DL-Lite with Attributes and Datatypes

Alessandro Artale and Vladislav Ryzhikov
KRDB Research Centre
Free University of Bozen-Bolzano, Italy
{*lastname*}@inf.unibz.it

Roman Kontchakov
Dept. of Comp. Science and Inf. Sys.
Birkbeck, University of London, UK
roman@dcs.bbk.ac.uk

Abstract.

We extend the *DL-Lite* languages by means of *attributes* and *datatypes*. Attributes—a notion borrowed from data models—associate concrete values from datatypes to abstract objects and in this way complement roles, which describe relationships between abstract objects. The extended languages remain tractable (with a notable exception) even though they contain both existential and (a limited form of) universal quantification. We present complexity results for two most important reasoning problems in *DL-Lite*: combined complexity of knowledge base satisfiability and data complexity of positive existential query answering.

1 Introduction

The *DL-Lite family* of description logics has recently been proposed and investigated in [7, 8] and later extended in [2, 15, 3]. The relevance of the *DL-Lite* family is witnessed by the fact that it forms the basis of OWL 2 QL, one of the three profiles of the Web Ontology Language, OWL 2 (www.w3.org/TR/owl2-profiles). According to the official W3C profiles document, the purpose of OWL 2 QL is to be the language of choice for applications that use very large amounts of data.

This paper extends the *DL-Lite* languages of [3] with so-called *attributes* ($\mathcal{A}$), which associate concrete values from datatypes with abstract objects. These extensions will be formalized in a new family of languages, *DL-Lite*$_\alpha^{(\mathcal{HLNA})}$ with $\alpha \in \{core, krom, horn, bool\}$, which contain role and attribute inclusions with both (unqualified) existential and (a limited form of) universal quantification. Original and tight complexity results for both knowledge base satisfiability and query answering will be presented in this paper.

The notion of *attributes*, borrowed from conceptual modelling formalisms, introduces a distinction between (abstract) objects and concrete values (integers, reals, strings, etc.) and, consequently, between concepts (sets of objects) and datatypes (sets of values), and between roles (relating objects to objects) and attributes (relating objects to values). The language *DL-Lite*$_\mathcal{A}$ [15] was introduced with the aim of capturing the notion of attributes in *DL-Lite* in the setting of *ontology-based data access* (OBDA). The datatypes of *DL-Lite*$_\mathcal{A}$ are modelled as pairwise *disjoint* sets of values (which are also disjoint from concepts); a similar choice is made by various DLs encoding conceptual models [9, 6, 1]. Furthermore, datatypes of *DL-Lite*$_\mathcal{A}$ are used for typing attributes *globally*: e.g., the concept inclusion $\exists \mathit{salary}^- \sqsubseteq \mathit{Real}$ can be used to constrain the range of attribute *salary* to the type *Real*. However, this means that even if associated with different concepts, attributes sharing the same name must have the same range restriction.

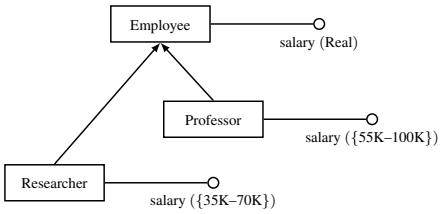

Figure 1. Salary example

We consider a more expressive language for attributes and datatypes in *DL-Lite*. We present two main extensions of the original *DL-Lite*$_\mathcal{A}$: (*i*) datatypes are not necessarily mutually disjoint; instead, Horn clauses define relations between them (including disjointness and subtype relations); (*ii*) range restrictions for attributes are *local* (rather than global): i.e., concept inclusions of the form $C \sqsubseteq \forall U.T$ specify that all values of the attribute U for instances of the concept C belong to datatype T. In this way, we capture a wider range of datatypes (e.g., intervals over the reals) and allow reuse of the very same attribute associated to different concepts, but with different range restrictions. As an example, consider the Entity-Relationship diagram in Fig. 1, which says, in particular, that

- employees' salary is of type *Real*, i.e., $\mathit{Employee} \sqsubseteq \forall \mathit{salary}.\mathit{Real}$;
- researchers' salary is in the range *35K–70K*, which is an interval type, a subset of *Real*, i.e., $\mathit{Researcher} \sqsubseteq \forall \mathit{salary}.\{\mathit{35K–70K}\}$;
- and professors' salary in the range *55K–100K*, i.e., $\mathit{Professor} \sqsubseteq \forall \mathit{salary}.\{\mathit{55K–100K}\}$;
- with researchers and professors being employees, i.e., $\mathit{Researcher} \sqsubseteq \mathit{Employee}$ and $\mathit{Professor} \sqsubseteq \mathit{Employee}$.

Local attributes are strictly more expressive than global attributes: for example, the concept inclusion $\top \sqsubseteq \forall \mathit{salary}.\mathit{Real}$ is equivalent to $\exists \mathit{salary}^- \sqsubseteq \mathit{Real}$ mentioned above and implies that *every* value of *salary* is a *Real*, independently from the type of the employee. Using local attributes we can infer concept disjointness from datatype disjointness for the *same* (existentially qualified) attribute. For example, assume that in the scenario of Fig. 1 we add the concept of *ForeignEmployee* as having at-least one *salary* that must be a *String* (to take account of the currency). Then *Employee* and *ForeignEmployee* become disjoint concepts—i.e., $\mathit{Employee} \sqcap \mathit{ForeignEmployee} \sqsubseteq \bot$ will be implied—because of disjointness of the respective datatypes and restrictions on the salary attribute. We also allow more general

datatype inclusions, which, for instance, can express that the intersection of a number of datatypes is empty.

Our work lies between the *DL-Lite*$_{\mathcal{A}}$ proposal and the extensions of DLs with concrete domains (see [13] for an overview). According to the concrete domain terminology, we consider a path-free extension with unary predicates—predicates coincide with datatypes with a fixed interpretation, as in *DL-Lite*$_{\mathcal{A}}$. Differently from the concrete domain approach, we do not require attributes to be functional; instead, we can specify generic number restrictions over them, similarly to extensions of $\mathcal{EL}$ with datatypes [5, 11] and the notion of datatype properties in OWL 2 [14, 10]. Our approach works as far as datatypes are *safe*, i.e., unbounded—query answering is coNP-hard in presence of datatypes of specific cardinalities [12, 16]—and no covering constraints hold between them—query answering becomes coNP-hard again in the presence of a datatype, whose extension is a subset of (is covered by) the union of two other datatypes (cf. Theorem 2).

We provide tight complexity results showing that for the Bool, Horn and core languages addition of *local* and *safe* range restrictions on attributes does not change the complexity of knowledge base satisfiability. On the other hand, surprisingly, for the Krom language complexity increases from NLogSpace to NP. These results reflect the intuition that universal restrictions on attributes—as studied in this paper—cannot introduce cyclic dependencies between concepts; on the other hand, unrestricted use of universal restrictions ($\forall R.C$) together with sub-roles, by which qualified existential restrictions ($\exists R.C$) can be encoded, results in ExpTime-completeness [8].

We complete our complexity results by showing that *positive existential query* answering (and so, conjunctive query answering) over core and Horn knowledge bases with attributes, local range restrictions and safe datatypes is still FO-rewritable and so, is in AC^0 for data complexity.

