $\mathcal{DLR}/\mathcal{DLR}_{reg}$, $\mathcal{SHIQ}$, and PDL with inverse and graded modalities. Moreover, answering unions of conjunctive queries over $\mathcal{ALCQIb}_{reg}$ knowledge bases (KBs) has been shown to be decidable (Calvanese, Eiter, and Ortiz 2007). Concepts and roles in this DL are formed according to the following syntax,

$$\begin{array}{lcl} C, C' & \longrightarrow & A \mid \neg C \mid C \sqcap C' \mid C \sqcup C' \mid \forall R.C \mid \\ & & \exists R.C \mid \geqslant n\, Q.C \mid \leqslant n\, Q.C \\ Q, Q' & \longrightarrow & P \mid P^- \mid Q \cap Q' \mid Q \cup Q' \mid Q \setminus Q' \\ R, R' & \longrightarrow & Q \mid R \cup R' \mid R \circ R' \mid R^* \mid C? \end{array}$$

where A denotes an *atomic concept*, P an *atomic role*, P^- the *inverse* of an atomic role, C an arbitrary *concept*, and R an arbitrary *role*. Furthermore, $\neg C$, $C \sqcap C'$, $C \sqcup C'$, $\forall R.C$, and $\exists R.C$ denote negation of concepts, concept intersection, concept union, value restriction, and qualified existential quantification on roles, respectively. We then use Q to denote *basic roles*, which are those roles that may occur in number restrictions, that are expressions of the form $\geqslant n\, Q.C$ and $\leqslant n\, Q.C$. A basic role can be an atomic role or its inverse, or a role obtained combining basic roles through set theoretic operators, i.e., intersection ("$\cap$"), union ("$\cup$"), and difference("$\setminus$"). W.l.o.g., we assume difference applied only to atomic roles and their inverses. Finally, observe that arbitrary roles are regular expressions over basic roles, where $R \circ R'$ and R^* respectively denote role composition and reflexive-transitive closure (a second order property), whereas $C?$, called *test role*, denotes the identity relation on instances of the concept C.

An $\mathcal{ALCQIb}_{reg}$ *knowledge base* (KB) is a pair $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ (the *TBox*) is a finite set of *general inclusion assertions* $C \sqsubseteq C'$ with C and C' arbitrary $\mathcal{ALCQIb}_{reg}$ concepts, and $\mathcal{A}$ (the *ABox*) is a finite set of *membership assertions* of the form $A(a)$, $P(a, b)$, and $a \neq b$, with A and P respectively an atomic concept and an atomic role occurring in $\mathcal{T}$, and a, b constants.

The semantics of $\mathcal{ALCQIb}_{reg}$ concepts and roles is given in terms of interpretations, where an interpretation $\mathcal{I} =$

$$
\begin{array}{lcl}
A^{\mathcal{I}} & \subseteq & \Delta^{\mathcal{I}} \\
\neg C^{\mathcal{I}} & = & \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
(C \sqcap C')^{\mathcal{I}} & = & C^{\mathcal{I}} \cap C'^{\mathcal{I}} \\
(C \sqcup C')^{\mathcal{I}} & = & C^{\mathcal{I}} \cap C'^{\mathcal{I}} \\
(\forall R.C)^{\mathcal{I}} & = & \{\, o \mid \forall o'.\, (o,o') \in R^{\mathcal{I}} \supset o' \in C^{\mathcal{I}} \,\} \\
(\exists R.C)^{\mathcal{I}} & = & \{\, o \mid \exists o'.\, (o,o') \in R^{\mathcal{I}} \wedge o' \in C^{\mathcal{I}} \,\} \\
(\geqslant n\, Q.C)^{\mathcal{I}} & = & \{\, o \mid |\{o' \in C^{\mathcal{I}} \mid (o,o') \in Q^{\mathcal{I}}\}| \geq n \,\} \\
(\leqslant n\, Q.C)^{\mathcal{I}} & = & \{\, o \mid |\{o' \in C^{\mathcal{I}} \mid (o,o') \in Q^{\mathcal{I}}\}| \leq n \,\}
\end{array}
$$

$$
\begin{array}{lcl}
P^{\mathcal{I}} & \subseteq & \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \\
(P^-)^{\mathcal{I}} & = & \{\, (o,o') \mid (o',o) \in P^{\mathcal{I}} \,\} \\
(Q \cap Q')^{\mathcal{I}} & = & Q^{\mathcal{I}} \cap Q'^{\mathcal{I}} \\
(Q \cup Q')^{\mathcal{I}} & = & Q^{\mathcal{I}} \cup Q'^{\mathcal{I}} \\
(Q \setminus Q')^{\mathcal{I}} & = & Q^{\mathcal{I}} \setminus Q'^{\mathcal{I}} \\
(R \cup R')^{\mathcal{I}} & = & R^{\mathcal{I}} \cup R'^{\mathcal{I}} \\
(R \circ R')^{\mathcal{I}} & = & R^{\mathcal{I}} \circ R'^{\mathcal{I}} \\
(R^*)^{\mathcal{I}} & = & (R^{\mathcal{I}})^* \\
(C?)^{\mathcal{I}} & = & \{\, (o,o) \mid o \in C^{\mathcal{I}} \,\}
\end{array}
$$

Figure 1: Interpretation of $\mathcal{ALCQI}b_{reg}$ concepts and roles

$(\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty interpretation domain $\Delta^{\mathcal{I}}$ and an interpretation function $\cdot^{\mathcal{I}}$, defined in Figure 1.

An interpretation $\mathcal{I}$ satisfies an inclusion assertion $C \sqsubseteq C'$, if $C^{\mathcal{I}} \subseteq C'^{\mathcal{I}}$.

To specify the semantics of $\mathcal{ALCQI}b_{reg}$ ABox assertions, we extend the interpretation function to constants, by assigning to each constant a a object $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. Note that, for $\mathcal{ALCQI}b_{reg}$, we do not enforce the unique name assumption on constants (Baader et al. 2003). Then, an interpretation $\mathcal{I}$ satisfies a membership assertion $A(a)$ if $a^{\mathcal{I}} \in A^{\mathcal{I}}$, a membership assertion $P(a,b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in P^{\mathcal{I}}$, and an assertion of the form $a \neq b$ if $a^{\mathcal{I}} \neq b^{\mathcal{I}}$.

We now turn to *DL-Lite*$_{\mathcal{A}}$, a member of the *DL-Lite* family (Calvanese et al. 2007c; 2006b), and hence a tractable DL particularly suited for dealing with KBs with very large ABoxes, which can be managed through relational database technology. *DL-Lite*$_{\mathcal{A}}$ distinguishes concepts from *value-domains*, which denote sets of (data) values, and roles from *attributes*, which denote binary relations between objects and values. Concepts, roles, attributes, and value-domains in this DL are formed according to the following syntax[3]:

$$
\begin{array}{lll}
B \longrightarrow A \mid \exists Q \mid \delta(U) & & E \longrightarrow \rho(U) \\
C \longrightarrow B \mid \neg B & & F \longrightarrow \top_D \mid T_1 \mid \cdots \mid T_n \\
Q \longrightarrow P \mid P^- & & V \longrightarrow U \mid \neg U \\
R \longrightarrow Q \mid \neg Q & &
\end{array}
$$

In such rules, A, P, and P^- respectively denote an atomic concept, an atomic role, and the inverse of an atomic role, exactly as in $\mathcal{ALCQI}b_{reg}$, Q and R respectively denote a basic and an arbitrary role as in $\mathcal{ALCQI}b_{reg}$, but with a different syntax, whereas B denotes a *basic concept*, U an *atomic attribute*, V an *arbitrary attribute*, E a *basic value-domain*, and F an *arbitrary value-domain*. Furthermore, $\delta(U)$ denotes the *domain* of U, i.e., the set of objects that U relates to values; $\rho(U)$ denotes the *range* of U, i.e., the set of values that U relates to objects; $\top_D$ is the universal value-domain; $T_1, \ldots, T_n$ are n pairwise disjoint unbounded value-domains, corresponding to RDF data types, such as `xsd:string`, `xsd:integer`, etc.

[3]The results given in this paper apply also to *DL-Lite*$_{\mathcal{A}}$ extended with role attributes (cf. (Calvanese et al. 2006a)), which are not considered here for the sake of simplicity.

A *DL-Lite*$_{\mathcal{A}}$ KB is a pair $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$ is the TBox and $\mathcal{A}$ the ABox. The TBox $\mathcal{T}$ is defined as follows:

(1) $\mathcal{T}$ is a finite set of assertions of the forms

$$
\begin{array}{cccc}
B \sqsubseteq C & Q \sqsubseteq R & E \sqsubseteq F & U \sqsubseteq V \\
 & (\mathsf{funct}\ Q) & (\mathsf{funct}\ U) &
\end{array}
$$

where, from left to right, assertions of the first row respectively denote inclusions between concepts, roles, value-domains, and attributes, whereas assertions of the second row denote functionality on roles and on attributes.

