On the Semantic Relationship between Datalog and Description Logics

Markus Krötzsch[1], Sebastian Rudolph[1], Peter H. Schmitt[2]

[1] Institute AIFB, Karlsruhe Institute of Technology, DE, firstname.lastname@kit.edu
[2] Institute for Theoretical Computer Science, Karlsruhe Institute of Technology, DE, pschmitt@ira.uka.de

Abstract. Translations to (first-order) datalog have been used in a number of inferencing techniques for description logics (DLs), yet the relationship between the semantic expressivities of function-free Horn logic and DL is understood only poorly. Although Description Logic Programs (DLP) have been described as DLs in the "expressive intersection" of DL and datalog, it is unclear what an intersection of two syntactically incomparable logics is, even if both have a first-order logic semantics. In this work, we offer a characterisation for DL fragments that can be expressed, in a concrete sense, in datalog. We then determine the largest such fragment for the DL $\mathcal{ALC}$, and provide an outlook on the extension of our methods to more expressive DLs.

1 Introduction

Ontologies and rules are two fundamental concepts in knowledge representation. Taking ontologies as the basic modelling paradigm has led to the development of Description Logics (DLs) with a wide range of successful knowledge representations languages. On the other hand rules are the central notion in Logic Programming building on first-order Horn logic. Both have been very prolific research areas and have recently received a boost in the context of the Semantic Web. As references for the purposes of this paper we point to [2] and [4]. Since decidability is an important concern for DL, function-free first-order Horn logic "*datalog*" is of particular interest.

Since the semantic frameworks for DL and datalog are very close it is natural that the research community started investigating the relationship between them. One direction explores how either formalism could be extended with features of the other. This line of research is represented by approaches such as $\mathcal{AL}$-log [8], CARIN [23], SWRL [13,14], $\mathcal{DL}$+*log* [27], DL-safe rules [26], DL Rules [21,11], but also Datalog$^{\pm}$ [5], and $\forall\exists$-rules [3]. Another direction aims at pin-pointing how both formalisms overlap. This has led to the study of Horn description logics [15,20] and Description Logic Programs (DLP) [12,28]. The latter is a family of DLs that can be faithfully expressed in first-order Horn-logic, and in particular in datalog, and the generalisation of this approach is the main topic of this paper.[3]

[3] Besides these two strands on integrating first-order rules with DLs, there are numerous works on extending DLs with non-monotonic features from logic programming [10,9,27,24,25] which are interesting in their own right but not closely related to this work.

It is known that fragments of various DLs can be translated into equivalent or equisatisfiable datalog programs, and this has also been exploited to solve reasoning tasks. This has been demonstrated, e.g., for the description logics Horn-$\mathcal{SHIQ}$ and $\mathcal{EL}^{++}$ [15,22,17,18]. In this paper we address the question whether there is a maximal fragment that can be mapped into datalog. This would give a precise meaning to the slogan of the "expressive intersection" of DL and datalog. The failure of naive attempts to define maximal fragments eventually led to the definition of a DLP fragment for a given DL in Section 3 below. In Section 4 we define the DLP fragment $\mathcal{DLP}_{\mathcal{ALC}}$ of $\mathcal{ALC}$ and prove its maximality. This result can be extended to $\mathcal{SROIQ}$ but the necessary canonical syntactic descriptions are too complex to be included in this paper. We thus rather provide a summary of the relevant results and methods in Section 5 and refer for details and omitted proofs to the technical report [19].

2 Preliminaries

We assume the reader to be familiar with DLs (see [19,2] for details and references), and restrict to notational remarks here. The largest DL we encounter is $\mathcal{SROIQ}^{\text{free}}$, the well-known DL $\mathcal{SROIQ}$ without any restrictions on *simplicity* and *regularity* of roles, though only the simpler DL $\mathcal{ALC}$ will be considered in detail within this paper. DL knowledge bases are defined over finite sets of individual names (constants) $\mathbf{I}$, concept names $\mathbf{A}$, and roles $\mathbf{R}$. We call $\mathscr{S} = \langle \mathbf{I}, \mathbf{A}, \mathbf{R} \rangle$ a *signature*. A signature $\mathscr{S}' = \langle \mathbf{I}', \mathbf{A}', \mathbf{R}' \rangle$ is called an *extension* of $\mathscr{S}$, in symbols $\mathscr{S} \subseteq \mathscr{S}'$, if $\mathbf{I} \subseteq \mathbf{I}'$ and $\mathbf{A} \subseteq \mathbf{A}'$ and $\mathbf{R} \subseteq \mathbf{R}'$.