The paper is organized as follows. Section 2 presents *DL-Lite* and its fragments. Section 3 discusses the notion of *safe* datatypes used in this paper. Sections 4 and 5 study combined complexity of KB satisfiability and data complexity of answering positive existential queries, respectively, when attributes and datatypes are present. Section 6 concludes this paper. Complete proofs of all the results can be found in the full version [4].

2 The Description Logic *DL-Lite*$_{bool}^{(\mathcal{HNA})}$

The language of *DL-Lite*$_{bool}^{(\mathcal{HNA})}$ contains *object names* $a_0, a_1, \ldots$, *value names* $v_0, v_1, \ldots$, *concept names* $A_0, A_1, \ldots$, *role names* $P_0, P_1, \ldots$, *attribute names* $U_0, U_1, \ldots$, and datatype names $T_0, T_1, \ldots$. Complex *roles* R, *datatypes* T and *concepts* C are defined as follows:

$$R ::= P_i \mid P_i^-, \qquad B ::= \top \mid \bot \mid A_i \mid \geq q\, R \mid \geq q\, U_i$$
$$T ::= \bot_{\mathcal{D}} \mid T_i, \qquad C ::= B \mid \neg C \mid C_1 \sqcap C_2,$$

where q is a positive integer. Concepts of the form B are called *basic concepts*. A *DL-Lite*$_{bool}^{(\mathcal{HNA})}$ *TBox*, $\mathcal{T}$, is a finite set of concept, role and attribute *inclusions* of the form:

$$C_1 \sqsubseteq C_2 \text{ and } C \sqsubseteq \forall U.T, \quad R_1 \sqsubseteq R_2, \quad U_1 \sqsubseteq U_2,$$

and an *ABox*, $\mathcal{A}$, is a finite set of assertions of the form:

$$A_k(a_i), \ \neg A_k(a_i), \ P_k(a_i, a_j), \ \neg P_k(a_i, a_j), \ U_k(a_i, v_j).$$

We standardly abbreviate $\geq 1\, R$ and $\geq 1\, U$ by $\exists R$ and $\exists U$, respectively. Taken together, a TBox $\mathcal{T}$ and an ABox $\mathcal{A}$ constitute the *knowledge base* (KB) $\mathcal{K} = (\mathcal{T}, \mathcal{A})$.

It is known [3] that reasoning with role inclusions and number restrictions (even in core TBoxes without attributes) is already rather costly, ExpTime-complete. Thus we impose the following syntactic restriction on sub-roles and sub-attributes [3]:

(inter$_R$) if R has a proper sub-role in $\mathcal{T}$ then it contains no negative occurrences[1] of $\geq q\, R$ or $\geq q\, R^-$ for $q \geq 2$;

(inter$_U$) if U has a proper sub-attribute in $\mathcal{T}$ then it contains no negative occurrences of $\geq q\, U$ for $q \geq 2$.

Semantics. As usual in description logic, an *interpretation*, $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, consists of a nonempty *domain* $\Delta^{\mathcal{I}}$ and an interpretation function $\cdot^{\mathcal{I}}$. The interpretation domain $\Delta^{\mathcal{I}}$ is the union of two nonempty disjoint sets: the *domain of objects* $\Delta_O^{\mathcal{I}}$ and the *domain of values* $\Delta_V^{\mathcal{I}}$. We assume that all interpretations agree on the semantics of datatypes and values: $\bot_{\mathcal{D}}^{\mathcal{I}} = \emptyset$ and $T_i^{\mathcal{I}} = \mathit{val}(T_i) \subseteq \Delta_V^{\mathcal{I}}$ is the set of values of each datatype T_i (which does not depend on a particular interpretation) and $v_j^{\mathcal{I}} = \mathit{val}(v_j) \in \Delta_V^{\mathcal{I}}$ is the value of each name v_j (which, again, does not depend on $\mathcal{I}$). Note that the datatypes do not have to be mutually disjoint—instead, we assume that datatype constraints can be captured by Horn clauses—we will clarify the assumptions in Section 3.

The interpretation function $\cdot^{\mathcal{I}}$ assigns an element $a_i^{\mathcal{I}} \in \Delta_O^{\mathcal{I}}$ to each object name a_i, a subset $A_k^{\mathcal{I}} \subseteq \Delta_O^{\mathcal{I}}$ of the domain of objects to each concept name A_k, a binary relation $P_k^{\mathcal{I}} \subseteq \Delta_O^{\mathcal{I}} \times \Delta_O^{\mathcal{I}}$ over the domain of objects to each role name P_k, and a binary relation $U_k^{\mathcal{I}} \subseteq \Delta_O^{\mathcal{I}} \times \Delta_V^{\mathcal{I}}$ to each attribute name U_k. We adopt the *unique name assumption* (UNA): $a_i^{\mathcal{I}} \neq a_j^{\mathcal{I}}$, for all $i \neq j$. It is known [3] that *not* adopting the UNA in *DL-Lite* languages with number restrictions leads to a significant increase in the complexity of reasoning: KB satisfiability goes from NLogSpace to PTime-hard with functionality constraints and even to NP-hard with arbitrary number restrictions; query answering loses the AC^0 data complexity. Complex roles and concept are interpreted in $\mathcal{I}$ in the standard way:

$$\begin{aligned}
(P_k^-)^{\mathcal{I}} &= \{(w', w) \in \Delta_O^{\mathcal{I}} \times \Delta_O^{\mathcal{I}} \mid (w, w') \in P_k^{\mathcal{I}}\},\\
\top^{\mathcal{I}} &= \Delta_O^{\mathcal{I}}, \qquad \bot^{\mathcal{I}} = \emptyset,\\
(C_1 \sqcap C_2)^{\mathcal{I}} &= C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}, \qquad (\neg C)^{\mathcal{I}} = \Delta_O^{\mathcal{I}} \setminus C^{\mathcal{I}},\\
(\geq q\, R)^{\mathcal{I}} &= \big\{w \in \Delta_O^{\mathcal{I}} \mid \sharp\{w' \mid (w, w') \in R^{\mathcal{I}}\} \geq q\big\},\\
(\geq q\, U)^{\mathcal{I}} &= \big\{w \in \Delta_O^{\mathcal{I}} \mid \sharp\{v \mid (w, v) \in U^{\mathcal{I}}\} \geq q\big\},\\
(\forall U.T)^{\mathcal{I}} &= \big\{w \in \Delta_O^{\mathcal{I}} \mid \forall v.\, (w, v) \in U^{\mathcal{I}} \to v \in T^{\mathcal{I}}\big\},
\end{aligned}$$

where $\sharp X$ is the cardinality of X. The *satisfaction relation* $\models$ is also standard:

$$\begin{aligned}
&\mathcal{I} \models C_1 \sqsubseteq C_2 \text{ iff } C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}, \quad \mathcal{I} \models R_1 \sqsubseteq R_2 \text{ iff } R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}},\\
&\mathcal{I} \models U_1 \sqsubseteq U_2 \text{ iff } U_1^{\mathcal{I}} \subseteq U_2^{\mathcal{I}}, \quad \mathcal{I} \models U_k(a_i, v_j) \text{ iff } (a_i^{\mathcal{I}}, v_j^{\mathcal{I}}) \in U_k^{\mathcal{I}},\\
&\mathcal{I} \models A_k(a_i) \text{ iff } a_i^{\mathcal{I}} \in A_k^{\mathcal{I}}, \quad \mathcal{I} \models P_k(a_i, a_j) \text{ iff } (a_i^{\mathcal{I}}, a_j^{\mathcal{I}}) \in P_k^{\mathcal{I}},\\
&\mathcal{I} \models \neg A_k(a_i) \text{ iff } a_i^{\mathcal{I}} \notin A_k^{\mathcal{I}}, \quad \mathcal{I} \models \neg P_k(a_i, a_j) \text{ iff } (a_i^{\mathcal{I}}, a_j^{\mathcal{I}}) \notin P_k^{\mathcal{I}}.
\end{aligned}$$

A KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ is said to be *satisfiable* (or *consistent*) if there is an interpretation, $\mathcal{I}$, satisfying all the members of $\mathcal{T}$ and $\mathcal{A}$. In this case we write $\mathcal{I} \models \mathcal{K}$ (as well as $\mathcal{I} \models \mathcal{T}$ and $\mathcal{I} \models \mathcal{A}$) and say that $\mathcal{I}$ is a *model of* $\mathcal{K}$ ($\mathcal{T}$ and $\mathcal{A}$).

A *positive existential query* $\mathsf{q}(x_1, \ldots, x_n)$ is a first-order formula $\varphi(x_1, \ldots, x_n)$ constructed by means of conjunction, disjunction and existential quantification starting from atoms of the from

[1] An occurrence of a concept on the right-hand (left-hand) side of a concept inclusion is called *negative* if it is in the scope of an odd (even) number of negations $\neg$; otherwise it is called *positive*.

$A_k(t_1)$, $T_k(t_1)$, $P_k(t_1, t_2)$ and $U_k(t_1, t_2)$, where A_k is a concept name, T_k a datatype name, P_k a role name, U_k an attribute name, and t_1, t_2 are *terms* taken from the list of variables $y_0, y_1, \ldots$, object names $a_0, a_1, \ldots$ and value names $v_0, v_1, \ldots$; object names and value names will be called *constants*. We write $\mathsf{q}(\vec{x})$ for a query with free variables $\vec{x} = x_1, \ldots, x_n$ and $\mathsf{q}(\vec{a})$ for the result of replacing every occurrence of x_i in $\varphi(\vec{x})$ with the ith component a_i of a vector of constants $\vec{a} = a_1, \ldots, a_n$. We will equivocate between DL and first-order interpretations and write $\mathcal{I} \models \mathsf{q}(\vec{a})$ to say that $\mathsf{q}(\vec{a})$ is true in $\mathcal{I}$. A *conjunctive query* is a positive existential query without disjunctions.

For a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, we say that a tuple $\vec{a}$ of constants from $\mathcal{A}$ is a *certain answer* to $\mathsf{q}(\vec{x})$ with respect to $\mathcal{K}$, and write $\mathcal{K} \models \mathsf{q}(\vec{a})$, if $\mathcal{I} \models \mathsf{q}(\vec{a})$ whenever $\mathcal{I} \models \mathcal{K}$. The *query answering problem* is: given a KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, a query $\mathsf{q}(\vec{x})$ and a tuple $\vec{a}$ of constants from $\mathcal{A}$, decide whether $\mathcal{K} \models \mathsf{q}(\vec{a})$.

Fragments of *DL-Lite*$_{bool}^{(\mathcal{HN}\mathcal{A})}$. We consider syntactic restrictions on the form of concept inclusions in *DL-Lite*$_{bool}^{(\mathcal{HN}\mathcal{A})}$ TBoxes. Following the naming scheme of the extended *DL-Lite* family [3], we adopt the following definitions. A KB $\mathcal{K}$ belongs to *DL-Lite*$_{krom}^{(\mathcal{HN}\mathcal{A})}$ if only negation is used in construction of its complex concepts:

$$C \ ::= \ B \mid \neg B \qquad \text{(Krom)}$$

(here and below the B are basic concepts). $\mathcal{K}$ is in *DL-Lite*$_{horn}^{(\mathcal{HN}\mathcal{A})}$ if its complex concepts are constructed by using only intersection:

$$C \ ::= \ B_1 \sqcap \cdots \sqcap B_k. \qquad \text{(Horn)}$$

Finally, we say $\mathcal{K}$ is in *DL-Lite*$_{core}^{(\mathcal{HN}\mathcal{A})}$ if its concept inclusions are of the form:

$$B_1 \sqsubseteq B_2, \quad B_1 \sqcap B_2 \sqsubseteq \bot. \qquad \text{(core)}$$

Note that the positive occurrences of B on the right-hand side of the above inclusions can also be of the form $\forall U.T$. As $B_1 \sqsubseteq \neg B_2$ is equivalent to $B_1 \sqcap B_2 \sqsubseteq \bot$, core TBoxes can be regarded as sitting in the intersection of Krom and Horn TBoxes.

The following table summarizes the obtained combined complexity results for KB satisfiability and data complexity results for query answering (with numbers coded in binary):

language	KB satisfiability	query answering
DL-Lite$_{core}^{(\mathcal{HN}\mathcal{A})}$	NLogSpace [Th.4]	AC^0 [Th.6]
DL-Lite$_{horn}^{(\mathcal{HN}\mathcal{A})}$	PTIME [Th.4]	AC^0 [Th.6]
DL-Lite$_{krom}^{(\mathcal{HN}\mathcal{A})}$	NP [Th.5]	CONP [3]
DL-Lite$_{bool}^{(\mathcal{HN}\mathcal{A})}$	NP [Th.4]	CONP [3]

3 Safe Datatypes

In this section we define the notion of *safe* datatypes and show that such restrictions are required for preserving data complexity of query answering.

DEFINITION 1. *A set of datatypes $\mathcal{D} = \{T_1, \ldots, T_n\}$ is called* safe *if* (i) *the difference between an arbitrary intersection of datatypes and an arbitrary union of datatypes is either empty or unbounded*; (ii) *all constraints between datatypes are in the form of Horn clauses $T_{i_1} \cap \cdots \cap T_{i_k} \subseteq_{\mathcal{D}} T_{i_0}$.*

A set of datatypes $\mathcal{D}$ is called weakly safe *if* (i′) *arbitrary intersections of datatypes are either empty or unbounded and* (ii) *holds.*

Restriction (*i*) has been independently introduced by Savkovic [16].

It follows, in particular, that if $\mathcal{D}$ is (weakly) safe we can assume that each non-empty datatype T_i is unbounded (note that query answering becomes CONP-hard in presence of datatypes of specific cardinalities [12]); and if $\mathcal{D}$ is safe then also arbitrary intersections of datatypes are either empty or unbounded. Thus, if $\mathcal{D}$ is safe then it is also weakly safe. Condition (*ii*) ensures that datatype constraints in $\mathcal{D}$ have the form of Horn clauses, $T_1 \cap \cdots \cap T_k \subseteq_{\mathcal{D}} T$, and thus computable in PTIME; we further restrict datatype constraints to $T_1 \subseteq_{\mathcal{D}} T_2$ and $T_1 \cap \cdots \cap T_k \subseteq_{\mathcal{D}} \bot_{\mathcal{D}}$ when dealing with the *core* language. Indeed, allowing covering constraints between datatypes leads to CONP-hardness of conjunctive query answering:

THEOREM 2. *Conjunctive query answering in DL-Lite*$_{core}^{(\mathcal{HN}\mathcal{A})}$ *with covering constraints on datatypes is* CONP*-hard* (*even without sub-roles, sub-attributes and number restrictions*).