(2) $\mathcal{T}$ satisfies the following conditions:

- for each atomic role P, if either $(\mathsf{funct}\ P)$ or $(\mathsf{funct}\ P^-)$ occur in $\mathcal{T}$, then $\mathcal{T}$ does not contain assertions of the form $Q' \sqsubseteq P$ or $Q' \sqsubseteq P^-$, where Q' is a basic role;
- for each atomic attribute U, if $(\mathsf{funct}\ U)$ occurs in $\mathcal{T}$, then $\mathcal{T}$ does not contain assertions of the form $U' \sqsubseteq U$, where U' is an atomic attribute.

Intuitively, the conditions stated at point (2) say that, in *DL-Lite*$_{\mathcal{A}}$ TBoxes, roles and attributes occurring in functionality assertions cannot be specialized.

In order to define *DL-Lite*$_{\mathcal{A}}$ ABoxes, we need to explicitly define the set of constants used in such ABoxes. We denote with Γ the alphabet for constants, which we assume partitioned into two sets, namely, Γ_V (the set of constant symbols for values), and Γ_O (the set of constant symbols for objects). In turn, Γ_V is partitioned into n sets $\Gamma_{V_1}, \ldots, \Gamma_{V_n}$, where each Γ_{V_i} is the set of constants for the values in the value-domain T_i. Notice that every Γ_{V_i} is infinite, that is, we only consider unbounded value-domains. Then, the ABox $\mathcal{A}$ is a finite set of membership assertions of the forms $A(a)$, $P(a,b)$, and $U(a,v)$, where A, P, and U are as above, a and b are object constants in Γ_O, and v is a value constant in Γ_V.

The semantics of *DL-Lite*$_{\mathcal{A}}$ concepts, roles, attributes and value-domains is given in terms of interpretations. We notice that all *DL-Lite*$_{\mathcal{A}}$ interpretations agree on the semantics assigned to each value-domain T_i and to each constant in Γ_V. More precisely, we assume to have a unique (infinite, non-empty) domain Δ_V used to interpret constants in Γ_V, and partitioned into n infinite, non-empty subsets

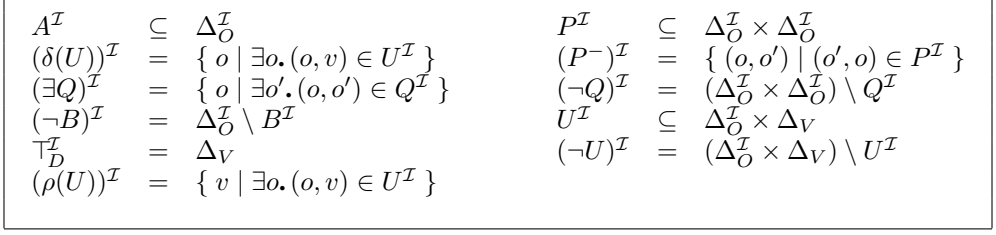

Figure 2: Interpretation of *DL-Lite*$_{\mathcal{A}}$ concepts, roles, and attributes

$val(T_1), \ldots, val(T_n)$, where each $val(T_i)$ is used to interpret the corresponding T_i, in such a way that each value constant $v \in \Gamma_{V_i}$ is interpreted as one distinct value in $val(T_i)$.

Then, a *DL-Lite*$_{\mathcal{A}}$ interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty interpretation domain $\Delta^{\mathcal{I}} = \Delta_V \cup \Delta_O^{\mathcal{I}}$, where $\Delta_O^{\mathcal{I}}$ is the domain used to interpret object constants in Γ_O, and of an interpretation function $\cdot^{\mathcal{I}}$, which is defined in Figure 2. We say that an interpretation $\mathcal{I}$ satisfies

- a concept (resp., value-domain, role, attribute) inclusion assertion $B \sqsubseteq C$ (resp., $E \sqsubseteq F$, $Q \sqsubseteq R$, $U \sqsubseteq V$), if $B^{\mathcal{I}} \subseteq C^{\mathcal{I}}$ (resp., $E^{\mathcal{I}} \subseteq F^{\mathcal{I}}$, $Q^{\mathcal{I}} \subseteq R^{\mathcal{I}}$,$U^{\mathcal{I}} \subseteq V^{\mathcal{I}}$),
- a role functionality assertion (funct Q), if for each $o, o', o'' \in \Delta_O^{\mathcal{I}}$, we have that $(o,o') \in Q^{\mathcal{I}}$ and $(o,o'') \in Q^{\mathcal{I}}$ implies $o' = o''$,
- an attribute functionality assertion (funct U), if for each $o \in \Delta_O^{\mathcal{I}}$ and $o'_v, o''_v \in \Delta_V$, we have that $(o,o'_v) \in U^{\mathcal{I}}$ and $(o,o''_v) \in U^{\mathcal{I}}$ implies $o'_v = o''_v$.

To specify the semantics of *DL-Lite*$_{\mathcal{A}}$ membership assertions, we extend the interpretation function to object constants (interpretation of value constants has been discussed above), by assigning to each constant $a \in \Gamma_O$ an object $a^{\mathcal{I}} \in \Delta_O^{\mathcal{I}}$. Note that, differently from $\mathcal{ALCQI}b_{reg}$, in *DL-Lite*$_{\mathcal{A}}$ we adopt the unique name assumption on both value and object constants, i.e., for each $a, b \in \Gamma$, $a \neq b$ implies $a^{\mathcal{I}} \neq b^{\mathcal{I}}$. Then, an interpretation $\mathcal{I}$ satisfies a membership assertion $A(a)$ (resp., $P(a,b)$, $U(a,v)$) if $a^{\mathcal{I}} \in A^{\mathcal{I}}$ (resp., $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in P^{\mathcal{I}}$, $(a^{\mathcal{I}}, v^{\mathcal{I}}) \in U^{\mathcal{I}}$).

With the above notions in place, we can introduce the definition of model for either a *DL-Lite*$_{\mathcal{A}}$ or an $\mathcal{ALCQI}b_{reg}$ KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$. Precisely, we say that an interpretation $\mathcal{I}$ is a *model* of $\mathcal{T}$ (resp., $\mathcal{A}$) if $\mathcal{I}$ satisfies all the assertions in $\mathcal{T}$ (resp., $\mathcal{A}$). A model of $\mathcal{K}$ is a model of both $\mathcal{T}$ and $\mathcal{A}$. Obviously, the meaning of satisfaction here depends on the DL in which $\mathcal{K}$ is specified (i.e., *DL-Lite*$_{\mathcal{A}}$ or $\mathcal{ALCQI}b_{reg}$).

We now concentrate on reasoning. The reasoning services we consider in this paper are *KB satisfiability*, i.e., determining whether a knowledge base $\mathcal{K}$ admits at least one model, and answering unions of conjunctive queries.

A *union of conjunctive queries* (UCQ) q over either an $\mathcal{ALCQI}b_{reg}$ or a *DL-Lite*$_{\mathcal{A}}$ KB $\mathcal{K}$ is an expression of the form $q(\vec{x}) \leftarrow \exists \vec{y_1}.conj_1(\vec{x}, \vec{y_1}) \vee \cdots \vee \exists \vec{y_n}.conj_n(\vec{x}, \vec{y_n})$, where $\vec{x}$ are the *distinguished variables*, $\vec{y_1}, \ldots, \vec{y_n}$ are the *non-distinguished* variables, and each $conj_i(\vec{x}, \vec{y_i})$ is a conjunction of atoms of the form $D(z)$, $S(z,z')$, $z = z'$, where D denotes a concept (resp., basic value-domain, or arbitrary value-domain) occurring in $\mathcal{K}$, S denotes an atomic role (resp., attribute) or the inverse of an atomic role (resp., attribute[4]) occurring in $\mathcal{K}$, and z, z' are constants in $\mathcal{K}$ or variables in $\vec{x}$ or $\vec{y_i}$, for some $i \in \{1, \ldots, n\}$. Given an interpretation $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, $q^{\mathcal{I}}$ is the set of tuples of $\Delta^{\mathcal{I}}$ that, when assigned to the distinguished variables $\vec{x}$, make the formula $\exists \vec{y_1}.conj_1(\vec{x}, \vec{y_1}) \vee \cdots \vee \exists \vec{y_n}.conj_n(\vec{x}, \vec{y_n})$ true in $\mathcal{I}$. Then, the set $cert(q, \mathcal{K})$ of *certain answers to q over $\mathcal{K}$* is the set of tuples $\vec{a}$ of constants appearing in $\mathcal{K}$ such that $\vec{a}^{\mathcal{I}} \in q^{\mathcal{I}}$, for every model $\mathcal{I}$ of $\mathcal{K}$. In the following, instead of query answering, we consider, w.l.o.g., boolean *query entailment* $\mathcal{K} \models q$, i.e., checking whether $\langle\rangle \in cert(q, \mathcal{K})$, where q is a Boolean UCQ (that is, a UCQ with no distinguished variables) and $\langle\rangle$ denotes the empty tuple.

