Towards Higher-Order *DL-Lite* (preliminary report)

Giuseppe De Giacomo, Maurizio Lenzerini, Riccardo Rosati

Dipartimento di Informatica e Sistemistica "Antonio Ruberti"
Sapienza Università di Roma
Via Ariosto 25, I-00183 Roma, Italy
lastname@dis.uniroma1.it

Abstract. In several application domains, the need arises of modeling and reasoning about meta-concepts and meta-properties. This is the case, for example, with information system interoperability, where a description of the data schema (or, the ontology) of one system should co-exist with the specification of the application domain described by the data schema itself. In logic, higher-order constructs are needed for a correct representation of concepts and properties at the meta-level. Current research on Description Logics mainly focuses on their ability for specifying the domain of interest, while the issue of devising suitable extensions to these logics for representing and reasoning about meta-level elements is largely unexplored. In this paper, we present the first results of our investigation on extending DL-Lite with higher-order capabilities. We show that basic higher-order constructs can be safely added to DL-lite while keeping all the reasoning tasks tractable, including answering conjunctive queries mixing object-level and meta-level elements.

1 Introduction

In many applications, the need arises of modeling and reasoning about meta-concepts and meta-properties. Roughly speaking, a meta-concept is a concept whose instances can be themselves concepts, and a meta-property is a relationship between meta-concepts. Meta-concept representation is needed, for example, for information system interoperability, where a description of the data schema (or, the ontology) of one system should co-exist with the specification of the application domain represented by the data schema itself. Another context where meta-level knowledge is important is formal ontology, where specific meta-properties (e.g., rigidity) are used to express relevant aspects of the intended meaning of the elements in an ontology. Meta-properties in such scenario typically impose constraints on the taxonomic structure of the concepts and the roles represented in the object-level part of the ontology [5], and reasoning about meta-properties can be useful for checking whether the ontology is "correct" with respect to specific methodological criteria.

The idea of representing concepts and properties at the meta-level is an old one in Knowledge Representation and Computer Science. Semantic networks, early Frame-based and Description-based systems incorporated specific mechanisms for representing concepts whose instances are themselves concepts [9, 1]. Conceptual modeling languages proposed in the 70's, such as TAXIS [11], provided both the notion of meta-class, and suitable facilities for describing properties of operators on meta-classes. The

notion of meta-class is also present in virtually all object-oriented languages, including modern programming languages (see, for instance, the Java class "class").

It is well-known that in logic, higher-order constructs are needed for a correct representation of concepts and properties at the meta-level. However, current research on Description Logics mainly focuses on their ability of specifying the domain of interest, and the issue of devising suitable extensions to these logics for representing and reasoning about meta-level elements is largely unexplored.

In this paper, we present the first results of our investigation on extending Description Logics of the DL-Lite family [4] with higher-order capabilities. The DLs belonging to this family are tailored towards capturing basic conceptual and ontological modeling constructs while keeping reasoning, including conjunctive query answering, tractable and first-order reducible (i.e., LOGSPACE in ABox complexity). The main outcome of our investigation is that basic higher-order constructs can be safely added to DL-lite while keeping all the reasoning tasks tractable, including answering conjunctive queries mixing object-level and meta-level elements. Specifically, we present the following contributions:

- We illustrate a proposal for extending *DL-Lite*$_{\mathcal{R}}$[1] with meta-level constructs. In the resulting logic, called *HiDL-Lite*, the distinction between objects, concepts and roles, is blurred, as every element of the ontology can be viewed in these three ways. Also, concepts and roles may occur in membership assertions as arguments (a position that is normally reserved to objects only), with no limitations on the instantiation strata. The notion of conjunctive query is also extended to accomodate atoms mixing the object-level and the meta-level.
- We describe an algorithm for answering conjunctive queries over *HiDL-Lite* knowledge bases. The algorithm actually works for a fairly general class of conjunctive queries, called ISA-ground, where all variables appearing as arguments of ISA-atoms are free. We show that the complexity of the algorithm is polynomial with respect to the size of the knowledge base, and LOGSPACE with respect to the size of the pure membership assertions (i.e., membership assertions on the elements of the ontology that are not used as predicates).

As we said before, the issue of extending DLs with higher-order constructs is largely unexplored. [2] is probably the first paper on this subject. The author studies "reification of concepts", which is a means to express meta-level classes, but the paper does not address the issue of meta-properties, nor the issue of query answering. Our work has connections with recent investigations on web languages, such as RDF and RDFS, where meta-modeling is one important subject. In [10], the author addresses the issue of decidability of reasoning on meta-properties in different fragments of OWL Full. While some of the fragments are decidable, the paper does not deal with conjunctive query answering. Moreover, the focus of [10] is not on tractability of reasoning. In [13], an analysis of reasoning in RDFS is carried out. The main difference with our work, however, is that in [13], the isa-predicate is interpreted under the "if" semantics (if c is a subclass of d, then the extension of c is a subset of the extension of d), while our

[1] For the sake of simplicity, we ignore *DL-Lite*$_{\mathcal{R}}$ disjointness assertions in this paper, but they could actually be safely included in our study.

semantics is based on the "exact" interpretation of the isa-predicate (i.e., the extension of c is a subset of the extension of d if and only if c is a subclass of d). Finally, compared to F-logic [7], in our approach higher-order constructs are added to a logic where incomplete information can be explicitly represented (as usual in DLs).