Proof. We prove the result by reduction of the complement of 2+2CNF (similar to instance checking in $\mathcal{ALE}$ [17]). Suppose we are given a CNF ψ in which every clause contains two positive and two negative literals (including the constants *true*, *false*). Let T be a datatype covered by non-empty disjoint T_0 and T_1. Let $\mathcal{T}$ contain the following concept inclusions for an attribute U and concepts B and C: $B \sqsubseteq \exists U$, $B \sqsubseteq \forall U.T$, $C \sqsubseteq \forall U.T_0$, and consider the following conjunctive query

$$\begin{aligned}\mathsf{q} = \exists y, \vec{t}, \vec{u}\, \big(&P_1(y, t_1) \land P_2(y, t_2) \land N_1(y, t_3) \land N_2(y, t_4) \\ &\land U(t_1, u_1) \land U(t_2, u_2) \land U(t_3, u_3) \land U(t_4, u_4) \\ &\land T_0(u_1) \land T_0(u_2) \land T_1(u_3) \land T_1(u_4)\big)\end{aligned}$$

with roles P_1, P_2, N_1 and N_2. We construct an ABox $\mathcal{A}_\psi$ with individuals *true* and *false* for the propositional constants, an individual x_i, for each propositional variable x_i in ψ, and an individual c_i, for each clause of ψ. Let $\mathcal{A}_\psi$ contain assertion $B(x_i)$, for each propositional variable x_i in ψ, assertions $C(\mathit{false}), U(\mathit{true}, v_1)$, for a value v_1 of datatype T_1, and the following assertions, for each clause $x_{j_{i1}} \vee x_{j_{i2}} \vee \neg x_{j_{i3}} \vee \neg x_{j_{i4}}$ of ψ:

$$P_1(c_i, x_{j_{i1}}), P_2(c_i, x_{j_{i2}}), N_1(c_i, x_{j_{i3}}), N_2(c_i, x_{j_{i4}})$$

(here the x_j may include propositional constants). It is readily checked that $(\mathcal{T}, \mathcal{A}_\psi) \not\models \mathsf{q}$ iff ψ is satisfiable. Indeed, if ψ is satisfiable we construct $\mathcal{I}$ by 'extending' $\mathcal{A}_\psi$ by $U(x_i, v_0)$ if x_i is false in the satisfying assignment and by $U(x_i, v_1)$ otherwise, where v_0 is in T_0 and v_1 in T_1 (recall that these datatypes are non-empty and disjoint). Conversely, if $(\mathcal{T}, \mathcal{A}_\psi) \not\models \mathsf{q}$ then there is a model $\mathcal{I}$ of $(\mathcal{T}, \mathcal{A}_\psi)$ in which q is false. Then the satisfying assignment can be defined as follows: a propositional variable x_i is true if one of the attribute U values of x_i belongs to datatype T_1—it does not matter whether other values belong to T_0 or not, the negative answer to the query q guarantees that ψ is true under such an assignment. □

The following theorem shows that without condition (*i*) we lose FO-rewritability of conjunctive queries in the presence of number restrictions.

THEOREM 3. *Conjunctive query answering in DL-Lite*$_{core}^{(\mathcal{HN}\mathcal{A})}$ *with datatypes not respecting condition* (i) *of Definition 1 is* CONP*-hard* (*even without sub-roles and sub-attributes*).

Proof. We modify the proof of Theorem 2. Assume that the difference between a datatype, T, and a union of two datatypes, T_0 and T_1, has a finite cardinality, say k. We replace the concept inclusion $B \sqsubseteq \exists U$ with $B \sqsubseteq \geq (k+1)\, U$, which forces a choice of at least

one U attribute value to be in either T_0 or T_1. In the former case, as before, we assume that the propositional variable gets value *false*, while in the latter case it gets value *true*. □

Thus, the safe condition essentially disallows the use of *enumerations* and any datatypes, whose non-empty intersection or difference has a finite number of elements. From now on we consider only (weakly) safe datatypes.

4 Complexity of KB Satisfiability in *DL-Lite*$_\alpha^{(\mathcal{HNA})}$

We first introduce the encoding of a *DL-Lite*$_{bool}^{(\mathcal{HNA})}$ KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ into a first-order sentence $\mathcal{K}^{\ddagger\mathbf{a}}$ with one variable, adopting the technique introduced in [3]. We denote by $role^{\pm}(\mathcal{K})$ the set of role names in $\mathcal{K}$ and their inverses, by $att(\mathcal{K})$ and $dt(\mathcal{K})$ the sets of attribute and datatype names in $\mathcal{K}$, respectively, and by $ob(\mathcal{A})$ and $val(\mathcal{A})$ the set of all object and value names in $\mathcal{A}$, respectively. To simplify the presentation, we will assume that $(R^-)^-$ is the same as R and will often use H for a role R or an attribute name U. We will also assume that all number restrictions are of the form $\exists R$ and $\exists T$ (i.e., only $q = 1$ is allowed) and that the ABox contains no negative assertions of the form $\neg A_k(a_i)$ and $\neg P_k(a_i, a_j)$—see the full version [4] for the treatment of the full language.

Every $a_i \in ob(\mathcal{A})$ is associated with the individual constant a_i, and every concept name A_i with the unary predicate $A_i(x)$. For each concept $\exists R$, we take a fresh unary predicate $ER(x)$. Intuitively, for a role name P_k, the predicate EP_k represents the objects with a P_k-successor—the domain of P_k—and EP_k^- the range of P_k. We also introduce individual constants, as representatives of the objects in the domain (dp_k) and the range (dp_k^-) of each role P_k. Similarly, for each attribute name U_i, we take a unary predicate $EU_i(x)$, representing the objects with at least one value of the attribute U. We also need, for each attribute name U_i and each datatype name T_j, a unary predicate $U_iT_j(x)$, representing the objects such that *all* their U_i attribute values belong to the datatype T_j (as usual, if they have attribute U_i values at all).