We use $\mathbf{FOL}_{=}$ to refer to standard first-order logic with equality. It is well known in the folklore of DL and easy to see that there exists a translation π of $\mathcal{SROIQ}$ and thus also of $\mathcal{ALC}$ into $\mathbf{FOL}_{=}$ that preserves logical inference, i.e. $\text{KB}_1 \models \text{KB}_2$ implies $\pi(\text{KB}_1) \models \pi(\text{KB}_2)$. A definition of π may e.g., be found in [19, Figure 3.4].

We use the term "*datalog*" to refer to the function-free Horn logic fragment of $\mathbf{FOL}_{=}$. A *datalog program* is a first-order theory which contains only formulae of the form $\forall \mathbf{x}.A_1 \wedge \ldots \wedge A_n \rightarrow B$ where A_i, B are atoms without function symbols of arity greater than 0, and universal quantifies over all variables occurring in the implications. We generally omit the quantifier, we simply write B if $n = 0$, and we use $\bot$ to denote the empty head.

It will not be sufficient for our work to consider knowledge bases KB such that π(KB) is equivalent to a datalog program. Semantic equivalence turns out to be too restrictive, it does e.g., not allow the use of new constant symbols denoting individuals whose existence is required by ABox axioms. Equisatisfiability on the other hand is too weak – it does not preserve relevant logical entailments. The following notion turns out to be a more appropriate middle-ground:

Definition 1. *Given* $\mathbf{FOL}_{=}$ *theories T and T' with signatures $\mathscr{S} \subseteq \mathscr{S}'$, then T'* semantically emulates *T if*

(1) every model of T' becomes a model of T when restricted to the interpretations of symbols from $\mathscr{S}$, and
(2) for every model $\mathcal{J}$ of T there is a model $\mathcal{I}$ of T' that has the same domain as $\mathcal{J}$, and that agrees with $\mathcal{J}$ on $\mathscr{S}$.

It is usually not necessary to mention the signatures of T and T' explicitly, since it is always possible to find minimal signatures for T and T' that satisfy condition (1) of Definition 1. The concept of semantic emulation is also known by the name *semantic conservative extension*. We will prefer semantic emulation for its brevity.

Definition 2. *Given* **FOL**$_=$ *theories T and T' with signatures $\mathscr{S} \subseteq \mathscr{S}'$, then T'* syntactically emulates *T if for every first-order formula φ over $\mathscr{S}$: $T \models \varphi$ iff $T' \models \varphi$.*

It is easy to see that *semantic emulation* implies *syntactic emulation*. This illustrates the strength and significance of *semantic emulation* for knowledge representation: whenever a theory T' semantically emulates a theory T, we find that T' and T encode the same information *about the symbols* in T, and in particular that T' cannot be distinguished from T when restricting to those symbols.

Note, *syntactic emulation* of T by T' can equivalently be characterized by the requirement that for every formula φ over $\mathscr{S}$ the sets $T \cup \{\varphi\}$ and $T' \cup \{\varphi\}$ be equisatisfiable.

We will later make use of the following lemma, which generalises the well-known least model property of datalog. The proof of this is straightforward by unravelling of the definitions.

Lemma 1. *Let $\mathcal{I}_1$, $\mathcal{I}_2$ be interpretations over the same domain which agree on the interpretation of constant and function symbols, and let T be a first-order theory that is satisfied by $\mathcal{I}_1$ and $\mathcal{I}_2$.*

1. *If T is a datalog program then also the intersection $\mathcal{I}_1 \cap \mathcal{I}_2$ satisfies T.*
2. *If T can be semantically emulated by a datalog program then also the intersection $\mathcal{I}_1 \cap \mathcal{I}_2$ satisfies T.*

The intersection of interpretations is defined in the obvious way based on the intersection of predicate extensions.