The paper is organized as follows. In Section 2 we introduce syntax and semantics of *HiDL-Lite*. In Section 3 we illustrate some simple examples of its usage. In Section 4 we describe our reasoning techniques, and discuss their computational complexity. Section 5 ends the paper with a brief description of future work.

2 *HiDL-Lite*

In this section we define the DL *HiDL-Lite*, which can be seen as an extension of *DL-Lite*$_\mathcal{R}$ [4] with meta-modeling capabilities.

2.1 Syntax of *HiDL-Lite*

We assume to have an alphabet of symbols $\mathcal{N}$, called *names*, and an alphabet of *variables* $\mathcal{V}$ (disjoint from $\mathcal{N}$). Symbols in $\mathcal{N} \cup \mathcal{V}$ are called *atomic elements*; they simultaneously denote objects, (atomic) concepts, and (atomic) roles in traditional DL terms. From $\mathcal{N}$ and $\mathcal{V}$ we define the set of *element terms*, or simply *elements*, as the set $\mathcal{E} = \mathcal{E}(\mathcal{N}) \cup \mathcal{E}(\mathcal{V})$, where

- $\mathcal{E}(\mathcal{N})$ is the set of expressions defined by the following abstract syntax:

$$E \leftarrow A \mid A^- \mid \exists A \mid \exists A^-$$

  where $A \in \mathcal{N}$, and
- $\mathcal{E}(\mathcal{V})$ is the set of expressions defined by the above syntax in the case where $A \in \mathcal{V}$.

Intuitively, an element term denotes an atomic element, the inverse of an atomic element, or the projection of an atomic element on either the first or the second component.

We can now turn our attention to *atomic formulas*, or simply *atoms*, over elements. These are formulas built according to the following syntax (where each e_i is an element from $\mathcal{E}$):

$$e_1 \sqsubseteq_\mathcal{C} e_2 \mid e_1 \sqsubseteq_\mathcal{R} e_2 \mid e_1(e_2) \mid e_1(e_2, e_3)$$

In the case of atoms of the form $e_1(e_2)$ and $e_1(e_2, e_3)$, we call the position of e_1 the *predicate position*, whereas we call the positions of e_2, e_3 *argument positions*. Also, we call *ISA-atoms* atoms of the form $e_1 \sqsubseteq_\mathcal{C} e_2$ or $e_1 \sqsubseteq_\mathcal{R} e_2$, and we say that an atom is *ground* if no variables occur in it.

A *knowledge base* (KB) is a finite set of ground atoms. A conjunctive query (CQ) over a KB is an open formula of the form:

$$q(\boldsymbol{x}) \leftarrow \exists \boldsymbol{y}.\varphi(\boldsymbol{x}, \boldsymbol{y})$$

where $\varphi(\boldsymbol{x}, \boldsymbol{y})$ is a conjunction of atoms whose variables range over the free variables $\boldsymbol{x}$ and the existentially quantified variables $\boldsymbol{y}$. A CQ is *Boolean* if it has no free variables, i.e., $\boldsymbol{x} = \epsilon$.

2.2 Semantics of *HiDL-Lite*

An interpretation domain Δ is a non-empty (possibly countably infinite) set. Given a domain Δ, an *interpretation function* over Δ is a triple $\mathcal{I} = \langle \mathcal{I}_o, \mathcal{I}_c, \mathcal{I}_r \rangle$ where:

- $\mathcal{I}_o$ is a function that maps each element of $\mathcal{E}(\mathcal{N})$ to a single domain object of Δ;
- $\mathcal{I}_c$ is a function that maps each $d \in \Delta$ into a subset of Δ;
- $\mathcal{I}_r$ is a function that maps each $d \in \Delta$ into a subset of $\Delta \times \Delta$.

The above functions are used to provide a triple interpretation for all elements in $\mathcal{E}(\mathcal{N})$. In particular, each such element e is interpreted as:

- a single object of Δ by $\mathcal{I}_o$,
- as a subset of Δ through the composition of $\mathcal{I}_o$ with $\mathcal{I}_c$,
- as a set of binary tuples from Δ through the composition of $\mathcal{I}_o$ and $\mathcal{I}_r$.