The encoding C^* of a concept C is then defined inductively:

$$\begin{array}{rlrl} \bot^* &= \bot, & (A_i)^* &= A_i(x), \\ \top^* &= \top, & (\neg C)^* &= \neg C^*(x), \\ (\exists H)^* &= EH(x), & (C_1 \sqcap C_2)^* &= C_1^*(x) \wedge C_2^*(x), \\ (\forall U.\bot_{\mathcal{D}})^* &= \neg E_1U(x), & (\forall U.T_i)^* &= UT_i(x). \end{array}$$

The following sentence then encodes the knowledge base $\mathcal{K}$:

$$\mathcal{K}^{\ddagger\mathbf{a}} = \forall x \Big[\mathcal{T}^*(x) \wedge \beta(x) \wedge \bigwedge_{R \in role^{\pm}(\mathcal{K})} \epsilon_R(x) \wedge \bigwedge_{U \in att(\mathcal{K})} \theta_U(x) \Big] \wedge \mathcal{A}^{\ddagger\mathbf{a}},$$

where

$$\mathcal{T}^*(x) = \bigwedge_{C_1 \sqsubseteq C_2 \in \mathcal{T}} \big(C_1^*(x) \rightarrow C_2^*(x)\big) \wedge \bigwedge_{H \sqsubseteq^*_{\mathcal{T}} H'} \big((\exists H)^*(x) \rightarrow (\exists H')^*(x)\big),$$

$$\mathcal{A}^{\ddagger\mathbf{a}} = \bigwedge_{A(a_i) \in \mathcal{A}} A^*(a_i) \wedge \bigwedge_{P(a_i,a_j) \in \mathcal{A}} \big((\exists P)^*(a_i) \wedge (\exists P^-)^*(a_j)\big) \wedge \bigwedge_{U(a_i,v_j) \in \mathcal{A}} \big((\exists U)^*(a_i) \wedge \bigwedge_{\substack{T \in dt(\mathcal{K}) \\ val(v_j) \notin val(T)}} \neg(\forall U.T)^*(a_i)\big),$$

and $\sqsubseteq^*_{\mathcal{T}}$ is the reflexive and transitive closure of the sub-role and sub-attribute relations of the TBox, i.e., of the union $\{(R, R'), (R^-, R'^-) \mid R \sqsubseteq R' \in \mathcal{T}\} \cup \{(U, U') \mid U \sqsubseteq U' \in \mathcal{T}\}$.

Roles are interpreted as binary predicates in a DL interpretation and so, the range of a role R is not empty whenever its domain contains an element. So, in order to capture this intuition, in $\mathcal{K}^{\ddagger\mathbf{a}}$ we include the following formula, for each $R \in role^{\pm}(\mathcal{K})$:

$$\epsilon_R(x) = ER(x) \rightarrow ER^-(dr^-).$$

Attributes are involved both in existential and universal quantification. So, the second conjunct of $\mathcal{T}^*$ reflects the fact that if an object has a U value (existential quantifier $\exists U$) then it also has a U' value, for each U' with $U \sqsubseteq^*_{\mathcal{T}} U'$; universal quantification propagates the datatypes in the opposite direction:

$$\beta(x) = \bigwedge_{U' \sqsubseteq U \in \mathcal{T}} \bigwedge_{T \in dt(\mathcal{K})} \big((\forall U.T)^* \rightarrow (\forall U'.T)^*\big).$$

We also need a formula that captures the relationships between datatypes, as defined by the Horn clauses in $\mathcal{D}$, for all attributes U:

$$\theta_U(x) = \bigwedge_{T_1 \cap \cdots \cap T_k \subseteq_{\mathcal{D}} T} \big((\forall U.T_1)^* \wedge \cdots \wedge (\forall U.T_k)^* \rightarrow (\forall U.T)^*\big).$$

We note that the formula $\theta_U(x)$, in particular for disjoint datatypes, e.g., with $T_1 \cap T_2 \subseteq_{\mathcal{D}} \bot_{\mathcal{D}}$, demonstrates a subtle interaction between attribute range constraints, $\forall U.T$, and minimal cardinality constraints, $\exists U$.

We now show that for the Bool, Horn and core languages the addition of attributes to *DL-Lite*$_\alpha^{(\mathcal{HN})}$ of [3] does not change the combined complexity of KB satisfiability:

THEOREM 4. *Checking KB satisfiability with weakly safe datatypes is* NP*-complete in DL-Lite*$_{bool}^{(\mathcal{HNA})}$*,* PTIME*-complete in DL-Lite*$_{horn}^{(\mathcal{HNA})}$ *and* NLOGSPACE*-complete in DL-Lite*$_{core}^{(\mathcal{HNA})}$*.*

Proof. (Sketch) We show that a given KB $\mathcal{K}$ is satisfiable iff the universal first-order sentence $\mathcal{K}^{\ddagger\mathbf{a}}$ is satisfiable. One direction is straightforward: if there is a model of $\mathcal{K}$ then the model of $\mathcal{K}^{\ddagger\mathbf{a}}$ can be defined on the same domain by taking, say, C^* to be $C^{\mathcal{I}}$. The key ingredient of the converse direction is the unravelling construction: every model of $\mathcal{K}^{\ddagger\mathbf{a}}$ can be unravelled into a DL interpretation—in essence, the points dp_k and dp_k^- are copied to recover the structure of roles as binary relations, while recovering attributes requires more subtlety; see [4] for more details. □

It is of interest to note that the complexity of KB satisfiability increases in the case of Krom TBoxes:

THEOREM 5. *Satisfiability of DL-Lite*$_{krom}^{(\mathcal{HNA})}$ *KBs is* NP*-hard with a single pair of disjoint datatypes even without role and attribute inclusions nor cardinalities* (*and so, for DL-Lite*$_{krom}^{\mathcal{A}}$*).*

Proof. The proof is by reduction of 3SAT. It exploits the structure of the formula $\theta_U(x)$ in $\mathcal{K}^{\ddagger\mathbf{a}}$: if datatypes T and T' are disjoint then the concept inclusion

$$\forall U.T \sqcap \forall U.T' \sqcap \exists U \sqsubseteq \bot,$$

although not in the syntax of *DL-Lite*$_{krom}^{(\mathcal{HNA})}$, is a logical consequence of $\mathcal{T}$. Using such ternary intersections with the full negation of the Krom fragment one can encode 3SAT. Let $\varphi = \bigwedge_{i=1}^m C_i$ be a 3CNF, where the C_i are ternary clauses over variables $p_1, \ldots, p_n$. Now, suppose $p_{i_1} \vee \neg p_{i_2} \vee p_{i_3}$ is the ith clause of φ. It is equivalent to $\neg p_{i_1} \wedge p_{i_2} \wedge \neg p_{i_3} \rightarrow \bot$ and so, can be encoded as follows:

$$\neg A_{i_1} \sqsubseteq \forall U_i.T, \qquad A_{i_2} \sqsubseteq \forall U_i.T', \qquad \neg A_{i_3} \sqsubseteq \exists U_i,$$

where $A_1, \dots, A_n$ are concept names for variables $p_1, \dots, p_n$, and U_i is an attribute for the ith clause (note that Krom concept inclusions of the form $\neg B \sqsubseteq B'$ are required, which is not available in core TBoxes). Let $\mathcal{T}$ consist of all such inclusions for clauses in φ. It can be seen that φ is satisfiable iff $\mathcal{T}$ is satisfiable. □

5 Query Answering: Data Complexity

In this section we study the data complexity of answering positive existential queries over a KB expressed in languages with attributes and datatypes. As follows from the proof of Theorem 4, for a *DL-Lite*$_{bool}^{(\mathcal{HNA})}$ KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$, every model $\mathfrak{M}$ of the first-order sentence $\mathcal{K}^{\ddagger_{\mathbf{a}}}$ induces a forest-shaped model $\mathcal{I}_{\mathfrak{M}}$ of $\mathcal{K}$ with the following properties:

(forest) The names $a \in \mathit{ob}(\mathcal{A}) \cup \mathit{val}(\mathcal{A})$ induce a partitioning of the domain $\Delta^{\mathcal{I}_{\mathfrak{M}}}$ into disjoint labelled trees $\mathfrak{T}_a = (T_a, E_a, \ell_a)$ with nodes T_a, edges E_a, root a, and a labelling function ℓ_a that assigns a role or an attribute name to each edge (indicating a minimal, w.r.t. $\sqsubseteq^*_{\mathcal{T}}$, role or attribute name that required a fresh successor due to an existential quantifier); the trees for $v \in \mathit{val}(\mathcal{A})$ consist of a single node, v.