3 Considerations for Defining DLP

In this section, we discuss and motivate a generic definition for DLP fragments of a description logic. A powerful tool for obtaining this definition is the construction of variants of logical expressions which preserve only the logical structure but may modify concrete signature symbols:

Definition 3. *Let F be a* **FOL**$_=$ *formula, a DL axiom, or a DL concept expression, and let $\mathscr{S}$ be a signature. An expression F' is a* variant *of F in $\mathscr{S}$ if F' can be obtained from F by replacing each occurrence of a role/concept/individual name with some role/concept/individual name in $\mathscr{S}$. Multiple occurrences of the same entity name in F need* not *be replaced by the same entity name of $\mathscr{S}$ in this process.*

A knowledge base KB′ *is a* variant *of a knowledge base* KB *if it is obtained from* KB *by replacing each axiom with a variant.*

Note that we do not require all occurrences of an entity name to be renamed together, so it is indeed possible to obtain $A \sqcap \neg B$ from $A \sqcap \neg A$. Considering all variants of

a formula or axiom allows us to study the semantics and expressivity of formulae based on their syntactic structure, disregarding any possible interactions between signature symbols. We therefore call a $\mathbf{FOL}_=$ formula, DL axiom, or DL concept expression F *name-separated* if no signature symbol occurs more than once in F.

Definition 4. *Given description logics $\mathcal{L}$ and $\mathcal{D}$, we call $\mathcal{D}$ a* DLP fragment *of $\mathcal{L}$ if*

(1) every axiom of $\mathcal{D}$ is an axiom of $\mathcal{L}$,
(2) there is a transformation function datalog *that maps a $\mathcal{D}$ axiom α to a datalog program* datalog(α) *such that* datalog(α) *semantically emulates α,*
(3) $\mathcal{D}$ is closed under variants, i.e. given any axiom α and an arbitrary variant α' of α, we find α is in $\mathcal{D}$ iff α' is.

Item (1) of this definition fixes the syntactic framework for DLP fragments. Item (2) states the property that motivates the study of DLP languages: every axioms of a DLP fragment can be expressed in datalog. DLP languages as discussed in the literature may require the use of auxiliary symbols for the translation to datalog [28], and the datalog program can no longer be semantically equivalent to the original knowledge base in this case, even if all consequences with respect to the original predicates are still the same. This motivates the use of semantic emulation as introduced in Definition 1.

Item (3) of Definition 4 reflects our desire to obtain fragments that correspond to well-behaved logical languages as opposed to being arbitrary collections of axioms. An obvious way to implement this would be to require DLP fragments to be described by a context-free grammar. A typical feature of grammars for logical languages is that they are parametrised by a logical signature that can be modified without changing the essential structural features of the language. This effect is mirrored by the requirement of item (3) without introducing detailed requirements on a suitable logical grammar. We will find grammatical descriptions in the cases we consider, though item (3) as such does not imply that this is possible.

Let us discuss for a moment an alternative to item (3) in Definition 4. It seems natural to require that membership in a fragment can be decided efficiently, say in polynomial time. Proposition 1 shows that in this case no maximal fragment can exist. Definition 4 allows fragments without any restriction on the complexity of the membership relation, but the maximal DLP fragment of $\mathcal{ALC}$ in Section 4 is described by a context-free language, and thus efficiently recognisable.

Proposition 1. *Given description logics $\mathcal{L}$ and $\mathcal{D}$, we call $\mathcal{D}$ a* P-DLP fragment *of $\mathcal{L}$ if items (1) and (2) of Definition 4 are satisfied, and in addition there is a polynomial procedure for deciding $\alpha \in \mathcal{D}$ for any DL axiom α.*

Unless the complexity classes P *and* PSpace *coincide, there is no maximal* P*-DLP fragment of $\mathcal{ALC}$: given any* P*-DLP fragment $\mathcal{D}$ of $\mathcal{ALC}$, there is a* P*-DLP fragment $\mathcal{D}'$ of $\mathcal{ALC}$ that covers more axioms, i.e. $\mathcal{D} \subset \mathcal{D}'$.*