An *interpretation* is a pair $\langle \Delta, \mathcal{I} \rangle$ where Δ is an interpretation domain and $\mathcal{I}$ is an interpretation function $\langle \mathcal{I}_o, \mathcal{I}_c, \mathcal{I}_r \rangle$ over Δ such that, for every name $A \in \mathcal{N}$, the following conditions hold:

- $((\exists A)^{\mathcal{I}_o})^{\mathcal{I}_c} = \{\langle d \rangle \mid \langle d, d' \rangle \in (A^{\mathcal{I}_o})^{\mathcal{I}_r}\}$;
- $((\exists A^-)^{\mathcal{I}_o})^{\mathcal{I}_c} = \{\langle d' \rangle \mid \langle d, d' \rangle \in (A^{\mathcal{I}_o})^{\mathcal{I}_r}\}$;
- $((A^-)^{\mathcal{I}_o})^{\mathcal{I}_r} = \{\langle d', d \rangle \mid \langle d, d' \rangle \in (A^{\mathcal{I}_o})^{\mathcal{I}_r}\}$.

With a little abuse of notation, in the following we will denote an interpretation by its interpretation function.

Besides interpretations, we need assignments to interpret elements of $\mathcal{E}(\mathcal{V})$. Given an interpretation $\mathcal{I}$, an *$\mathcal{I}$-assignment* is a function $\mu : \mathcal{E}(\mathcal{V}) \rightarrow \Delta$ such that the following conditions hold:

- for every variable $x \in \mathcal{V}$, $\mu(\exists x)^{\mathcal{I}_c} = \{d \mid \langle d, d' \rangle \in \mu(x)^{\mathcal{I}_r}\}$;
- for every variable $x \in \mathcal{V}$, $\mu(\exists x^-)^{\mathcal{I}_c} = \{d \mid \langle d', d \rangle \in \mu(x)^{\mathcal{I}_r}\}$;
- for every variable $x \in \mathcal{V}$, $\mu(x^-)^{\mathcal{I}_c} = \{\langle d, d' \rangle \mid \langle d', d \rangle \in \mu(x)^{\mathcal{I}_r}\}$.

Satisfaction of an atom with respect to an interpretation $\mathcal{I}$ and an $\mathcal{I}$-assignment μ is defined as follows:

- for each element $e \in \mathcal{E}(\mathcal{N}) \cup \mathcal{E}(\mathcal{V})$, if $e \in \mathcal{E}(\mathcal{N})$ then $e^{\mathcal{I}_o,\mu} = e^{\mathcal{I}_o}$, otherwise $e^{\mathcal{I}_o,\mu} = \mu(e)$:
- $\mathcal{I}, \mu \models e_1 \sqsubseteq_{\mathcal{C}} e_2$ iff $(e_1^{\mathcal{I}_o,\mu})^{\mathcal{I}_c} \subseteq (e_2^{\mathcal{I}_o,\mu})^{\mathcal{I}_c}$;
- $\mathcal{I}, \mu \models e_1 \sqsubseteq_{\mathcal{R}} e_2$ iff $(e_1^{\mathcal{I}_o,\mu})^{\mathcal{I}_r} \subseteq (e_2^{\mathcal{I}_o,\mu})^{\mathcal{I}_r}$;
- $\mathcal{I}, \mu \models e_1(e_2)$ iff $e_1^{\mathcal{I}_o,\mu} \in (e_2^{\mathcal{I}_o,\mu})^{\mathcal{I}_c}$;
- $\mathcal{I}, \mu \models e_1(e_2, e_3)$ iff $\langle e_1^{\mathcal{I}_o,\mu}, e_2^{\mathcal{I}_o,\mu} \rangle \in (e_3^{\mathcal{I}_o,\mu})^{\mathcal{I}_r}$.

Notice that ground atoms are independent of the $\mathcal{I}$-assignment μ, so we simply say that a ground atom α is satisfied by $\mathcal{I}$ (without mentioning μ).

A KB $\mathcal{K}$ is satisfied by $\mathcal{I}$ iff all (ground) atoms in $\mathcal{K}$ are satisfied by $\mathcal{I}$. As usual, the interpretations $\mathcal{I}$ such that $\mathcal{K}$ is satisfied by $\mathcal{I}$ are called the *models* of $\mathcal{K}$.

A KB $\mathcal{K}$ is satisfiable if and only if it has at least one model. We say that $\mathcal{K}$ logically implies an atomic formula α, if and only if α is satisfied by all models of $\mathcal{K}$. The following property immediately follows from the previous definitions.

Proposition 1. *Every HiDL-Lite KB is satisfiable.*

Given a Boolean CQ q of the form $q \leftarrow \exists \boldsymbol{y}.\varphi(\boldsymbol{y})$, an interpretation $\mathcal{I}$ and an $\mathcal{I}$-assignment μ, we say that q is satisfied in $\mathcal{I}$ and μ if and only if all atoms in $\varphi(\boldsymbol{y})$ are satisfied by $\mathcal{I}, \mu$.

Given a Boolean CQ q and a KB $\mathcal{K}$, we say that q is logically implied by $\mathcal{K}$ (denoted by $\mathcal{K} \models q$) if and only if for each model $\mathcal{I}$ of $\mathcal{K}$ there exists an $\mathcal{I}$-assignment μ compatible such that q is satisfied by $\mathcal{I}$ and μ.