Path-based identification constraints

The goal of this section is to formally introduce the constraints studied in this paper. In particular, we illustrate both syntax and semantics of such constraints, and we discuss their use by means of some examples. Finally, we compare our proposal with related work.

Since the form of identification constraints proposed in this paper is based on paths, we start our presentation by introducing the notion of path used in our context.

A *path* in a DL $\mathcal{L}$ is an expression built according to the following syntax,

$$\pi \longrightarrow S \mid D? \mid \pi \circ \pi$$

where S denotes an atomic role (resp., attribute) or the inverse of an atomic role (resp., attribute) in $\mathcal{L}$, D denotes a concept (resp., a basic value-domain, or an arbitrary value-domain) in $\mathcal{L}$, and $\pi_1 \circ \pi_2$ denotes the composition of paths π_1 and π_2. Finally, the expression $D?$ is called a test relation, which represents the identity relation on instances of D. Test relations are used in all those cases in which we want to impose that a path involves instances of a certain concept, basic value-domain, or arbitrary value-domain. For example, consider again the context described in the introduction in which students are enrolled in programs offered by universities. Formally, the path connecting a student to the universities is

[4]The semantics of the inverse of an atomic attribute is the obvious one.

HAS-STUDENT$^-$ ∘ *HAS-PROGRAM* ∘ *OFFERED-BY*. Now, if we would like to consider only paths involving master programs, modeled with the concept *masterProgram*, a subconcept of the concept *program*, we should specify the path *HAS-STUDENT*$^-$ ∘ *HAS-PROGRAM* ∘ *masterProgram*? ∘ *OFFERED-BY*.

A path π denotes a complex property for the instances of concepts: given an object o in a certain interpretation $\mathcal{I}$, every object that is reachable from o in $\mathcal{I}$ by means of π is called a π-*filler* for o. Note that previous works on path-based identification constraints generally refer to DLs where roles are total functions (Toman and Weddell 2005; 2008). Our study, instead, focuses on DLs where roles are arbitrary binary relations (and their inverse), and therefore there may be several distinct π-fillers, or no π-fillers at all, for o.

If π is a path, the length of π, denoted $length(\pi)$, is 0 if π has the form $D?$, is 1 if π has the form S, and is $length(\pi_1)+length(\pi_2)$ if π has the form $\pi_1 \circ \pi_2$. A path with length 1 is called *simple*.

We now turn our attention to identification constraints.

Definition 1 A *path-based identification constraint* (or, simply, identification constraint – IdC) in a DL $\mathcal{L}$ is an assertion of the form

$$(\mathsf{id}\ C\ \pi_1, \ldots, \pi_n)$$

where C is a concept in $\mathcal{L}$, $n \geq 1$, and $\pi_1, \ldots, \pi_n$ (called the *components* of the identifier) are paths in $\mathcal{L}$ such that $length(\pi_i) \geq 1$ for all $i \in \{1, \ldots, n\}$.

Intuitively, the above constraint asserts that for any two different instances o, o' of C, there is at least one π_i such that o and o' differ in the set of their π_i-fillers.

An IdC $(\mathsf{id}\ C\ \pi_1, \ldots, \pi_n)$ is called *simple* if all paths $\pi_1, \ldots, \pi_n$ are simple, is called *single-path IdC* if $n = 1$, and is called *local* if at least one of the paths $\pi_1, \ldots, \pi_n$ is simple (the term "local" emphasizes that at least one of the paths refers to a local property of C). Note that we have ruled out components of length 0 in IdCs, since they would make an IdC trivially satisfied in every interpretation (see below).

In order to define the semantics of IdCs, we first define the semantics of paths, and then specify the conditions for an interpretation to satisfy an IdC.

The extension $\pi^{\mathcal{I}}$ of a path π in an interpretation $\mathcal{I}$ is defined as follows:

- if $\pi = S$, then $\pi^{\mathcal{I}} = S^{\mathcal{I}}$,
- if $\pi = D?$, then $\pi^{\mathcal{I}} = \{\ (o, o) \mid o \in D^{\mathcal{I}}\ \}$,
- if $\pi = \pi_1 \circ \pi_2$, then $\pi^{\mathcal{I}} = \pi_1^{\mathcal{I}} \circ \pi_2^{\mathcal{I}}$, where $\circ$ denotes the composition operator on relations.

As a notation, we write $\pi^{\mathcal{I}}(o)$ to denote the set of π-fillers for o in $\mathcal{I}$, i.e., $\{o' \mid (o, o') \in \pi^{\mathcal{I}}\}$. Then, an interpretation $\mathcal{I}$ satisfies the IdC $(\mathsf{id}\ C\ \pi_1, \ldots, \pi_n)$ if for all $o, o' \in C^{\mathcal{I}}$, $\pi_1^{\mathcal{I}}(o) \cap \pi_1^{\mathcal{I}}(o') \neq \emptyset \wedge \cdots \wedge \pi_n^{\mathcal{I}}(o) \cap \pi_n^{\mathcal{I}}(o') \neq \emptyset$ implies $o = o'$. Observe that this definition is coherent with the intuitive reading of IdCs discussed above, in particular by sanctioning that two different instances o, o' of C differ in the set of their π_i-fillers when such sets are disjoint.

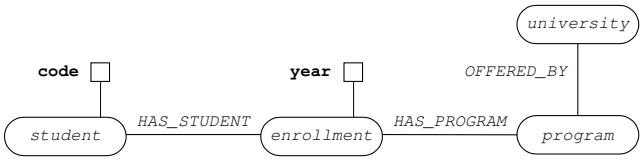

Figure 3: Diagrammatic representation of the students ontology

Examples

We start by considering again the students example described in the introduction. In Figure 3, we show a schematic representation of (part of) the ontology for such a domain of interest. Concepts are represented by ovals, attributes are represented by squares connected to the concepts they refer to, and roles are drawn as lines connecting the appropriate concepts. The TBox assertions capturing the students domain are given in Figure 4 (the TBox is expressed in *DL-Lite*$_{\mathcal{A}}$ with IdCs).

INCLUSION ASSERTIONS		
$\exists$*HAS-STUDENT*	$\sqsubseteq$	*enrolment*
$\exists$*HAS-STUDENT*$^-$	$\sqsubseteq$	*student*
$\exists$*HAS-PROGRAM*	$\sqsubseteq$	*enrolment*
$\exists$*HAS-PROGRAM*$^-$	$\sqsubseteq$	*program*
$\exists$*OFFERED-BY*	$\sqsubseteq$	*program*
$\exists$*OFFERED-BY*$^-$	$\sqsubseteq$	*university*
student	$\sqsubseteq$	$\delta(\mathbf{code})$
enrolment	$\sqsubseteq$	$\delta(\mathbf{year})$
FUNCTIONAL ASSERTIONS		
(funct **code**)		(funct *HAS-STUDENT*)
(funct **year**)		(funct *HAS-PROGRAM*)
IDENTIFICATION CONSTRAINTS		
(id *enrolment HAS-STUDENT*, *HAS-PROGRAM*)		
(id *student HAS-STUDENT*$^-$ ∘ *HAS-PROGRAM* ∘ *OFFERED-BY*, **code**)		

Figure 4: The TBox in *DL-Lite*$_{\mathcal{A}}$ with IdCs for the students example

As for the IdCs given in the TBox, the first one imposes that no two instances of *enrolment* exist that are connected to the same pair of fillers for *HAS-STUDENT* and *HAS-PROGRAM*, whereas the second IdC imposes that no two students s and s' exist with both the same code and a non-empty intersection of the sets of universities that are the π-fillers for s and s', where π = *HAS-STUDENT*$^-$ ∘ *HAS-PROGRAM*∘*OFFERED-BY*. Note that in the two IdCs, *HAS-STUDENT* is used in both the direct and the inverse direction, and also that *OFFERED-BY* is a non-functional role.

Let us now consider a second, more complex example. We aim at modeling the annual national football[5] championships in Europe, where the championship for a specific year and for a specific nation is called *league* (e.g., the 2008 Spanish Liga). A league is structured in terms of a set of

[5]Football is called "soccer" in the United States.