Proof. We first need to argue that, even with unlimited resources for the datalog translation, it is not possible that DLP supports all $\mathcal{ALC}$ axioms. We show that, if the concept expression C is satisfiable and does not contain the symbols R, A_1, A_2, and c, then no datalog program semantically emulates the axiom $\alpha := (C \sqcap \exists R.(A_1 \sqcup A_2))(c)$. For a contradiction, suppose that α is semantically emulated by a datalog theory datalog(α). By

construction, α is satisfiable, and so is $\{\alpha, A_i \sqsubseteq \bot\}$ for each $i = 1, 2$. By Definition 2, we find that $\mathsf{datalog}(\alpha) \cup \{A_i \sqsubseteq \bot\}$ is satisfiable, too. Thus, there are models $\mathcal{I}_i$ of $\mathsf{datalog}(\alpha)$ such that $A_i^{\mathcal{I}_i} = \emptyset$. By the least model property of datalog, there is also a model $\mathcal{I}$ of $\mathsf{datalog}(\alpha)$ such that $A_1^{\mathcal{I}} = A_2^{\mathcal{I}} = \emptyset$. But then $\mathsf{datalog}(\alpha) \cup \{A_1 \sqcup A_2 \sqsubseteq \bot\}$ is satisfiable although $\{\alpha, A_1 \sqcup A_2 \sqsubseteq \bot\}$ is not, contradicting the supposed semantic emulation.

We can now show that there is some unsatisfiable axiom that is not in $\mathcal{D}$. To this end, recall that deciding (un)satisfiability of $\mathcal{ALC}$ concept expressions is PSPACE hard. However, we just showed that, if the axiom $\alpha = (C \sqcap \exists R.(A_1 \sqcup A_2))(c)$ is in $\mathcal{D}$ with symbols R, A_1, A_2, c not in C, then the concept C is unsatisfiable. Thus, if $\mathcal{D}$ contains all unsatisfiable $\mathcal{ALC}$ axioms of the form of α, then deciding whether $\alpha \in \mathcal{D}$ is equivalent to deciding whether C is unsatisfiable (since one can clearly construct α from C in polynomial time). By the assumed efficient recognisability, this would yield a polynomial decision procedure for $\mathcal{ALC}$ concept satisfiability – a contradiction.

Therefore, there is an unsatisfiable axiom α with $\alpha \notin \mathcal{D}$. Now let $\mathcal{D}'$ be defined as $\mathcal{D} \cup \{\alpha\}$. The transformation is given by $\mathsf{datalog}'(\alpha) = \mathsf{datalog}(\alpha)$ if $\alpha \in \mathcal{D}$, and $\mathsf{datalog}'(\alpha) = \{\top \to A(x), A(x) \to \bot\}$ otherwise, where A is a new predicate symbol. It is immediate that this defines a P-DLP fragment of $\mathcal{ALC}$. □

This proof exemplifies a general problem that occurs when trying to define DLP: the question whether an axiom is expressible in datalog is typically computationally harder than one would like to admit for a language definition. This result carries over to more expressive DLs, and remains valid even if requirements such as closure under common normal form transformations are added to the definition of fragments. The fact that this problem is avoided by item (3) in Definition 4 confirms our intuition that this requirement closely relates to the possibility of representing DLP fragments syntactically, i.e. without referring to complex semantic conditions.

Proposition 2. *Consider a class K of knowledge bases that belong to a DLP fragment of some description logic, and such that the maximal size of axioms in K is bounded. Deciding satisfiability of knowledge bases in K is possible in polynomial time.*

Proof. Let the maximal size of axioms be bounded by N. Let V be a vocabulary with N concept, role and constant symbols. By assumption we know that for every of the finitely many axioms α of size less than N there is a translation $\mathit{datalog}(\alpha)$. We use this as a (finite) look-up table in the definition of $\mathit{datalog}_K(\beta)$ for axioms β in KB $\in K$: Find a renaming $\alpha = \sigma(\beta)$ such that α is an expression in the vocabulary V. Here σ is a usual 1-1 renaming of symbols, not a variant in the sense of Definition 3. Look up the datalog program $\mathit{datalog}(\alpha)$ and set $\mathit{datalog}_{\mathrm{KB}}(\beta) = \sigma(\mathit{datalog}(\alpha))$. It is easy to see that $\mathsf{datalog}_{\mathrm{KB}}(\beta)$ still satisfies item (2) of Definition 4. Thus satisfiability of KB $\in K$ can be decided by checking satisfiability of $\bigcup_{\beta \in \mathrm{KB}} \mathsf{datalog}_{\mathrm{KB}}(\beta)$. The maximal number of variables occurring within these datalog programs may also be bounded by N. Satisfiability of datalog with at most N variables per rule can be decided in time polynomial in 2^N [7]. The renamings σ can likewise be found in time polynomial in 2^N. Since N is a constant, this yields a polynomial time upper bound for deciding satisfiability of knowledge bases in K. □