(copy) There is a map $\mathit{cp}\colon \Delta^{\mathcal{I}_{\mathfrak{M}}} \to \mathit{ob}(\mathcal{A}) \cup \mathit{val}(\mathcal{A}) \cup \{dr \mid R \in \mathit{role}^{\pm}(\mathcal{K})\}$, such that $\mathit{cp}(a) = a$, if $a \in \mathit{ob}(\mathcal{A}) \cup \mathit{val}(\mathcal{A})$, and $\mathit{cp}(w) = dr$, if $\ell_a(w', w) = R^-$, for $(w', w) \in E_a$.

(role) For every role (attribute name) H,

$$H^{\mathcal{I}_{\mathfrak{M}}} = \big\{(a_i, a_j) \mid H'(a_i, a_j) \in \mathcal{A},\ H' \sqsubseteq^*_{\mathcal{T}} H\big\} \cup \big\{(w, w') \in E_a \mid \ell_a(w, w') = H', H' \sqsubseteq^*_{\mathcal{T}} H, a \in \mathit{ob}(\mathcal{A})\big\}.$$

THEOREM 6. *The positive existential query answering problem for DL-Lite*$_{horn}^{(\mathcal{HNA})}$ *and DL-Lite*$_{core}^{(\mathcal{HNA})}$ *is in* AC^0 *for data complexity.*

Proof. We adopt the technique of the proof of Theorem 7.1 [3]. Suppose that we are given a *consistent DL-Lite*$_{horn}^{(\mathcal{HNA})}$ KB $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ and a positive existential query in prenex form $\mathsf{q}(\vec{x}) = \exists \vec{y}\, \varphi(\vec{x}, \vec{y})$ in the signature of $\mathcal{K}$. Let $\mathfrak{M}_0$ be the *minimal Herbrand model* of (the universal Horn sentence) $\mathcal{K}^{\ddagger_{\mathbf{a}}}$, and let $\mathcal{I}_0 = (\Delta^{\mathcal{I}_0}, \cdot^{\mathcal{I}_0})$ be the *canonical* model of $\mathcal{K}$, i.e., the model induced by $\mathfrak{M}_0$ (see its construction in the full version [4]). The following properties hold, for all basic concepts B and datatypes T:

$$a_i^{\mathcal{I}_0} \in B^{\mathcal{I}_0} \text{ iff } \mathcal{K} \models B(a_i), \text{ for } a_i \in \mathit{ob}(\mathcal{A}), \tag{1}$$

$$w \in B^{\mathcal{I}_0} \text{ iff } \mathcal{K} \models \exists R \sqsubseteq B, \text{ for } w \text{ with } \mathit{cp}(w) = dr, \tag{2}$$

$$v_i^{\mathcal{I}_0} \in T^{\mathcal{I}_0} \text{ iff } \mathit{val}(v_i) \in \mathit{val}(T), \text{ for } v_i \in \mathit{val}(\mathcal{A}), \tag{3}$$

$$v \in T^{\mathcal{I}_0} \text{ iff } w \in B_1^{\mathcal{I}_0}, \dots, B_k^{\mathcal{I}_0}, \mathcal{T} \models B_1 \sqcap \dots \sqcap B_k \sqsubseteq \forall U.T, \text{ for } (w, v) \in U^{\mathcal{I}_0} \text{ and } v \notin \mathit{val}(\mathcal{A}). \tag{4}$$

Formula (1) describes conditions when a named object, a_i, belongs to a basic concept, B, in the canonical model $\mathcal{I}_0$—we say it describes the *type* of a_i. Similarly, (2) describes types of unnamed objects, which are copies of the dr, for roles R; it is worth pointing out that those types are determined by a single concept, $\exists R$. The same two properties were used in the proof of he proof of Theorem 7.1 [3]. The other two properties are specific to datatypes: (3) describes the type of a named datatype value and (4) the type of an unnamed datatype value. We note that (4) holds only for safe datatypes, and even weakly safe datatypes cannot guarantee that in the process of unravelling it is always possible, for every $w \in (\exists U)^{\mathcal{I}_0}$, to pick a fresh attribute U value of the 'minimal type', i.e., a datatype value that belongs *only* to datatypes T with $w \in (\forall U.T)^{\mathcal{I}_0}$.

It is straightforward to check that the canonical model $\mathcal{I}_0$ provides correct answers to all queries:

LEMMA 7. *$\mathcal{K} \models \mathsf{q}(\vec{a})$ iff $\mathcal{I}_0 \models \mathsf{q}(\vec{a})$, for all tuples $\vec{a}$.*

The *depth* of a point $w \in \Delta^{\mathcal{I}_0}$ is the length of the shortest path in the respective tree to its root. Denote by W_m the set of points of depth $\leq m$ (including also values $v \in \Delta_V^{\mathcal{I}_0}$) that were taken to satisfy existential quantifiers for objects in W_{m-1}. Our next lemma shows that to check whether $\mathcal{I}_0 \models \mathsf{q}(\vec{a})$ it suffices to consider points of depth $\leq m_0$ in $\Delta^{\mathcal{I}_0}$, for some m_0 that does not depend on $|\mathcal{A}|$:

LEMMA 8. *If $\mathcal{I}_0 \models \exists \vec{y}\, \varphi(\vec{a}, \vec{y})$ then there is an assignment $\mathfrak{a}_0$ in W_{m_0} such that $\mathcal{I}_0 \models^{\mathfrak{a}_0} \varphi(\vec{a}, \vec{y})$ and $\mathfrak{a}_0(y_i) \in W_{m_0}$, for all $y_i \in \vec{y}$, where $m_0 = |\vec{y}| + |\mathit{role}^{\pm}(\mathcal{T})| + 1$.*

To complete the proof of Theorem 6, we encode the problem '$\mathcal{I}_0 \models \mathsf{q}(\vec{a})$?' as a model checking problem for first-order formulas over the ABox $\mathcal{A}$ considered as a first-order model, also denoted by $\mathcal{A}$, with domain $\mathit{ob}(\mathcal{A}) \cup \mathit{val}(\mathcal{A})$; we assume that this first-order model also contains all datatype extensions. Now we define a first-order formula $\varphi_{\mathcal{T},\mathsf{q}}(\vec{x})$ in the signature of $\mathcal{T}$ and q such that (*i*) $\varphi_{\mathcal{T},\mathsf{q}}(\vec{x})$ depends on $\mathcal{T}$ and q but not on $\mathcal{A}$, and (*ii*) $\mathcal{A} \models \varphi_{\mathcal{T},\mathsf{q}}(\vec{a})$ iff $\mathcal{I}_0 \models \mathsf{q}(\vec{a})$.