INCLUSION ASSERTIONS					
league	$\sqsubseteq$	$\exists$*OF*	*match*	$\sqsubseteq$	$\exists$*HOST*
$\exists$*OF*	$\sqsubseteq$	*league*	*playedMatch*	$\sqsubseteq$	*match*
$\exists$*OF*$^-$	$\sqsubseteq$	*nation*	*league*	$\sqsubseteq$	$\delta(\mathbf{year})$
round	$\sqsubseteq$	$\exists$*BELONGS-TO*	*match*	$\sqsubseteq$	$\delta(\mathbf{code})$
$\exists$*BELONGS-TO*	$\sqsubseteq$	*round*	*round*	$\sqsubseteq$	$\delta(\mathbf{code})$
$\exists$*BELONGS-TO*$^-$	$\sqsubseteq$	*league*	*playedMatch*	$\sqsubseteq$	$\delta(\mathbf{playedOn})$
match	$\sqsubseteq$	$\exists$*PLAYED-IN*	*playedMatch*	$\sqsubseteq$	$\delta(\mathbf{homeGoals})$
$\exists$*PLAYED-IN*	$\sqsubseteq$	*match*	*playedMatch*	$\sqsubseteq$	$\delta(\mathbf{hostGoals})$
$\exists$*PLAYED-IN*$^-$	$\sqsubseteq$	*round*	$\rho(\mathbf{playedOn})$	$\sqsubseteq$	`xsd:date`
match	$\sqsubseteq$	$\exists$*HOME*	$\rho(\mathbf{homeGoals})$	$\sqsubseteq$	`xsd:nonNegativeInteger`
$\exists$*HOME*	$\sqsubseteq$	*match*	$\rho(\mathbf{hostGoals})$	$\sqsubseteq$	`xsd:nonNegativeInteger`
$\exists$*HOME*$^-$	$\sqsubseteq$	*team*	$\rho(\mathbf{code})$	$\sqsubseteq$	`xsd:positiveInteger`
$\exists$*HOST*	$\sqsubseteq$	*match*	$\rho(\mathbf{year})$	$\sqsubseteq$	`xsd:positiveInteger`
$\exists$*HOST*$^-$	$\sqsubseteq$	*team*			

FUNCTIONAL ASSERTIONS	
(funct *OF*)	(funct $\mathbf{year}$)
(funct *BELONGS-TO*)	(funct $\mathbf{code}$)
(funct *PLAYED-IN*)	(funct $\mathbf{playedOn}$)
(funct *HOME*)	(funct $\mathbf{homeGoals}$)
(funct *HOST*)	(funct $\mathbf{hostGoals}$)

IDENTIFICATION CONSTRAINTS	
1. (id *league OF*, $\mathbf{year}$)	6. (id *playedMatch* $\mathbf{playedOn}$, *HOST*)
2. (id *round BELONGS-TO*, $\mathbf{code}$)	7. (id *playedMatch* $\mathbf{playedOn}$, *HOME*)
3. (id *match PLAYED-IN*, $\mathbf{code}$)	8. (id *league* $\mathbf{year}$, *BELONGS-TO*$^-$ $\circ$ *PLAYED-IN*$^-$ $\circ$ *HOME*)
4. (id *match HOME*, *PLAYED-IN*)	9. (id *league* $\mathbf{year}$, *BELONGS-TO*$^-$ $\circ$ *PLAYED-IN*$^-$ $\circ$ *HOST*)
5. (id *match HOST*, *PLAYED-IN*)	10. (id *match HOME*, *HOST*, *PLAYED-IN* $\circ$ *BELONGS-TO* $\circ$ $\mathbf{year}$)

Figure 6: The TBox in *DL-Lite*$_\mathcal{A}$ with IdCs for the football leagues example

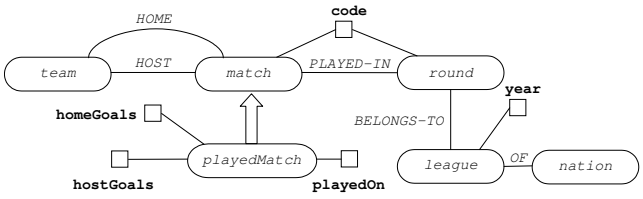

Figure 5: Diagrammatic representation of the football leagues ontology

rounds. In every round, a set of *matches* is scheduled, where each match is characterized by one *home team* and one *host team*. When a scheduled match takes place, it is played in a specific date, and for every match that has been played, the number of goals scored by the home team and by the host team are given. Note that different matches scheduled for the same round can be played in different dates.

In Figure 5 we show a schematic representation of (part of) the ontology for the football leagues domain. In this figure, the arrow represents inclusion between concepts. The TBox assertions capturing the above aspects are shown in the upper parts of Figure 6. The resulting TBox is expressed in terms of inclusion and functional *DL-Lite*$_\mathcal{A}$ assertions.

A natural question to ask is how accurate such assertions are in representing the semantics of our football domain. It is not hard to see that the ontology fails to model the following:

1. no nation has two leagues in the same year;
2. within a league, the code associated to a round is unique;
3. every match is identified by its code within its round;
4. a team is the home team of at most one match per round;
5. as above for the host team;
6. no home team participates in different played matches in the same date
7. as above for the host team;
8. no home team plays in different leagues in the same year;
9. as above for the host team;
10. no pair (home team, host team) plays different matches in the same year.

All the above aspects require the notion of identifier. The resulting ontology, expressed as a TBox in *DL-Lite*$_\mathcal{A}$ with IdCs, is illustrated in Figure 6 (the labels of the IdCs refer to the item numbers in the above list). Note that all the IdCs in the TBox are local.

Related work

Identification constraints have been investigated extensively in Databases (Abiteboul, Hull, and Vianu 1995) and in Conceptual Modeling (Brodie, Mylopoulos, and Schmidt 1984). Their importance in Description Logics has been recognized in the mid '90s, when they were introduced essentially in two variants: as a concept construct and as an assertion.

In particular, an identification construct (with simple identification, in our terminology) was added to the well-

known DL Classic (Borgida and Weddell 1997), where it was shown to keep logical implication PTIME. In (Calvanese, De Giacomo, and Lenzerini 1995; De Giacomo and Lenzerini 1995) a similar construct, but with a nonstandard semantics, was added to expressive DLs, which extended $\mathcal{ALCQI}$, and hence allowed for negating the identification construct. Recently, in (Toman and Weddell 2008) it has been shown that allowing for negating the identification construct with a standard semantics leads to undecidability, even in very simple DLs: it suffices to allow Boolean combinations of atomic concepts and identification constructs with functional identification paths of length 1. The best decidability result on path-based identification constructs is that in (Toman and Weddell 2005), in which EXPTIME-completeness is established for reasoning in the DL $\mathcal{DLFAD}$, which is essentially a variant of $\mathcal{ALCI}$ in which identification paths include functional roles only, and inverse roles are not allowed in paths.

A different line of research has looked at identification constraints as assertions (as we do here). In (Calvanese, De Giacomo, and Lenzerini 2001), identification assertions with paths of length 1 (simple IdCs, in our terminology) were added to $\mathcal{DLR}$, a DL that extends $\mathcal{ALCQI}$ with relations of arbitrary arity, and it was shown that such an extension preserves the EXPTIME-completeness of logical implication. Simple IdCs were also investigated in the context of the *DL-Lite*-family in (Calvanese et al. 2007a).

Identification assertions were studied in (Lutz et al. 2005) for an extension of $\mathcal{ALC}$ with concrete domains, allowing for identification paths of arbitrary length but involving functional roles only, and one final concrete feature. Note that inverse roles were not considered in that work. In (Nguyen and Le Thanh 2007), a tableaux procedure for logical implication in $\mathcal{SHOIQ}$ extended with identification assertions, was proposed. Again, identification assertions allow only for simple identification paths, as in (Calvanese, De Giacomo, and Lenzerini 2001). Finally, we note that OWL-DL, both in version 1 and in version 2, does not include identification constraints. However, proposals of adding a weak form of identification assertions are currently considered[6]. Syntactically, such identification assertions are simple IdCs, but they are interpreted only over individuals that are explicitly present in the ontology (i.e., they require that both the identified object and the identifying objects are named individuals in the ontology), making them akin to epistemic constraints (Calvanese et al. 2007b).

Reasoning in expressive DLs with IdCs

In this section, we investigate the addition of IdCs in expressive DLs of the $\mathcal{ALCI}$ family.

Undecidability of $\mathcal{ALCQI}b_{reg}$ with IdCs

We first show that if we allow for arbitrary IdCs, then reasoning becomes undecidable already in $\mathcal{ALCI}$. To do so, we exhibit a reduction from the unconstrained quadrant *tiling problem* (van Emde Boas 1997), which consists in deciding whether the first quadrant of the integer grid can be tiled using a finite set of square tile types in such a way that adjacent tiles respect adjacency conditions. Tiling problems are well suited to show undecidability of variants of description and dynamic logics (van Emde Boas 1997). The crux of the undecidability proof consists in enforcing that the tiles lie on an integer grid. Once the grid structure is enforced, it is typically easy to impose the adjacency conditions on the tiles. In our case, we exploit IdCs to construct the grid, following an idea already proposed in (Calvanese, De Giacomo, and Lenzerini 2001).