It is interesting that the previous result does not require any assumptions on the computational complexity of recognising or translating DLP axioms. Intuitively, Propo-

Concepts necessarily equivalent to $\top$:	$\mathbf{L}^{\mathcal{A}}_{\top} ::= \top \mid \forall \mathbf{R}.\mathbf{L}^{\mathcal{A}}_{\top} \mid \mathbf{L}^{\mathcal{A}}_{\top} \sqcap \mathbf{L}^{\mathcal{A}}_{\top} \mid \mathbf{L}^{\mathcal{A}}_{\top} \sqcup \mathbf{C}$
Concepts necessarily equivalent to $\bot$:	$\mathbf{L}^{\mathcal{A}}_{\bot} ::= \bot \mid \exists \mathbf{R}.\mathbf{L}^{\mathcal{A}}_{\bot} \mid \mathbf{L}^{\mathcal{A}}_{\bot} \sqcap \mathbf{C} \mid \mathbf{L}^{\mathcal{A}}_{\bot} \sqcup \mathbf{L}^{\mathcal{A}}_{\bot}$
Body ($C \in \mathbf{L}^{\mathcal{A}}_{B}$ iff $\neg C \sqsubseteq A$ in $\mathcal{DLP}_{\mathcal{ALC}}$):	$\mathbf{L}^{\mathcal{A}}_{B} ::= \mathbf{L}^{\mathcal{A}}_{\top} \mid \mathbf{L}^{\mathcal{A}}_{\bot} \mid \neg \mathbf{A} \mid \forall \mathbf{R}.\mathbf{L}^{\mathcal{A}}_{B} \mid \mathbf{L}^{\mathcal{A}}_{B} \sqcap \mathbf{L}^{\mathcal{A}}_{B} \mid \mathbf{L}^{\mathcal{A}}_{B} \sqcup \mathbf{L}^{\mathcal{A}}_{B}$
Head ($C \in \mathbf{L}^{\mathcal{A}}_{H}$ iff $A \sqsubseteq C$ in $\mathcal{DLP}_{\mathcal{ALC}}$):	$\mathbf{L}^{\mathcal{A}}_{H} ::= \mathbf{L}^{\mathcal{A}}_{B} \mid \mathbf{A} \mid \forall \mathbf{R}.\mathbf{L}^{\mathcal{A}}_{H} \mid \mathbf{L}^{\mathcal{A}}_{H} \sqcap \mathbf{L}^{\mathcal{A}}_{H} \mid \mathbf{L}^{\mathcal{A}}_{H} \sqcup \mathbf{L}^{\mathcal{A}}_{B}$
Assertions ($C \in \mathbf{L}^{\mathcal{A}}_{a}$ iff $C(a)$ in $\mathcal{DLP}_{\mathcal{ALC}}$):	$\mathbf{L}^{\mathcal{A}}_{a} ::= \mathbf{L}^{\mathcal{A}}_{H} \mid \exists \mathbf{R}.\mathbf{L}^{\mathcal{A}}_{a} \mid \mathbf{L}^{\mathcal{A}}_{a} \sqcap \mathbf{L}^{\mathcal{A}}_{a} \mid \mathbf{L}^{\mathcal{A}}_{a} \sqcup \mathbf{L}^{\mathcal{A}}_{B}$

Fig. 1. $\mathcal{DLP}_{\mathcal{ALC}}$ concepts in negation normal form

sition 2 states that reasoning in any DLP language is necessarily "almost" tractable. Indeed, many DLs allow complex axioms to be decomposed into a number of simpler normal forms of bounded size, and in any such case tractability is obtained. Moreover, Proposition 2 clarifies why Horn-$\mathcal{SHIQ}$ cannot be in DLP: ExpTime worst-case complexity of reasoning can be proven for a class K of Horn-$\mathcal{SHIQ}$ knowledge bases as in the above proposition (see [20], noting that remaining complex axioms can be decomposed in Horn-$\mathcal{SHIQ}$).