Given a non-Boolean CQ q of the form $q(\boldsymbol{x}) \leftarrow \exists \boldsymbol{y}.\varphi(\boldsymbol{x}, \boldsymbol{y})$, with $\boldsymbol{x} = \langle x_1, \ldots, x_n \rangle$, a *grounding tuple* of q is a tuple $\boldsymbol{E} = \langle E_1, \ldots, E_n \rangle$ of ground elements from $\mathcal{E}(\mathcal{N})$ such that if x_i occurs in an atom of the form $\exists x_i$ in q, then the element E_i is of the form A or A^- with $A \in \mathcal{N}$. We denote by $q_{\boldsymbol{E}} \leftarrow \exists \boldsymbol{y}.\varphi(\boldsymbol{E}, \boldsymbol{y})$ the Boolean CQ, obtained from q by substituting $\boldsymbol{x}$ with $\boldsymbol{E}$. The set of *certain answers* to q in $\mathcal{K}$ (denoted by $ans(q, \mathcal{K})$) is the set of grounding tuples $\boldsymbol{E}$ that make $q_{\boldsymbol{E}} \leftarrow \exists \boldsymbol{y}.\varphi(\boldsymbol{E}, \boldsymbol{y})$ logically implied by $\mathcal{K}$.

3 Examples of meta-modeling in *HiDL-Lite*

In this section, we illustrate the modeling capabilities of *HiDL-Lite* through examples. Let us consider the following simple KB $\mathcal{K}_1$ expressed in *DL-Lite*$_{\mathcal{R}}$:

$$\begin{array}{l} \exists Member \sqsubseteq_{\mathcal{C}} Employee \\ \exists Member^- \sqsubseteq_{\mathcal{C}} Dept \\ Employee \sqsubseteq_{\mathcal{C}} \exists Member \\ Director \sqsubseteq_{\mathcal{R}} Member \\ \exists Director \sqsubseteq_{\mathcal{C}} Manager \\ Dept \sqsubseteq_{\mathcal{C}} \exists Director^- \\ Manager \sqsubseteq_{\mathcal{C}} Employee \\ Director(John, Research) \end{array}$$

Interesting queries that we can pose to $\mathcal{K}_1$ by using the constructs of *HiDL-Lite* include, for example:

– Return all roles $John$ participates in as first component:

$$q(x) \leftarrow \exists y.x(John, y)$$

which returns $Director, Member$.

Suppose now that we want to describe the above knowledge base in terms of a set of meta-concepts and meta-properties, specified by the following *HiDL-Lite* knowledge base $\mathcal{K}_2$:

$$
\begin{array}{l}
Concept \sqsubseteq_{\mathcal{C}} Element \\
Role \sqsubseteq_{\mathcal{C}} Element \\
RoleDomain \sqsubseteq_{\mathcal{R}} Concept \\
RoleRange \sqsubseteq_{\mathcal{R}} Concept \\
\exists HasName \sqsubseteq_{\mathcal{C}} Concept \\
\exists HasName^{-} \sqsubseteq_{\mathcal{C}} Name \\
\exists RN \sqsubseteq_{\mathcal{C}} Name \\
\exists RN^{-} \sqsubseteq_{\mathcal{C}} Name \\
SY \sqsubseteq_{\mathcal{R}} RN \\
HY \sqsubseteq_{\mathcal{R}} RN \\
RT \sqsubseteq_{\mathcal{R}} RN
\end{array}
$$

Note that $\mathcal{K}_2$ includes usual meta-level concepts (such as the concept "concept"), as well as concepts and relationships used to model a set of lexical terms. In particular, lexical relations of interest to $\mathcal{K}_2$ are the *synonym (SY)*, the *hyperonym (HY)*, and the *related-term (RT)* relations.

The description of $\mathcal{K}_1$ in terms of $\mathcal{K}_2$ corresponds to the following set of membership assertions expressed in *HiDL-Lite*:

$$
\begin{array}{l}
RoleDomain(Member, Employee) \\
RoleRange(Member, Dept) \\
RoleDomain(Director, Manager) \\
RoleRange(Director, Dept) \\
HasName(Employee, \text{“}Employee\text{”}) \\
SY(\text{“}Impiegato\text{”}, \text{“}Employee\text{”}) \\
HY(\text{“}Worker\text{”}, \text{“}Employee\text{”}) \\
RT(\text{“}Worker\text{”}, \text{“}Job\text{”}) \\
\ldots
\end{array}
$$

Now we can ask queries to $\mathcal{K}_1 \cup \mathcal{K}_2$ mixing the meta-level, the linguistic level, and the conceptual level. For example:

- Classify all concepts:

  $$q(x_1, x_2) \leftarrow x_1 \sqsubseteq_{\mathcal{C}} x_2 \wedge Concept(x_1) \wedge Concept(x_2)$$

  which returns the classification of the concepts explicitly named in $\mathcal{K}_2$, namely: $(Employee, Employee), (Manager, Employee), (Manager, Manger), (Dept, Dept)$.
- Return all pairs of equivalent roles to which $John$ participates:

  $$q(x_1, x_2) \leftarrow \exists y. x_1 \sqsubseteq_{\mathcal{R}} x_2 \wedge x_2 \sqsubseteq_{\mathcal{R}} x_1 \wedge x_1(John, y) \wedge Role(x_1), Role(x_2)$$

  which in our case returns again $(Director, Director)$, $(Member, Member)$.
- Return all synonyms of concepts:

  $$q(x, z) \leftarrow \exists y. HasName(z, y) \wedge SY(x, y)$$

– Return all concepts denoted by hyperonyms of the name of concept $Employee$:

$$q(x) \leftarrow \exists y_1, y_2.HasName(Employee, y_1) \wedge HY(y_2, y_1) \wedge HasName(x, y_2)$$

– Return all pairs of subsumer-subsumee whose names are one the hyperonym of the other:

$$q(x_1, x_2) \leftarrow \exists y_1, y_2.x_1 \sqsubseteq_{\mathcal{C}} x_2 \wedge HasName(x_1, y_1) \wedge HY(y_2, y_1) \wedge HasName(x_2, y_2)$$

– Return all pairs of elements $\langle \alpha, \beta \rangle$ such that α is an instance of β, and whose names are one the hyperhonym of the other:

$$q(x_1, x_2) \leftarrow \exists y_1, y_2.x_2(x_1) \wedge HasName(x_1, y_1) \wedge HY(y_2, y_1) \wedge HasName(x_2, y_2)$$

– Return all concepts whose names are *related-term (RT)*, and are indeed linked by a role in the ontology, returning the role as well:

$$q(x_1, x_2, x_3) \leftarrow \exists y_1, y_2. \exists x_3 \sqsubseteq_{\mathcal{C}} x_1 \wedge \exists x_3^- \sqsubseteq_{\mathcal{C}} x_2 \wedge HasName(x_1, y_1) \wedge HasName(x_2, y_2) \wedge RT(y_1, y_2)$$

4 Query answering

In this section we sketch query answering, and in fact other forms of reasoning, in *HiDL-Lite*. Our aim is to provide upper bounds for reasoning in *HiDL-Lite*.

4.1 Reduction to *DL-Lite*$_{\mathcal{R}}$

We provide a query answering technique based on reducing answering CQs in *HiDL-Lite* to answering CQs in *DL-Lite*$_{\mathcal{R}}$. Our technique works for the class of ISA-ground CQs, which is defined as follows: a CQ q is *ISA-ground* if all its ISA-atoms are ground.

From now on, we assume that every CQ is ISA-ground. Moreover, and without loss of generality, we assume that a CQ is a Boolean CQ.

Preliminary definitions Given a KB $\mathcal{K}$, we denote by $\mathcal{N}(\mathcal{K})$ the set of names occurring in $\mathcal{K}$. Then, we call *active domain of* $\mathcal{K}$ (denoted by *adom*$(\mathcal{K})$) the set of elements built over the alphabet $\mathcal{N}(\mathcal{K})$. Without loss of generality, from now on we assume that *adom*$(\mathcal{K})$ is non-empty, i.e., that $\mathcal{K}$ is non-empty (this condition can always be satisfied by adding a trivial ISa-atom, e.g., $A \sqsubseteq_{\mathcal{C}} A$).

Given a CQ q, a variable x is a *metavariable* of q if x occurs in a predicate position in q. Otherwise, x is called an *object variable*. E.g., in the query $q(x) \leftarrow \exists y, z.C(x) \wedge x(y, z) \wedge A \sqsubseteq_{\mathcal{C}} z$, the variables x and z are metavariables, while y is an object variable.

We call a CQ *metaground* if it does not contain metavariables.

Given a KB $\mathcal{K}$ and a CQ q, we denote by *MG*$(q, \mathcal{K})$ the set of conjunctive queries corresponding to all the ground instantiations of meta-variables in q with elements in *adom*$(\mathcal{K})$. Obviously, every CQ in *MG*$(q, \mathcal{K})$ is metaground.

We are now able to establish an important property for ISA-ground CQs.

Proposition 2. *For every KB* $\mathcal{K}$ *and for every ISA-ground CQ* q, $\mathcal{K} \models q$ *iff there exists* $q' \in$ *MG*$(q, \mathcal{K})$ *such that* $\mathcal{K} \models q'$.

KB translation to *DL-Lite*$_\mathcal{R}$ Given a *HiDL-Lite* KB $\mathcal{K}$, we define a *DL-Lite*$_\mathcal{R}$ KB $DL(\mathcal{K})$ as follows.