Theorem 2 *KB satisfiability, and thus query answering, in $\mathcal{ALCI}$ with IdCs is undecidable.*

Proof (sketch). The reduction is from the unconstrained quadrant *tiling problem* (van Emde Boas 1997). Formally, a *tiling system* is a triple $S = (\mathcal{D}, \mathcal{H}, \mathcal{V})$ where $\mathcal{D}$ is a finite set of elements representing tile types and $\mathcal{H}$ and $\mathcal{V}$ are two binary relations over $\mathcal{D}$. The *unconstrained quadrant tiling problem* consists in verifying the existence of a tiling *consistent with* S, i.e., a mapping τ from $\mathbb{N} \times \mathbb{N}$ to $\mathcal{D}$ such that $(\tau(h,k), \tau(h+1,k)) \in \mathcal{H}$ and $(\tau(h,k), \tau(h,k+1)) \in \mathcal{V}$, for $h, k \in \mathbb{N}$. Such a problem is undecidable, more precisely Π_0^0-complete (Berger 1966; van Emde Boas 1997).

Given a tiling system $S = (\mathcal{D}, \mathcal{H}, \mathcal{V})$, we construct an $\mathcal{ALCI}$ TBox with single-path IdCs $\mathcal{T}_S$ as follows. We use two atomic roles P_0 and P_1 that alternate in the horizontal and vertical directions to form a grid. We also use eight atomic concepts A_i^j, for $i \in \{0,1\}$, $j \in \{0,1,2,3\}$, that are pairwise disjoint i.e., $A_i^j \sqsubseteq \neg A_{i'}^{j'}$, for $i \neq i'$ or $j \neq j'$. See Figure 7, where a node (i,j) denotes an instance of A_i^j, and an edge labeled i denotes an instance of P_i. We enforce the grid structure by means of the following assertions in $\mathcal{T}_S$, for each $i \in \{0,1\}$, $j \in \{0,1,2,3\}$:

$$A_i^j \sqsubseteq \exists P_{(i+j) \bmod 2}.A_{1-i}^j \sqcap \exists P_{(i+j) \bmod 2}.A_i^{(j+1) \bmod 4} \sqcap \exists P^-_{(i+j+1) \bmod 2}.A_{1-i}^j \sqcap \exists P^-_{(i+j+1) \bmod 2}.A_i^{(j+3) \bmod 4}$$

$$(\mathsf{id}\ A_i^j\ P_{(i+j) \bmod 2} \circ P_{(i+j+1) \bmod 2}).$$

We enforce the adjacency conditions on the tiles of the first quadrant by using one concept for each tile type in $\mathcal{D}$ and

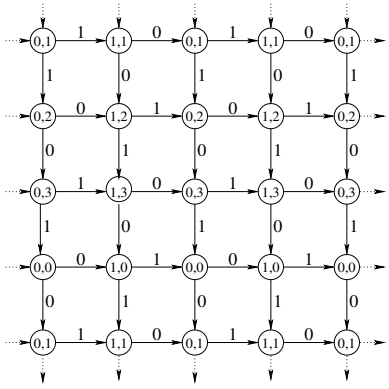

Figure 7: Grid structure enforced by $\mathcal{T}_S$

[6] http://www.w3.org/2007/OWL/wiki/Easy_Keys

introducing in $\mathcal{T}_S$ the following assertions: for each $D_h \in \mathcal{D}$, $i \in \{0,1\}$, $j \in \{0,1,2,3\}$

$$D_h \sqcap A_i^j \sqsubseteq \forall P_{(i+j) \bmod 2}.\bigsqcup_{(D_h,D_k)\in\mathcal{H}} D_k \sqcap \forall P_{(i+j+1) \bmod 2}.\bigsqcup_{(D_h,D_k)\in\mathcal{V}} D_k.$$

Finally, to represent the origin of the tiling, we use the concept $C_0 = A_0^0 \sqcap \bigsqcup_{D_h\in\mathcal{D}} D_h$. Then, it is possible to show that the tiling problem associated to S admits a solution if and only if the KB $\langle \mathcal{T}_S, \{C_0(a)\}\rangle$ is unsatisfiable.

Notice, that we can easily modify the reduction to show undecidability also in the presence of IdCs that are not single-path. Indeed, introduce an additional atomic role P, and for each $i \in \{0,1\}$, $j \in \{0,1,2,3\}$, add to $\mathcal{T}_S$ an assertion: $A_i^j \sqsubseteq \exists P$. Moreover, replace the single-path IdC above with:

$$(\mathsf{id}\ \ A_i^j\ \ P_{(i+j) \bmod 2} \circ P_{(i+j+1) \bmod 2},\ P \circ P^- \circ P_{(i+j) \bmod 2} \circ P_{(i+j+1) \bmod 2}).$$

Since each instance of an A_i^j concept is forced to have an outgoing P-edge, two such instances will have a common $P_{(i+j) \bmod 2} \circ P_{(i+j+1) \bmod 2}$-successor if and only if they also have a common $P \circ P^- \circ P_{(i+j) \bmod 2} \circ P_{(i+j+1) \bmod 2}$-successor. Actually, both successors will coincide. □

Decidability of $\mathcal{ALCQI}b_{reg}$ with local IdCs

We now show that, if we extend $\mathcal{ALCQI}b_{reg}$ with local IdCs, then answering unions of conjunctive queries becomes decidable. The intuition behind the decidability result is that local IdCs preserve the tree-model property of $\mathcal{ALCQI}b_{reg}$. More precisely, given an interpretation $\mathcal{I}$, consider the undirected graph $G_{\mathcal{I}}$ whose set of nodes is $\Delta^{\mathcal{I}}$ and in which there is an edge (o,o') iff $(o,o') \in P^{\mathcal{I}}$ or $(o',o) \in P^{\mathcal{I}}$, for some atomic role P. A tree-shaped model $\mathcal{I}$ is a model for which $G_{\mathcal{I}}$ is a tree. In the presence of an ABox, with arbitrarily connected individuals, tree-shaped models might be ruled out. However, given a KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$, let us define a *canonical model* of $\mathcal{K}$ to be a (forest-shaped) model of $\mathcal{K}$ in which each ABox individual is the root of a tree-shaped model of $\mathcal{T}$ (Calvanese and De Giacomo 2003; Vardi and Wolper 1986).

Theorem 3 *Let $\mathcal{K}$ be an $\mathcal{ALCQI}b_{reg}$ KB with local IdCs. Then $\mathcal{K}$ is satisfiable iff it has a canonical model.*

Proof (sketch). It suffices to show that we can transform an arbitrary model $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ of $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$ into a canonical model. For simplicity, we discuss here the case where the ABox $\mathcal{A}$ contains a single individual a. The construction can immediately be generalized, by repeating it for each individual in $\mathcal{A}$, while taking into account in the first steps also adjacent ABox individuals. The transformation is based on the standard unfolding process (see, e.g., (Vardi and Wolper 1986)), where, starting from the object $o_a \in \Delta^{\mathcal{I}}$ denoting a, a tree model $\mathcal{I}_t$ is constructed, together with a mapping ϑ from $\Delta^{\mathcal{I}_t}$ to a (suitably chosen) subset of $\Delta^{\mathcal{I}}$. Each object o' in $\Delta^{\mathcal{I}_t}$ is introduced in order to satisfy an existential quantification or an at-least number restriction, proceeding by induction on the length of a (suitably chosen) path in $\mathcal{I}$ from o_a to $\vartheta(o')$. The constructed model $\mathcal{I}_t$ is such that, when $\vartheta(o') = o$, then o and o' satisfy the same concepts of the syntactic closure (Calvanese, Eiter, and Ortiz 2007) of $\mathcal{K}$, and for each role R in the syntactic closure of $\mathcal{K}$, when $(o',o'') \in R^{\mathcal{I}_t}$ then $(\vartheta(o'), \vartheta(o'')) \in R^{\mathcal{I}}$. Notice that, during the unfolding process, multiple "copies" of an object $o \in \Delta^{\mathcal{I}}$ may be introduced in $\Delta^{\mathcal{I}_t}$, i.e., multiple objects $o', o'' \in \Delta^{\mathcal{I}_t}$ such that $\vartheta(o') = \vartheta(o'') = o$. We show that a local IdC $\alpha = (\mathsf{id}\ C\ \pi_1, \ldots, \pi_n)$ that is satisfied in $\mathcal{I}$ will also be satisfied in $\mathcal{I}_t$. Towards a contradiction, assume that α is not satisfied in $\mathcal{I}_t$, i.e., that the unfolding process generates objects $\{u_1', u_2', o_1', \ldots, o_n'\} \subseteq \Delta^{\mathcal{I}_t}$, with $u_1' \neq u_2'$, such that $\{u_1', u_2'\} \subseteq C^{\mathcal{I}_t}$ and $\{(u_1', o_i'), (u_2', o_i')\} \subseteq \pi_i^{\mathcal{I}_t}$, for $i \in \{1, \ldots, n\}$. Since α is local, there is an atomic or inverse atomic role S (that, together with tests, constitutes π_1) such that $\{(u_1', o_1'), (u_2', o_1')\} \subseteq S^{\mathcal{I}_t}$. Since $u_1' \neq u_2'$, and $\mathcal{I}_t$ is a tree-model, this can occur only in one of the following cases:

- Both u_1' and u_2' are S^--children of o_1' in $\mathcal{I}_t$. Then, in order for the unfolding process to generate two distinct children of a node, also in $\mathcal{I}$ we have that $\vartheta(u_1') \neq \vartheta(u_2')$.
- o_1' is an S-child of u_1', and u_2' is an S^--child of o_1'. Then, the unfolding process has first generated o_1' as an S-child of u_1', due to the presence of the S-successor $\vartheta(o_1')$ of $\vartheta(u_1')$ in $\mathcal{I}$. In such a situation, a new S^--child u_2' of o_1' will be generated only if required by the presence of an S^--successor of $\vartheta(o_1')$ in $\mathcal{I}$ that is different from $\vartheta(u_1')$. Hence $\vartheta(u_1') \neq \vartheta(u_2')$.
- o_1' is an S-child of u_2', and u_1' is an S^--child of o_1'. This case is symmetric to the preceding one.