4 The DLP Fragment of $\mathcal{ALC}$

Using Definition 4, it is now possible to investigate DLP fragments of relevant description logics. In this paper, we detail this approach for $\mathcal{ALC}$; some remarks on the more complex case of $\mathcal{SROIQ}$ are given in Section 5 below. It turns out that the largest DLP fragment of $\mathcal{ALC}$ exists, and can be defined as follows, where we use the negation normal form NNF for simplifying our presentation.

Definition 5. *We define the description logic $\mathcal{DLP}_{\mathcal{ALC}}$ to contain all knowledge bases consisting only of $\mathcal{ALC}$ axioms which are*

- *GCIs $C \sqsubseteq D$ such that $\mathsf{NNF}(\neg C \sqcup D)$ is an $\mathbf{L}^{\mathcal{A}}_{H}$ concept as defined in Fig. 1, or*
- *ABox axioms $C(a)$ where $\mathsf{NNF}(C)$ is an $\mathbf{L}^{\mathcal{A}}_{a}$ concept as defined in Fig. 1.*

The headings in Fig. 1 give the basic intuition about the significance of the various concept languages. The distinction of head and body concepts is typical for many works on DLP and Horn DLs, while our use of additional assertional concepts takes into account that emulation allows for some forms of Skolemisation. Typical example representatives of the respective grammars are $\neg A \sqcap \forall R.(\neg B \sqcup \neg C)$ for $\mathbf{L}^{\mathcal{A}}_{B}$, $\neg A \sqcup (B \sqcap \forall R.C)$ for $\mathbf{L}^{\mathcal{A}}_{H}$, and $\neg A \sqcup \exists R.B$ for $\mathbf{L}^{\mathcal{A}}_{a}$.

Though name separation prevents most forms of semantic interactions within concepts, we still require grammars for $\mathbf{L}^{\mathcal{A}}_{\top}$ and $\mathbf{L}^{\mathcal{A}}_{\bot}$ to characterise concepts all variants of which are equivalent to $\top$ and $\bot$, respectively. This includes concept expressions such as $A \sqcap \exists R.\bot$ and $B \sqcup \forall R.\top$.

We start with an easy observation on Definition 5. This result will not explicitly be used later on but might add to the understanding of this definition.

Lemma 2. *Consider arbitrary $\mathcal{ALC}$ concept expressions C that do not contain quantifiers $\forall$, $\exists$, and the symbols $\top$ and $\bot$.*

1. *If $C \in \mathbf{L}_B^{\mathcal{A}}$ then C has a conjunctive normal form $\bigsqcap_i \bigsqcup_j C_{i,j}$ with $C_{i,j}$ a negated atom for all i, j.*
2. *If $C \in \mathbf{L}_H^{\mathcal{A}}$ or $C \in \mathbf{L}_a^{\mathcal{A}}$ then C has a conjunctive normal form $\bigsqcap_i \bigsqcup_j C_{i,j}$ with $C_{i,j}$ negated or unnegated atoms and for every i there is at most one j such that $C_{i,j}$ is an unnegated atom.*
   (Since the assumptions require that C does not contain quantifiers there is no difference here between $C \in \mathbf{L}_H^{\mathcal{A}}$ and $C \in \mathbf{L}_a^{\mathcal{A}}$.)

Proof. Notice, that $C \notin \mathbf{L}_\top^{\mathcal{A}}$ and $C \notin \mathbf{L}_\bot^{\mathcal{A}}$ since neither $\top$ nor $\bot$ occur in C. For item (1), note that if $C \in \mathbf{L}_B^{\mathcal{A}}$ then either C is a negated atom, or $C = C_1 \sqcap C_2$ or $C = C_1 \sqcup C_2$ with $C_i \in \mathbf{L}_B^{\mathcal{A}}$. The claim now follows easily from the induction hypothesis on C_1, C_2.