Let $\mathcal{N}$ be a set of names (disjoint from $adom(\mathcal{K})$) such that $|\mathcal{N}| = |adom(\mathcal{K})|$ and let $\nu : adom(\mathcal{K}) \rightarrow \mathcal{N}$ be a bijective function. Then, we define the sets

$$\mathcal{N}^o = \{n^o \mid n = \nu(e) \text{ for some } e \in adom(\mathcal{K})\}$$
$$\mathcal{N}^c = \{n^c \mid n = \nu(e) \text{ for some } e \in adom(\mathcal{K})\}$$
$$\mathcal{N}^r = \{n^r \mid n = \nu(e) \text{ for some } e \in adom(\mathcal{K})\}$$

and the functions $\nu_o : adom(\mathcal{K}) \rightarrow \mathcal{N}^o$, $\nu_c : adom(\mathcal{K}) \rightarrow \mathcal{N}^c$, $\nu_r : adom(\mathcal{K}) \rightarrow \mathcal{N}^r$, as follows:

- for each $e \in adom(\mathcal{K})$, $\nu_o(e) = n^o$ where $n = \nu(e)$;
- for each $e \in adom(\mathcal{K})$, $\nu_c(e) = n^c$ where $n = \nu(e)$;
- for each $e \in adom(\mathcal{K})$, $\nu_r(e) = n^r$ where $n = \nu(e)$.

The *DL-Lite*$_\mathcal{R}$ KB $DL(\mathcal{K}) = \langle \mathcal{T}, \mathcal{A} \rangle$ is defined over the alphabet of individual names $\mathcal{N}^o$, the alphabet of concept names $\mathcal{N}^c$, and the alphabet of role names $\mathcal{N}^r \cup \{isa_C, isa_R\}$ as follows:

- for every assertion of the form $e_1(e_2) \in \mathcal{K}$, $\nu_c(e_1)(\nu_o(e_2)) \in \mathcal{A}$;
- for every assertion of the form $e_1(e_2, e_3) \in \mathcal{K}$, $\nu_r(e_1)(\nu_o(e_2), \nu_o(e_3)) \in \mathcal{A}$;
- for every assertion of the form $e_1 \sqsubseteq_\mathcal{C} e_2 \in \mathcal{K}$, $\nu_c(e_1) \sqsubseteq \nu_c(e_2) \in \mathcal{T}$;
- for every assertion of the form $e_1 \sqsubseteq_\mathcal{R} e_2 \in \mathcal{K}$, $\nu_r(e_1) \sqsubseteq \nu_r(e_2) \in \mathcal{T}$;
- for every name A occurring in $adom(\mathcal{K})$, the following inclusion assertions belong to $\mathcal{T}$:
  - $\nu_c(\exists A) \sqsubseteq \exists \nu_r(A)$;
  - $\exists \nu_r(A) \sqsubseteq \nu_c(\exists A)$;
  - $\nu_c(\exists A^-) \sqsubseteq \exists (\nu_r(A))^-$;
  - $\exists (\nu_r(A))^- \sqsubseteq \nu_c(\exists A^-)$;
  - $\nu_r(A^-) \sqsubseteq (\nu_r(A))^-$;
  - $(\nu_r(A))^- \sqsubseteq \nu_r(A^-)$.
- for each pair of elements e_1, e_2:
  - if $\mathcal{T} \models_{\textit{DL-Lite}_\mathcal{R}} \nu_c(e_1) \sqsubseteq \nu_c(e_2)$, then $isa_C(\nu_o(e_1), \nu_o(e_2)) \in \mathcal{A}$;
  - if $\mathcal{T} \models_{\textit{DL-Lite}_\mathcal{R}} \nu_r(e_1) \sqsubseteq \nu_r(e_2)$, then $isa_R(\nu_o(e_1), \nu_o(e_2)) \in \mathcal{A}$.

We remark that, in the above definition, the symbol $\models_{\textit{DL-Lite}_\mathcal{R}}$ denotes entailment in the description logic *DL-Lite*$_\mathcal{R}$.

Query translation to *DL-Lite*$_\mathcal{R}$ Finally, given a metaground CQ q, we denote by $\tau(q)$ the CQ obtained as follows:

- replace every atom of the form $e_1(e_2)$ with the atom $\nu_c(e_1)(\nu_o(e_2))$;
- replace every atom of the form $e_1(e_2, e_3)$ with the atom $\nu_r(e_1)(\nu_o(e_2), \nu_o(e_3))$;
- replace every atom of the form $e_1 \sqsubseteq_\mathcal{C} e_2$ with the atom $isa_C(\nu_o(e_1), \nu_o(e_2))$;
- replace every atom of the form $e_1 \sqsubseteq_\mathcal{R} e_2$ with the atom $isa_R(\nu_o(e_1), \nu_o(e_2))$.

Query answering algorithm We now present the algorithm Entails, that decides entailment of an ISA-ground CQ with respect to a *HiDL-Lite* KB $\mathcal{K}$. The algorithm reduces the above entailment to entailment of a conjunctive query in *DL-Lite*$_{\mathcal{R}}$.