Let $u_1 = \vartheta(u_1')$, $u_2 = \vartheta(u_2')$, and $o_i = \vartheta(o_i')$, for $i \in \{1, \ldots, n\}$. By construction of $\mathcal{I}_t$ and ϑ, we have that $\{u_1, u_2\} \subseteq C^{\mathcal{I}}$ and $\{(u_1, o_i), (u_2, o_i)\} \subseteq \pi_i^{\mathcal{I}}$, for $i \in \{1, \ldots, n\}$. In all three cases above we have $u_1 \neq u_2$, which contradicts the fact that α is satisfied in $\mathcal{I}$. □

The above result allows us to exploit techniques based on automata on infinite trees to solve the decision problem for query answering in $\mathcal{ALCQI}b_{reg}$ with local IdCs. Specifically, we adapt to our purposes the automata based algorithm proposed in (Calvanese, Eiter, and Ortiz 2007) for checking whether $\mathcal{K} \models q$, where $\mathcal{K} = \langle \mathcal{T}_{in}, \mathcal{A}\rangle$ is an $\mathcal{ALCQI}b_{reg}$ KB and q is a Boolean CQ. The algorithm proceeds as follows:

1. Construct a two-way alternating automaton on infinite trees (2ATA) $A_{\mathcal{K}}$ accepting trees that represent canonical models of $\mathcal{K}$.
2. Construct a 2ATA A_q accepting trees that represent interpretations (over the alphabet of $\mathcal{K}$) in which q is not satisfied.
3. Complement A_q and intersect it with $A_{\mathcal{K}}$, obtaining a (1-way) non-deterministic automaton on infinite trees (1NTA) $A_{\mathcal{K}\not\models q}$.

Then $A_{\mathcal{K}\not\models q}$ accepts infinite trees that represent counterexample models to $\mathcal{K} \models q$. Hence, $\mathcal{K} \models q$ iff $A_{\mathcal{K}\not\models q}$ is empty, i.e., accepts no infinite tree (Calvanese, Eiter, and Ortiz 2007).

We exploit such a technique for checking conjunctive query entailment $\mathcal{K} \models q$, where $\mathcal{K} = \langle \mathcal{T}, \mathcal{A}\rangle$, with $\mathcal{T} =$

$\mathcal{T}_{in} \cup \mathcal{C}$, where $\mathcal{T}_{in}$ is a set of $\mathcal{ALCQI}b_{reg}$ concept inclusion assertions, and $\mathcal{C}$ is a set of local IdCs. Let $\mathcal{K}_{in} = \langle \mathcal{T}_{in}, \mathcal{A} \rangle$. To deal with $\mathcal{K}_{in}$ and with the CQ q, we proceed exactly as described above for $\mathcal{ALCQI}b_{reg}$, and construct a 1NTA $A_{\mathcal{K}_{in} \not\models q}$ that accepts infinite trees representing counterexample models to $\mathcal{K}_{in} \models q$.

As a preliminary step, we define the notion of *negation of a set of IdCs*. Given an IdC $\alpha = (\mathsf{id}\ C\ \pi_1, \ldots, \pi_n)$, we define its negation $\neg\alpha$ as the Boolean CQ with an inequality

$$\neg\alpha \ = \ \exists \vec{x}. C(x) \wedge C(x') \wedge x \neq x' \wedge \bigwedge_{1 \leq i \leq n} (\tau(\pi_i(x, x_i)) \wedge \tau(\pi_i(x', x_i)))$$

where $\vec{x}$ are all variables appearing in the atoms of $\neg\alpha$, and $\tau(\pi(x, y))$ is inductively defined as follows:

1. if $\pi = C_1? \circ \cdots \circ C_h? \circ S \circ D_1? \circ \cdots \circ D_k?$ (with $h \geq 0$, $k \geq 0$), then $\tau(\pi(x, y)) = C_1(x) \wedge \cdots \wedge C_h(x) \wedge S(x, y) \wedge D_1(y) \wedge \cdots \wedge D_k(y)$;
2. if $\pi = \pi_1 \circ \pi_2$, where $\mathit{length}(\pi_1) = 1$ and $\mathit{length}(\pi_2) \geq 1$, then $\tau(\pi(x, y)) = \tau(\pi_1(x, z)) \wedge \tau(\pi_2(z, y))$, where z is a new variable symbol (i.e., a variable symbol not occurring elsewhere in the query).

Intuitively, $\neg\alpha$ encodes the violation of α by asking for the existence of two distinct instances of C identified, according to α, by the same set of objects. Moreover, if $\mathcal{C} = \{\alpha_1, \ldots, \alpha_m\}$ then $\neg\mathcal{C}$ is the Boolean UCQ with inequalities having the form $\neg\alpha_1 \vee \cdots \vee \neg\alpha_m$.

In the rest of this section, we assume w.l.o.g. that all concepts occurring in (local) IdCs are atomic. By extending the technique in (Calvanese, Eiter, and Ortiz 2007) to deal with inequalities[7], we can construct from $\neg\mathcal{C}$ a 2ATA $A_{\neg\mathcal{C}}$ accepting trees representing interpretations in which $\neg\mathcal{C}$ holds, and hence at least one of the IdCs in $\mathcal{C}$ is violated. Note that, in general it is not sufficient to restrict the attention to tree-shaped models when checking entailment of a UCQ with inequalities[8]. However, Theorem 3 ensures that it is indeed sufficient in the case where the inequalities express the violation of *local* IdCs.

By complementing the 2ATA $A_{\neg\mathcal{C}}$, we obtain a 1NTA $A_{\mathcal{C}}$ accepting the trees representing interpretations in which all IdCs in $\mathcal{C}$ are satisfied. Finally, by intersecting $A_{\mathcal{K}_{in} \not\models q}$ with $A_{\mathcal{C}}$, we obtain a 1NTA $A_{\mathcal{K} \not\models q}$ accepting all trees representing models of $\mathcal{K}$. It follows from results in (Calvanese, Eiter, and Ortiz 2007; Vardi 1998) that $A_{\mathcal{K} \not\models q}$ is a 1NTA whose emptiness can be checked in time double exponential in the size of $\mathcal{K}$ and q. Given the lower bounds in (Lutz 2007), we get the following result.

Theorem 4 *Answering UCQs in $\mathcal{ALCQI}b_{reg}$ with local IdCs is 2*EXPTIME*-complete w.r.t. combined complexity.*

Reasoning in *DL-Lite*$_{\mathcal{A}}$ with IdCs

In this section, we study reasoning in *DL-Lite*$_{\mathcal{A}}$ KBs extended with IdCs. In particular, we show that adding *arbitrary* IdCs to *DL-Lite*$_{\mathcal{A}}$ KBs makes reasoning computationally harder, while adding *local* IdCs to *DL-Lite*$_{\mathcal{A}}$ KBs does not increase the computational complexity of reasoning. We recall that both KB satisfiability and answering UCQs in *DL-Lite*$_{\mathcal{A}}$ are in LOGSPACE with respect to ABox complexity (i.e., with respect to the size of the ABox).

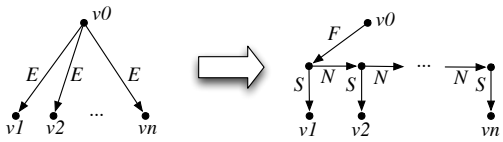

Figure 8: Representation of a graph

First, we provide the definition of *DL-Lite*$_{\mathcal{A}}$ with IdCs.