For item (2), by the assumptions on C we have $C \in \mathbf{L}_H^{\mathcal{A}}$ if one of the following cases holds true:

1. $C \in \mathbf{L}_B^{\mathcal{A}}$. Then the claim follows from part (1) of the lemma.
2. C is an atom. Then the claim is obviously true.
3. $C = C_1 \sqcap C_2$ with $C_i \in \mathbf{L}_H^{\mathcal{A}}$. If C_i' is a conjunctive normal form of C_i satisfying the claim then $C_1' \sqcap C_2'$ is a conjunctive normal form of C satisfying the claim.
4. $C = C_1 \sqcup C_2$ with $C_i \in \mathbf{L}_H^{\mathcal{A}}$ and $C_1 \in \mathbf{L}_B^{\mathcal{A}}$. Let $\bigsqcap_i \bigsqcup_j C_{i,j}^1$ and $\bigsqcap_m \bigsqcup_n C_{m,n}^2$ be the conjunctive normal forms that exist by induction hypothesis satisfying the respective claims. A conjunctive normal form of $C = C_1 \sqcup C_2$ is obtained as the conjunction of all $\bigsqcup_j C_{i,j}^1 \sqcup \bigsqcup_n C_{m,n}^2$ for all combinations of i, m. Since $\bigsqcup_j C_{i,j}^1$ contains at most one positive atom and $\bigsqcup_n C_{m,n}^2$ contains only negative atoms we are finished. □

It is obvious that $\mathcal{DLP}_{\mathcal{ALC}}$ satisfies items (1) and (3) of Definition 4, so what remains to show is that $\mathcal{DLP}_{\mathcal{ALC}}$ knowledge bases can indeed be expressed in datalog. Following the grammatical structure of $\mathcal{DLP}_{\mathcal{ALC}}$, we specify three auxiliary functions for constructing datalog programs to semantically emulate a $\mathcal{DLP}_{\mathcal{ALC}}$ knowledge base. The following two lemmata can be proven by simple inductions, see [19] for further details.

Lemma 3. *Given a concept name A, and a concept $C \in \mathbf{L}_H^{\mathcal{A}}$, Fig. 2 recursively defines a datalog program $\mathsf{dlg}_H^{\mathcal{A}}(A \sqsubseteq C)$ that semantically emulates $A \sqsubseteq C$.*

For an example of this transformation, consider the $\mathbf{L}_H^{\mathcal{A}}$ concept $E = \neg B \sqcup (C \sqcap \forall R.D)$. Then $\mathsf{dlg}_H^{\mathcal{A}}(A \sqsubseteq E)$ consists of the following rules:

$$\begin{aligned} A(x) \wedge X_1(x) &\rightarrow X_2(x) \\ B(x) &\rightarrow X_1(x) \\ X_2(x) &\rightarrow C(x) \\ X_2(x) \wedge R(x,y) &\rightarrow X_3(x) \\ X_3(x) &\rightarrow D(x) \end{aligned}$$

Clearly, this rule set could be further simplified to obtain the three rules $A(x) \wedge B(x) \rightarrow X_2(x)$, $X_2(x) \rightarrow C(x)$, $X_2(x) \wedge R(x,y) \rightarrow D(x)$ which are easily seen to semantically emulate $A \sqsubseteq E$.

Lemma 4. *Given a constant a and a concept $C \in \mathbf{L}_a^{\mathcal{A}}$, Fig. 3 recursively defines a datalog program $\mathsf{dlg}_H^{\mathcal{A}}(C(a), \bot)$ that semantically emulates $C(a)$.*

C	$\mathsf{dlg}_H^{\mathcal{A}}(A \sqsubseteq C)$
$D \in \mathbf{L}_B^{\mathcal{A}}$	$\mathsf{dlg}_B^{\mathcal{A}}(\neg X \sqsubseteq D) \cup \{A(x) \wedge X(x) \rightarrow \bot\}$
B	$\{A(x) \rightarrow B(x)\}$
$\forall R.D$	$\mathsf{dlg}_H^{\mathcal{A}}(X \sqsubseteq D)$ $\cup \{A(x) \wedge R(x,y) \rightarrow X(y)\}$
$D_1 \sqcap D_2$	$\mathsf{dlg}_H^{\mathcal{A}}(A \sqsubseteq D_1) \cup \mathsf{dlg}_H^{\mathcal{A}}(A \sqsubseteq D_2)$
$D_1 \sqcup D_2 \in (\mathbf{L}_H^{\mathcal{A}} \sqcup \mathbf{L}_B^{\mathcal{A}})$	$\mathsf{dlg}_H^{\mathcal{A}}(X_2 \sqsubseteq D_1) \cup \mathsf{dlg}_B^{\mathcal{A}}(\neg X_1 \sqsubseteq D_2) \cup \{A(x) \wedge X_1(x) \rightarrow X_2(x)\}$