Algorithm Entails$(\mathcal{K}, q)$
Input: *HiDL-Lite* KB $\mathcal{K}$, ISA-ground CQ q
Output: true if $\mathcal{K} \models q$, false otherwise
begin
  if there exists $q' \in MG(q, \mathcal{K})$ such that $DL(\mathcal{K}) \models_{\textit{DL-Lite}_{\mathcal{R}}} \tau(q')$
  then return true
  else return false
end

The following theorem states correctness of the above algorithm, which follows from the above definitions and from Proposition 2.

Theorem 1. *Given a HiDL-Lite KB $\mathcal{K}$ and an ISA-ground CQ q, $\mathcal{K} \models q$ iff* Entails$(q, \mathcal{K})$ *returns true.*

4.2 Complexity results

We now analyze complexity of query answering in *HiDL-Lite*. Our aim is to show that, for the class of queries considered in this section, *HiDL-Lite* preserves the good computational properties of *DL-Lite*$_{\mathcal{R}}$.

Combined complexity We first establish the combined complexity (i.e., the complexity with respect to the size of both the KB and the query) of answering ISA-ground CQs in *HiDL-Lite*. The following property immediately follows from Theorem 1 and from the fact that answering CQs over *DL-Lite*$_{\mathcal{R}}$ KBs is NP-complete with respect to combined complexity.

Theorem 2. *Given a HiDL-Lite KB $\mathcal{K}$ and an ISA-ground CQ q, deciding whether $\mathcal{K} \models q$ is NP-complete with respect to the size of $\mathcal{K}$ and q.*

KB complexity Then, we study complexity of answering ISA-ground CQs with respect to the size of the KB only. We start from the following property.

Lemma 1. *Given a HiDL-Lite KB $\mathcal{K}$, $DL(\mathcal{K})$ can be constructed in time polynomial in the size of $\mathcal{K}$ only.*

Moreover, observe that the number of queries in $MG(q, \mathcal{K})$ is polynomial in the size of $\mathcal{K}$, and the size of every CQ in $MG(q, \mathcal{K})$ is the same as the size of q. This fact and the above lemma imply the following result.

Theorem 3. *Given a HiDL-Lite KB $\mathcal{K}$ and an ISA-ground CQ q, deciding whether $\mathcal{K} \models q$ is PTIME with respect to the size of $\mathcal{K}$.*

Pure ABox complexity Finally, in order to compare the computational properties of *DL-Lite*$_\mathcal{R}$ with respect to ABox complexity, we make the further assumption that the *HiDL-Lite* KB is divided in two parts: such a partition isolates the "purely extensional" atoms, i.e., the atoms of the KB that are not ISA-atoms and whose arguments are "pure objects" in the KB: informally, an element e is a pure object in $\mathcal{K}$ if there exists no atom in the KB that is not an ISA-atom and in which the name of e occurs in a predicate position. From the semantic viewpoint, such assertions constitute the part of the *HiDL-Lite* KB $\mathcal{K}$ corresponding to the ABox in a *DL-Lite*$_\mathcal{R}$ KB.

Formally, we define such partitioned KBs as follows.

First, we associate every element e in $\mathcal{E}(\mathcal{N})$ with a name in $\mathcal{N}$, as follows: if e has the form A, $\exists A$, $\exists A^-$, or A^-, where $A \in \mathcal{N}$, then the name of e is A.

Moreover, we call *metanames of* $\mathcal{K}$ (denoted by *MN*$(\mathcal{K})$) the set of all names $A \in \mathcal{N}(\mathcal{K})$ such that there is an element e with name A occurring in a predicate position in $\mathcal{K}$.

Given a KB $\mathcal{K}$, we define *ObjElem*$(\mathcal{K})$ as the set of all elements e such that $e \in$ *adom*$(\mathcal{K})$ and the name of e does not belong to *MN*$(\mathcal{K})$. Then, given a KB $\mathcal{K}$, we define the sets of assertions *obj*$(\mathcal{K})$ and *meta*$(\mathcal{K})$ as follows:

$$\begin{aligned} \textit{obj}(\mathcal{K}) &= \{e_1(e_2) \mid e_1(e_2) \in \mathcal{K} \text{ and } e_2 \in \textit{ObjElem}(\mathcal{K})\} \cup \\ &\quad \{e_1(e_2, e_3) \mid e_1(e_2, e_3) \in \mathcal{K} \text{ and } e_2, e_3 \in \textit{ObjElem}(\mathcal{K})\} \\ \textit{meta}(\mathcal{K}) &= \mathcal{K} - \textit{obj}(\mathcal{K}) \end{aligned}$$

Finally, we define partitioned KBs. A *partitioned HiDL-Lite KB* is a pair $\langle \mathcal{K}_A, \mathcal{K}_M \rangle$ such that:

- $\mathcal{K}_A = \textit{obj}(\mathcal{K}_A \cup \mathcal{K}_M)$;
- $\mathcal{K}_M = \textit{meta}(\mathcal{K}_A \cup \mathcal{K}_M)$;
- every name occurring in a predicate position in $\mathcal{K}_A$ also occurs in $\mathcal{K}_M$.