Definition 5 A KB in *DL-Lite*$_{\mathcal{A}}$ with IdCs is a pair $\langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{A}$ is a *DL-Lite*$_{\mathcal{A}}$ ABox, and $\mathcal{T}$ is the union of two sets $\mathcal{T}'$, $\mathcal{C}$, where $\mathcal{T}'$ is a *DL-Lite*$_{\mathcal{A}}$ TBox, and $\mathcal{C}$ is a set of IdCs such that

- all concepts identified in $\mathcal{C}$ are basic concepts, i.e., the concept C in each IdC (cf. Definition 1) of $\mathcal{C}$ is of the form $\exists Q$, $\delta(U)$, or $\rho(U)$;
- all concepts appearing in the test relations in $\mathcal{C}$ are of the form A, $\exists Q$, $\delta(U)$, $\rho(U)$, $\top_D$, T_1, . . ., or T_n;
- for each IdC α in $\mathcal{C}$, every role or attribute that occurs (in either direct or inverse direction) in a path of α is not specialized in $\mathcal{T}'$, i.e., it does not appear in the right-hand side of assertions of the form $Q \sqsubseteq Q'$ or $U \sqsubseteq U'$.

Notice that the last condition is the natural generalization of the analogous condition enforced on functional roles and attributes in *DL-Lite*$_{\mathcal{A}}$ KBs (cf. Preliminaries).

NLOGSPACE-hardness of *DL-Lite*$_{\mathcal{A}}$ with IdCs

We now show that adding arbitrary IdCs to *DL-Lite*$_{\mathcal{A}}$ makes reasoning computationally harder, even when we add just single-path IdCs.

Theorem 6 *KB satisfiability in DL-Lite*$_{\mathcal{A}}$[9] *with single-path IdCs is* NLOGSPACE*-hard with respect to ABox complexity.*

Proof (sketch). The proof is based on a reduction from reachability in directed graphs, which is NLOGSPACE-hard.

Let $G = \langle V, E \rangle$ be a directed graph. Reachability is the problem of deciding, given two vertices $s, t \in V$ whether there is an oriented path formed by edges in E that, starting from s allows one to reach t. We consider the graph represented through first-child and next-sibling functional relations F, N, S, and denote with V^+ the set of vertices augmented by the nodes used in such a representation (cf. Figure 8).

From G and the two vertices s, t we define the KB $\mathcal{K} = \langle \mathcal{T}, \mathcal{A} \rangle$ as follows:

$$\begin{aligned}\mathcal{A} = {} & \{\mathcal{R}(a, b), \mathcal{R}(a', b') \mid (a, b) \in \mathcal{R}, \text{ for } \mathcal{R} \in \{F, N, S\}\} \cup \\ & \{A(a) \mid a \in V^+\} \cup \\ & \{P(s, \mathit{init}), P(s', \mathit{init}), C(t), C(t')\} \\ \mathcal{T} = {} & \{A \sqsubseteq \exists P\} \cup \{(\mathsf{id}\ C\ P)\} \cup \\ & \{(\mathsf{id}\ \exists P^-\ P^- \circ \mathcal{R}^- \circ P) \mid \mathcal{R} \in \{F, N, S\}\}\end{aligned}$$

[7] A 2ATA can easily check that two variables are being mapped to different nodes of a tree.

[8] Checking entailment of a CQ with a single inequality is undecidable in expressive DLs (Calvanese, De Giacomo, and Lenzerini 1998).

[9] Actually, the result holds for the weaker logic *DL-Lite*$_{\mathcal{F}}$ (Calvanese et al. 2007c).

In other words, we encode in $\mathcal{A}$ two copies of (the representation of) G. In addition, we include in $\mathcal{A}$ the assertions $P(s, init)$ and $P(s', init)$ connecting the two copies of the start vertex s to an individual $init$ that does not correspond to any vertex of (the representation of) G. We also include the assertions $C(t)$ and $C(t')$, which are exploited to encode the reachability test. As for the TBox, we enforce that each individual contributing to the encoding of the two copies of G has an outgoing P edge. Moreover, we enforce through path-identification constraints that each object that is the target of such a P edge is identified by a suitable path. Finally, we enforce the identification constraint (id C P) to imply equality of t and t'. It can be shown that t is reachable from s in G iff $\mathcal{K}$ is unsatisfiable. □

LOGSPACE membership of *DL-Lite*$_\mathcal{A}$ with local IdCs

In the following, we restrict our attention to local IdCs. We first present the algorithm KBSat for KB satisfiability in *DL-Lite*$_\mathcal{A}$ with local IdCs.

Algorithm KBSat($\mathcal{K}$)
Input: *DL-Lite*$_\mathcal{A}$ KB with local IdCs $\mathcal{K} = \langle \mathcal{T} \cup \mathcal{C}, \mathcal{A} \rangle$
Output: true if $\mathcal{K}$ satisfiable, false otherwise
begin
  if $\langle \mathcal{T}, \mathcal{A} \rangle$ is unsatisfiable
  then return false
  else begin
    $q := \neg \mathcal{C}$;
    $q' :=$ *PerfectRef*$(q, \mathcal{T})$;
    return $\langle \emptyset, \mathcal{A} \rangle \not\models q'$
  end
end

The above algorithm decides satisfiability of a *DL-Lite*$_\mathcal{A}$ KB with local IdCs as follows. First, it checks satisfiability of the ordinary *DL-Lite*$_\mathcal{A}$ KB $\langle \mathcal{T}, \mathcal{A} \rangle$ (cf. (Poggi et al. 2008)) obtained from $\mathcal{K}$ by eliminating the local IdCs: if $\langle \mathcal{T}, \mathcal{A} \rangle$ is unsatisfiable, then $\mathcal{K}$ is also unsatisfiable. Otherwise, the algorithm first computes the query $\neg \mathcal{C}$ corresponding to the negation of the local IdCs in $\mathcal{K}$ (cf. previous section). Then, the algorithm computes the query q' corresponding to the *perfect reformulation* of query q with respect to the TBox assertions in $\mathcal{T}$. Due to lack of space, we can only provide an informal description of *PerfectRef*. Such a reformulation is almost identical to the reformulation algorithm reported in (Calvanese et al. 2007c) for the case of UCQs without inequalities: in this modified version, the inequality predicate is considered as a new primitive role, and variables occurring in inequality atoms are never "reduced" (i.e., transformed by unification steps into non-join variables) by the reformulation. The query computed by this reformulation method is still a Boolean UCQ with inequalities.

Finally, the algorithm checks whether the ABox $\mathcal{A}$ entails the Boolean UCQ q' thus computed (as explained e.g., in (Calvanese et al. 2007c), this can be done by simply evaluating q' over $\mathcal{A}$ interpreted as a relational database, which can be done in LOGSPACE w.r.t. the size of $\mathcal{A}$): if this is the case, then the algorithm returns false, because this implies that the ABox violates some IdC that is logically implied by $\mathcal{T} \cup \mathcal{C}$. Otherwise, the algorithm returns true.

The following lemma states correctness of KBSat.

Lemma 7 *Let $\mathcal{K} = \langle \mathcal{T} \cup \mathcal{C}, \mathcal{A} \rangle$ be a DL-Lite$_\mathcal{A}$ KB with local IdCs. Then, KBSat($\mathcal{K}$) returns true if $\mathcal{K}$ is satisfiable, false otherwise.*

Correctness of the algorithm KBSat immediately implies the following upper bound for KB satisfiability.

Theorem 8 *KB satisfiability in DL-Lite$_\mathcal{A}$ with local IdCs is* LOGSPACE *with respect to ABox complexity.*

We now turn our attention to query answering. We remark that in UCQs over *DL-Lite*$_\mathcal{A}$ KBs that we consider, predicates in unary atoms are only basic concepts, and basic or arbitrary value-domains. We start by establishing a fundamental "separation" property for local IdCs in *DL-Lite*$_\mathcal{A}$.