Denote by $\mathit{con}(\mathcal{K})$ the set of basic concepts in $\mathcal{K}$ together with all concepts of the form $\forall U.T$, for attribute names U and datatypes T from $\mathcal{T}$. We begin by defining formulas $\psi_B(x)$, for $B \in \mathit{con}(\mathcal{K})$, that describe the types of named objects (cf. (1)): for all $a_i \in \mathit{ob}(\mathcal{A})$,

$$\mathcal{A} \models \psi_B(a_i) \text{ iff } a_i^{\mathcal{I}_0} \in B^{\mathcal{I}_0}, \quad \text{if } B \text{ is a basic concept,} \tag{5}$$

$$\mathcal{A} \models \psi_{\forall U.T}(a_i) \text{ iff } a_i^{\mathcal{I}_0} \in B_1^{\mathcal{I}_0}, \dots, B_k^{\mathcal{I}_0} \text{ and } \mathcal{T} \models B_1 \sqcap \dots \sqcap B_k \sqsubseteq \forall U.T. \tag{6}$$

These formulas are defined as the 'fixed-points' of sequences $\psi_B^0(x), \psi_B^1(x), \dots$ defined by taking $\psi_B^0(x) = B^*$ if B is A, $\bot$ or $\top$, $\psi_B^0(x) = \bigvee_{H' \sqsubseteq^*_{\mathcal{T}} H} \exists y\, H'(x, y)$ if $B = \exists H$, $\psi_B^0(x) = \bot$ if $B = \forall U.T$ and

$$\psi_B^i(x) = \psi_B^0(x) \vee \bigvee_{B_1 \sqcap \dots \sqcap B_k \sqsubseteq B \in \mathsf{ext}(\mathcal{T})} \big(\psi_{B_1}^{i-1}(x) \wedge \dots \wedge \psi_{B_k}^{i-1}(x)\big),$$

where $\mathsf{ext}(\mathcal{T})$ is the extension of $\mathcal{T}$ with the following:

- $\exists H \sqsubseteq \exists H'$, for all $H \sqsubseteq^*_{\mathcal{T}} H'$,
- $\forall U.T \sqsubseteq \forall U'.T$, for all $U' \sqsubseteq^*_{\mathcal{T}} U$ and $T \in \mathit{dt}(\mathcal{K})$,
- $\forall U.T_1 \sqcap \dots \sqcap \forall U.T_k \sqsubseteq \forall U.T$, for all $T_1 \cap \dots \cap T_k \subseteq_{\mathcal{D}} T$.

(We again assume that all number restrictions are of the form $\exists R$ and $\exists U$; the full version [4] treats arbitrary number restrictions). It should be clear that there is N with $\psi_B^N(x) \equiv \psi_B^{N+1}(x)$, for all B at the same time, and that N does not exceed the cardinality of $\mathit{con}(\mathcal{K})$. We set $\psi_B(x) = \psi_B^N(x)$.

Next we define sentences $\theta_{B,dr}$, for $B \in \mathit{con}(\mathcal{K})$ and dr with $R \in \mathit{role}^{\pm}(\mathcal{K})$, that describe types of the unnamed points, i.e., copies of the dr (cf. (2)): for all w with $\mathit{cp}(w) = dr$,

$$\mathcal{A} \models \theta_{B,dr} \text{ iff } w \in B^{\mathcal{I}_0}, \text{ if } B \text{ is a basic concept,} \tag{7}$$

$$\mathcal{A} \models \theta_{\forall U.T,dr} \text{ iff } \mathcal{T} \models \exists R \sqsubseteq \forall U.T. \tag{8}$$

Note that the type of copies of dr is determined by a single concept, $\exists R$, and therefore, there is no need to consider conjunctions in (8); see also (6). We inductively define a sequence $\theta_{B,dr}^0, \theta_{B,dr}^1, \dots$ by taking $\theta_{B,dr}^0 = \top$ if $B = \exists R$ and $\theta_{B,dr}^0 = \bot$ otherwise, with $\theta_{B,dr}^i$ defined similarly to ψ_B^i above. As with the ψ_B, set $\theta_{B,dr} = \theta_{B,dr}^N$.

Now, suppose $\mathcal{I}_0 \models^{\mathfrak{a}_0} \varphi(\vec{a}, \vec{y})$ and $\mathfrak{a}_0(y_i) \in W_{m_0}$, for every $y_i \in \vec{y}$, where m_0 is as in Lemma 8. Recall that our aim is to compute

the answer to this query in the first-order model $\mathcal{A}$ representing the ABox. This model, however, does not contain points in $W_{m_0} \setminus W_0$, and to represent them, we use the following 'trick.' By **(forest)**, every $w \in W_{m_0}$ is uniquely determined by a pair (a, σ), where a is the root of the tree $\mathfrak{T}_a$ containing w and σ is the sequence of labels $\ell_a(u, v)$ on the path from a to w. Not every such pair, however, corresponds to an element in W_{m_0}. In order to identify points in W_{m_0}, we consider the following directed graph $G_{\mathcal{T}} = (V_{\mathcal{T}}, E_{\mathcal{T}})$, where $V_{\mathcal{T}}$ is the set of equivalence classes $[H] = \{H' \mid H \sqsubseteq^*_{\mathcal{T}} H' \text{ and } H' \sqsubseteq^*_{\mathcal{T}} H\}$ and $E_{\mathcal{T}}$ is the set of all pairs $([R], [H])$ such that $\mathcal{T} \models \exists R^- \sqsubseteq \exists H$ and $R^- \not\sqsubseteq^*_{\mathcal{T}} H$, and H has no proper sub-role/attribute satisfying this property. Let $\Sigma_{\mathcal{T},m_0}$ be the set of all paths in the graph $G_{\mathcal{T}}$ of length $\leq m_0$: more precisely,

$$\Sigma_{\mathcal{T},m_0} = \big\{\varepsilon\big\} \cup V_{\mathcal{T}} \cup \big\{([H_1], \dots, [H_n]) \mid 2 \leq n \leq m_0 \text{ and } ([H_j], [H_{j+1}]) \in E_{\mathcal{T}}, \text{ for } 1 \leq j < n\big\}.$$

By the unravelling procedure, we have $\sigma \in \Sigma_{\mathcal{T},m_0}$, for all pairs (a, σ) representing elements of W_{m_0}. We note, however, that a pair (a, σ) with $\sigma = ([H], \dots) \in \Sigma_{\mathcal{T},m_0}$ corresponds to a $w \in W_{m_0}$ only if a has not enough H-witnesses in $\mathcal{A}$.

In the first-order rewriting $\varphi_{\mathcal{T},\mathsf{q}}$ we are about to define we assume that the bound variables y_i range over W_0 and represent the first component of the pairs (a, σ) (these y_i should not be confused with the y_i in the original query q, which range over W_{m_0}), whereas the second component is encoded in the ith member σ_i of a vector $\vec{\sigma}$. Note that constants and free variables need no second component, σ, and, to unify the notation, for each term t we denote its σ-component by $t^{\vec{\sigma}}$, which is defined as follows: $t^{\vec{\sigma}} = \varepsilon$ if t is a constant or free variable and $t^{\vec{\sigma}} = \sigma_i$ if $t = y_i$.