Notice that the last condition of the above definition can be always satisfied by adding trivial ISA-atoms (e.g., by adding, for every name A occurring in a predicate position in $\mathcal{K}_A$, the ISA-atom $A \sqsubseteq_\mathcal{C} A$).

We now analyze complexity of answering ISA-ground CQs over partitioned *HiDL-Lite* KBs. In particular, we are able to provide the following upper bound with respect to the size of $\mathcal{K}_A$.

Theorem 4. *Given a partitioned HiDL-Lite KB $\langle \mathcal{K}_A, \mathcal{K}_M \rangle$ and an ISA-ground CQ q, deciding whether $\mathcal{K} \models q$ is LOGSPACE with respect to the size of $\mathcal{K}_A$.*

The above theorem follows from the following properties: (i) answering CQs in *DL-Lite*$_\mathcal{R}$ is LOGSPACE with respect to the size of the ABox; (ii) from the definition of partitioned KB, it follows that the reduction to query answering in *DL-Lite*$_\mathcal{R}$ shown in the previous subsection is LOGSPACE with respect to the size of $\mathcal{K}_A$ (in particular, observe that *MG*$(q, \mathcal{K})$ does not depend on $\mathcal{K}_A$).

5 Conclusions

We have presented the first results of our investigation on extending DL-Lite with higher-order capabilities. Our basic result is that higher-order constructs can be safely

added to DL-lite while keeping all the reasoning tasks tractable, including answering conjunctive queries mixing object-level and meta-level elements.

We are currently working on several extensions to the work reported here. First, while we have considered only ISA-ground conjunctive queries in this paper, our goal is to devise reasoning procedures and complexity characterization of general (non-ISA-ground) conjunctive queries s, and also union of conjunctive queries. Second, we are studying the impact of adding constructs that induce negative information such as functional restrictions and disjointness assertions, in the spirit of *DL-Lite*$_{\mathcal{F}}$ [4] and *DL-Lite*$_{\mathcal{A}}$ [12]. Finally, we are considering the addition of $\sqsubseteq_{\mathcal{C}}$ and $\sqsubseteq_{\mathcal{R}}$ to the alphabet used in the atomic formulas, similarly to RDFS [8, 6], with the goal of understanding the impact on the decidability and the complexity of query answering.

Acknowledgments. This research has been partially supported by FET project TONES (Thinking ONtologiES), funded by the EU under contract number FP6-7603, by project HYPER, funded by IBM through a Shared University Research (SUR) Award grant, and by MIUR FIRB 2005 project "Tecnologie Orientate alla Conoscenza per Aggregazioni di Imprese in Internet" (TOCAI.IT).

References

1. G. Attardi and M. Simi. Consistency and completeness of OMEGA, a logic for knowledge representation. In *Proc. of IJCAI'81*, pages 504–510, 1981.
2. L. Badea. Reifying concepts in description logics. In *Proc. of IJCAI'97*, pages 142–147, 1997.
3. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Data complexity of query answering in description logics. In *Proc. of KR 2006*, pages 260–270, 2006.
4. D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati. Tractable reasoning and efficient query answering in description logics: The DL-Lite family. *J. of Automated Reasoning*, 39(3):385–429, 2007.
5. N. Guarino and C. A. Welty. An overview of OntoClean. In S. Staab and R. Studer, editors, *Handbook on Ontologies*. Springer, 2004.
6. P. Hayes. RDF semantics. W3C Recommendation, Feb. 2004. Available at `http://www.w3.org/TR/rdf-mt/`.
7. M. Kifer, G. Lausen, and J. Wu. Logical foundations of Object-Oriented and frame-based languages. *J. of the ACM*, 42(4):741–843, 1995.
8. G. Klyne and J. J. Carroll. Resource description framework (RDF): Concepts and abstract syntax. W3C Recommendation, Feb. 2004. Available at `http://www.w3.org/TR/rdf-concepts/`.
9. F. Lehmann, editor. *Semantic Networks in Artificial Intelligence*. Pergamon Press, Oxford (United Kingdom), 1992.
10. B. Motik. On the properties of metamodeling in owl. *J. Log. Comput.*, 17(4):617–637, 2007.
11. J. Mylopoulos, P. A. Bernstein, and H. K. T. Wong. A language facility for designing database-intensive applications. *ACM Trans. on Database Systems*, 5(2):185–207, 1980.
12. A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati. Linking data to ontologies. *J. on Data Semantics*, 2008. To appear.
13. H. J. ter Horst. Completeness, decidability and complexity of entailment for rdf schema and a semantic extension involving the owl vocabulary. *J. Web Sem.*, 3(2-3):79–115, 2005.