Theorem 9 *Let $\mathcal{K} = \langle \mathcal{T} \cup \mathcal{C}, \mathcal{A} \rangle$ be a DL-Lite$_\mathcal{A}$ KB with local IdCs, and let q be a Boolean UCQ. If $\mathcal{K}$ is satisfiable, then $\mathcal{K} \models q$ iff $\langle \mathcal{T}, \mathcal{A} \rangle \models q$.*

Proof (sketch). We use here the notion of *chase* of a *DL-Lite*$_\mathcal{A}$ KB $\langle \mathcal{T}, \mathcal{A} \rangle$, denoted by *chase*$(\langle \mathcal{T}, \mathcal{A} \rangle)$ (see (Poggi et al. 2008; Calvanese et al. 2007c)). We prove that, if $\mathcal{K}$ is satisfiable, then *chase*$(\langle \mathcal{T}, \mathcal{A} \rangle)$ is a model of $\mathcal{C}$. In fact, since roles and attributes occurring in IdCs cannot be specialized, the following crucial property holds: for each role or attribute S and for each labeled null n, in *chase*$(\langle \mathcal{T}, \mathcal{A} \rangle)$ there is at most one fact of the form $S(n, t)$ and at most one fact of the form $S(t, n)$, where t is any constant or labeled null value. From this property, it immediately follows that a local IdC (recall that a local IdC has at least one path of length 1) can *not* be violated by a labeled null value in *chase*$(\langle \mathcal{T}, \mathcal{A} \rangle)$. On the other hand, if a local IdC were violated in *chase*$(\langle \mathcal{T}, \mathcal{A} \rangle)$ by some pair of constants, then $\mathcal{K}$ would be unsatisfiable, thus contradicting the hypothesis. Consequently, no IdC in $\mathcal{C}$ is violated in *chase*$(\langle \mathcal{T}, \mathcal{A} \rangle)$, thus *chase*$(\langle \mathcal{T}, \mathcal{A} \rangle)$ is a model of $\mathcal{C}$, and hence a model of $\mathcal{K}$. Now, as shown in (Calvanese et al. 2007c), for every Boolean UCQ q, $\langle \mathcal{T}, \mathcal{A} \rangle \models q$ iff *chase*$(\langle \mathcal{T}, \mathcal{A} \rangle) \models q$. Thus, if $\langle \mathcal{T}, \mathcal{A} \rangle \not\models q$ then $\mathcal{K} \not\models q$. On the other hand, trivially if $\mathcal{K} \not\models q$ then $\langle \mathcal{T}, \mathcal{A} \rangle \not\models q$, hence the claim follows. □

The above theorem indicates that answering UCQs in *DL-Lite*$_\mathcal{A}$ with local IdCs can be done by first verifying KB satisfiability, and then executing query answering on the *DL-Lite*$_\mathcal{A}$ KB obtained by eliminating the IdCs. In other words, we have the following remarkable property: to answer UCQs over *DL-Lite*$_\mathcal{A}$ KBs with local IdCs, we can essentially reuse the same machinery developed for ordinary *DL-Lite*$_\mathcal{A}$ KBs. The above property and the known computational properties of query answering in *DL-Lite*$_\mathcal{A}$ immediately imply the following theorem.

Theorem 10 *Answering UCQs in DL-Lite$_\mathcal{A}$ with local IdCs is* LOGSPACE *with respect to ABox complexity.*

Finally, it can be shown that logical implication of TBox assertions (including identification constraints) in *DL-Lite*$_\mathcal{A}$ with local IdCs is polynomial, i.e., adding local IdCs to *DL-Lite*$_\mathcal{A}$ KBs does not increase the computational complexity of TBox reasoning.

Conclusions

Motivated by the importance of modeling identification in ontologies, we have presented a study on the notion of path-based identification constraints in DLs. The results described in this paper provide the basis for extending current DL reasoners with suitable capabilities for dealing with identification constraints, both in the case of expressive DLs, and in the case of tractable DLs. As for the latter, we have incorporated local path-based identification constraints in the *DL-Lite*$_{\mathcal{A}}$ reasoner QuOnto (Acciarri et al. 2005), implementing the technique illustrated in the previous section. For the same reasoner, we are planning to add a method for dealing with non-local path-based identification constraints, in particular by treating such constraints as epistemic formulae, so as to essentially restrict their scope to the set of named individuals in the knowledge base.

Acknowledgments

This research has been partially supported by the FET project TONES (Thinking ONtologiES), funded by the EU in the 6th Framework Programme under contract number FP6-7603, and by the MIUR FIRB 2005 project "Tecnologie Orientate alla Conoscenza per Aggregazioni di Imprese in Internet" (TOCAI.IT).

References

Abiteboul, S.; Hull, R.; and Vianu, V. 1995. *Foundations of Databases*. Addison Wesley Publ. Co.

Acciarri, A.; Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; Palmieri, M.; and Rosati, R. 2005. QUONTO: QUerying ONTOlogies. In *Proc. of AAAI 2005*, 1670–1671.

Amoroso, A.; Esposito, G.; Lembo, D.; Urbano, P.; and Vertucci, R. 2008. Ontology-based data integration with MASTRO-I for configuration and data management at SELEX Sistemi Integrati. Submitted for publication.

Baader, F.; Calvanese, D.; McGuinness, D.; Nardi, D.; and Patel-Schneider, P. F., eds. 2003. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press.

Bechhofer, S.; van Harmelen, F.; Hendler, J.; Horrocks, I.; McGuinness, D. L.; Patel-Schneider, P. F.; and Stein, L. A. 2004. OWL Web Ontology Language reference. W3C Recommendation. Available at `http://www.w3.org/TR/owl-ref/`.

Berger, R. 1966. The undecidability of the dominoe problem. *Mem. Amer. Math. Soc.* 66:1–72.

Borgida, A., and Weddell, G. E. 1997. Adding uniqueness constraints to description logics (preliminary report). In *Proc. of DOOD'97*, 85–102.

Brodie, M. L.; Mylopoulos, J.; and Schmidt, J. W., eds. 1984. *On Conceptual Modeling: Perspectives from Artificial Intelligence, Databases, and Programming Languages*. Springer.

Calvanese, D., and De Giacomo, G. 2003. Expressive description logics. In Baader et al. (2003). chapter 5, 178–218.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; Poggi, A.; and Rosati, R. 2006a. Linking data to ontologies: The description logic *DL-Lite*$_A$. In *Proc. of OWLED 2006*.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2006b. Data complexity of query answering in description logics. In *Proc. of KR 2006*, 260–270.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007a. Can OWL model football leagues? In *Proc. of OWLED 2007*.

Calvanese, D.; De Giacomo, G.; Lembo, D.; Lenzerini, M.; and Rosati, R. 2007b. EQL-Lite: Effective first-order query processing in description logics. In *Proc. of IJCAI 2007*, 274–279.

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

Calvanese, D.; De Giacomo, G.; and Lenzerini, M. 1995. Structured objects: Modeling and reasoning. In *Proc. of DOOD'95*, volume 1013 of *LNCS*, 229–246. Springer.

Calvanese, D.; De Giacomo, G.; and Lenzerini, M. 1998. On the decidability of query containment under constraints. In *Proc. of PODS'98*, 149–158.

Calvanese, D.; De Giacomo, G.; and Lenzerini, M. 2001. Identification constraints and functional dependencies in description logics. In *Proc. of IJCAI 2001*, 155–160.

Calvanese, D.; Eiter, T.; and Ortiz, M. 2007. Answering regular path queries in expressive description logics: An automata-theoretic approach. In *Proc. of AAAI 2007*, 391–396.

Chen, P. P. 1976. The Entity-Relationship model: Toward a unified view of data. *ACM Trans. on Database Systems* 1(1):9–36.

De Giacomo, G., and Lenzerini, M. 1995. What's in an aggregate: Foundations for description logics with tuples and sets. In *Proc. of IJCAI'95*, 801–807.

Fowler, M., and Scott, K. 1997. *UML Distilled – Applying the Standard Object Modeling Laguage*. Addison Wesley Publ. Co.

Hull, R. B., and King, R. 1987. Semantic database modelling: Survey, applications and research issues. *ACM Computing Surveys* 19(3):201–260.

Lutz, C.; Areces, C.; Horrocks, I.; and Sattler, U. 2005. Keys, nominals, and concrete domains. *J. of Artificial Intelligence Research* 23:667–726.

Lutz, C. 2007. Inverse roles make conjunctive queries hard. In *Proc. of DL 2007*, volume 250 of *CEUR Electronic Workshop Proceedings,* `http://ceur-ws.org/Vol-250/`, 100–111.

Nguyen, T. D. T., and Le Thanh, N. 2007. Integrating identification constraints in web ontology. In *Proc. of the 9th Int. Conf. on Enterprise Information Systems*, 338–343.

Poggi, A.; Lembo, D.; Calvanese, D.; De Giacomo, G.; Lenzerini, M.; and Rosati, R. 2008. Linking data to ontologies. *J. on Data Semantics* X:133–173.

Toman, D., and Weddell, G. E. 2005. On the interaction between inverse features and path-functional dependencies in description logics. In *Proc. of IJCAI 2005*, 603–608.

Toman, D., and Weddell, G. E. 2008. On keys and functional dependencies as first-class citizens in description logics. *J. of Automated Reasoning* 40(2–3):117–132.

van Emde Boas, P. 1997. The convenience of tilings. In Sorbi, A., ed., *Complexity, Logic, and Recursion Theory*, volume 187 of *Lecture Notes in Pure and Applied Mathematics*. Marcel Dekker Inc. 331–363.

Vardi, M. Y., and Wolper, P. 1986. Automata-theoretic techniques for modal logics of programs. *J. of Computer and System Sciences* 32:183–221.

Vardi, M. Y. 1998. Reasoning about the past with two-way automata. In *Proc. of ICALP'98*, volume 1443 of *LNCS*, 628–641. Springer.

Welty, C., and Guarino, N. 2001. Support for ontological analysis of taxonomic relationships. *Data and Knowledge Engineering* 39(1):51–74.