Let k be the number of bound variables y_i and let $\Sigma^k_{\mathcal{T},m_0}$ be the set of k-tuples $\vec{\sigma} = (\sigma_1, \dots, \sigma_k)$ with $\sigma_i \in \Sigma_{\mathcal{T},m_0}$. Given an assignment $\mathfrak{a}_0$ in W_{m_0}, we denote by $\mathit{split}(\mathfrak{a}_0)$ the pair $(\mathfrak{a}, \vec{\sigma})$ made of an assignment $\mathfrak{a}$ in $\mathcal{A}$ and $\vec{\sigma} \in \Sigma^k_{\mathcal{T},m_0}$ such that $t^{\vec{\sigma}} = ([H_1], \dots, [H_n])$, for a sequence $H_1, \dots, H_n$ of ℓ_a-labels on the path from a to $\mathfrak{a}_0(t)$. We define now, for every $\vec{\sigma} \in \Sigma^k_{\mathcal{T},m_0}$, concept name A, role or attribute name H and datatype name T:

$$A^{\vec{\sigma}}(t) = \begin{cases} \psi_A(t), & \text{if } t^{\vec{\sigma}} = \varepsilon, \\ \theta_{A,\mathit{inv}(ds)}, & \text{if } t^{\vec{\sigma}} = \sigma'.[S], \end{cases}$$

$$H^{\vec{\sigma}}(t_1, t_2) = \begin{cases} H^{\mathcal{T}}(t_1, t_2), & \text{if } t_1^{\vec{\sigma}} = t_2^{\vec{\sigma}} = \varepsilon, \\ (t_1 = t_2), & \text{if } t_1^{\vec{\sigma}}.[S] = t_2^{\vec{\sigma}}, \text{ or } t_2^{\vec{\sigma}} = t_1^{\vec{\sigma}}.[S^-], \text{ for } S \sqsubseteq^*_{\mathcal{T}} H, \\ \bot, & \text{otherwise}, \end{cases}$$

$$T^{\vec{\sigma}}(t) = \begin{cases} T(t), & \text{if } t^{\vec{\sigma}} = \varepsilon, \\ \psi_{\forall U.T}(t), & \text{if } t^{\vec{\sigma}} = [U], \\ \theta_{\forall U.T,ds^-}, & \text{if } t^{\vec{\sigma}} = \sigma'.[S].[U]. \end{cases}$$

LEMMA 9. *For each assignment $\mathfrak{a}_0$ in W_{m_0} with $\mathit{split}(\mathfrak{a}_0) = (\mathfrak{a}, \sigma)$,*

$$\mathcal{I}_0 \models^{\mathfrak{a}_0} A(t) \text{ iff } \mathcal{A} \models^{\mathfrak{a}} A^{\vec{\sigma}}(t), \text{ for concept names } A,$$
$$\mathcal{I}_0 \models^{\mathfrak{a}_0} H(t_1, t_2) \text{ iff } \mathcal{A} \models^{\mathfrak{a}} H^{\vec{\sigma}}(t_1, t_2), \text{ for roles and attribute names } H,$$
$$\mathcal{I}_0 \models^{\mathfrak{a}_0} T(t) \text{ iff } \mathcal{A} \models^{\mathfrak{a}} T^{\vec{\sigma}}(t), \text{ for datatype names } T.$$

Finally, we define the first-order rewriting of q and $\mathcal{T}$ by taking:

$$\varphi_{\mathcal{T},\mathsf{q}}(\vec{x}) = \exists \vec{y} \bigvee_{\vec{\sigma} \in \Sigma^k_{\mathcal{T},m_0}} \Big(\varphi^{\vec{\sigma}}(\vec{x}, \vec{y}) \wedge \bigwedge_{\substack{1 \leq i \leq k \\ \sigma_i = ([H_i], \dots) \neq \varepsilon}} \big(\neg \psi^0_{\exists H_i}(y_i) \wedge \psi_{\exists H_i}(y_i)\big)\Big),$$

where $\varphi^{\vec{\sigma}}(\vec{x}, \vec{y})$ is the result of attaching the superscript $^{\vec{\sigma}}$ to each atom of φ; the last conjunct ensures that each pair (a, σ_i) corresponds an element of $w \in W_{m_0}$. Correctness of this rewriting follows from Lemma 9, see the full version [4]. □

6 Conclusions

We extended *DL-Lite* with *local attributes*—allowing the use of the same attribute associated to different concepts—and *safe* datatypes where datatype constraints can be expressed with Horn-like clauses. Notably, this is the first time that *DL-Lite* is equipped with a form of the universal restriction $\forall U.T$. We showed that such an extension is harmless with the only exception of the Krom fragment, where the complexity rises from NLOGSPACE to NP. We studied also the problem of answering positive existential queries and showed that for the Horn and core extensions the problem remains in AC^0(i.e., FO-rewritable).

As a future work we are interested in relaxing the safe condition for datatypes, in particular we conjecture that the restriction on the boundedness of datatype difference can be relaxed for particular concrete domains.

REFERENCES

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

[2] A. Artale, D. Calvanese, R. Kontchakov, and M. Zakharyaschev, '*DL-Lite* in the light of first-order logic', in *Proc. of the 22nd Nat. Conf. on Artificial Intelligence (AAAI 2007)*, pp. 361–366, (2007).

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

[4] A. Artale, R. Kontchakov, and V. Ryzhikov, 'DL-Lite with attributes, datatypes and sub-roles (full version)', Technical Report BBKCS-12-01, Department of Computer Science and Information Systems, Birkbeck, University of London, (2012).

[5] F. Baader, S. Brandt, and C. Lutz, 'Pushing the $\mathcal{EL}$ envelope', in *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence, IJCAI-05*. Morgan-Kaufmann Publishers, (2005).

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

[7] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, and R. Rosati, '*DL-Lite*: Tractable description logics for ontologies', in *Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI)*, pp. 602–607, (2005).

[8] 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', *Journal of Automated Reasoning*, **39**(3), 385–429, (2007).

[9] D. Calvanese, M. Lenzerini, and D. Nardi, 'Unifying class-based representation formalisms', *Journal of Artificial Intelligence Research*, **11**, 199–240, (1999).

[10] B. Cuenca Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, and U. Sattler, 'OWL 2: The next step for OWL', *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, **6**(4), 309–322, (2008).

[11] M. Despoina, Y. Kazakov, and I. Horrocks, 'Tractable extensions of the description logic EL with numerical datatypes', *Journal of Automated Reasoning*, (2011).

[12] E. Franconi, Y. A. Ibáñez-García, and I. Seylan, 'Query answering with DBoxes is hard', *Electr. Notes Theor. Comput. Sci.*, **278**, 71–84, (2011).

[13] C. Lutz, 'Description logics with concrete domains—a survey', in *Advances in Modal Logics Volume 4*. King's College Publications, (2003).

[14] J. Pan and I. Horrocks, 'OWL-Eu: Adding customised datatypes into OWL', *Web Semantics: Science, Services and Agents on the World Wide Web*, **4**(1), (2011).

[15] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati, 'Linking Data to Ontologies', *Journal on Data Semantics*, **X**, 133–173, (2008).

[16] O. Savković, *Managing Datatypes in Ontology-Based Data Access*, MSc dissertation, European Master in Computational Logic, Faculty of Computer Science, Free University of Bozen-Bolzano, October 2011.

[17] A. Schaerf, 'On the complexity of the instance checking problem in concept languages with existential quantification', *Journal of Intelligent Information Systems*, **2**, 265–278, (1993).