C	$\mathsf{dlg}_B^{\mathcal{A}}(\neg A \sqsubseteq C)$
$D \in \mathbf{L}_\top^{\mathcal{A}}$	$\{\}$
$D \in \mathbf{L}_\bot^{\mathcal{A}}$	$\{A(x)\}$
$\neg B$	$\{B(x) \rightarrow A(x)\}$
$\forall R.D$	$\mathsf{dlg}_B^{\mathcal{A}}(\neg X \sqsubseteq D) \cup \{R(x,y) \wedge X(y) \rightarrow A(x)\}$
$D_1 \sqcap D_2 \in (\mathbf{L}_B^{\mathcal{A}} \sqcap \mathbf{L}_B^{\mathcal{A}})$	$\mathsf{dlg}_B^{\mathcal{A}}(\neg A \sqsubseteq D_1) \cup \mathsf{dlg}_B^{\mathcal{A}}(\neg A \sqsubseteq D_2)$
$D_1 \sqcup D_2 \in (\mathbf{L}_B^{\mathcal{A}} \sqcup \mathbf{L}_B^{\mathcal{A}})$	$\mathsf{dlg}_B^{\mathcal{A}}(\neg X_1 \sqsubseteq D_1) \cup \mathsf{dlg}_B^{\mathcal{A}}(\neg X_2 \sqsubseteq D_2) \cup \{X_1(x) \wedge X_2(x) \rightarrow A(x)\}$
A, B concept names, R a role, $X_{(i)}$ fresh concept names	

Fig. 2. Transforming axioms $\mathbf{A} \sqsubseteq \mathbf{L}_H^{\mathcal{A}}$ and $\neg\mathbf{A} \sqsubseteq \mathbf{L}_B^{\mathcal{A}}$ to datalog

Again, this transformation is designed for a concise definition, not for optimised output. For an example, consider the $\mathbf{L}_a^{\mathcal{A}}$ concept $E = \neg B \sqcup \exists R.C$. Then $\mathsf{dlg}_H^{\mathcal{A}}(E(a), \bot)$ consists of the following rules (X_i and Y indicating fresh concept names as in the definition of the transformation):

$$\begin{array}{rcl@{\qquad}rcl} B(x) & \rightarrow & X_1(x) & X_2(a) & \rightarrow & R(a,b) \\ X_2(a) & \rightarrow & Y(b) & X_3(x) \wedge X_4(x) & \rightarrow & X_2(x) \\ & \rightarrow & X_3(x) & X_1(x) & \rightarrow & X_4(x) \\ & \rightarrow & X_5(b) & X_5(x) \wedge X_6(x) & \rightarrow & X_7(x) \\ X_7(x) & \rightarrow & C(x) & Y(x) & \rightarrow & X_6(x) \end{array}$$

As before, this rule set can be simplified significantly by eliminating most of the introduced auxiliary concept symbols. Doing this, we obtain the three rules $B(x) \rightarrow X_2(x)$, $X_2(a) \rightarrow R(a,b)$, and $X_2(a) \rightarrow C(b)$, which again are easily seen to semantically emulate $E(a)$ as claimed. Here, the fresh constant symbol b acts as a Skolem constant that represents the individual that the existential concept expression may require to exist.

Combining the previous lemmata, we obtain the emulation theorem for $\mathcal{DLP}_{\mathcal{ALC}}$.

Theorem 1. *For every $\mathcal{DLP}_{\mathcal{ALC}}$ axiom α as in Definition 5, one can construct a datalog program* $\mathsf{dlg}(\alpha)$ *that emulates α.*

Proof. If $\alpha = C \sqsubseteq D$ is a TBox axiom, define $\mathsf{datalog}(\alpha) := \mathsf{dlg}_H^{\mathcal{A}}(A \sqsubseteq \mathsf{NNF}(\neg C \sqcup D)) \cup \{A(x)\}$. If $\alpha = C(a)$ is an ABox axiom, define $\mathsf{datalog}(\alpha) := \mathsf{dlg}_a^{\mathcal{A}}(C(a), \bot)$. The result follows by Lemma 3 and 4. □

C	$\mathsf{dlg}_a^{\mathcal{A}}(C(a), E)$
$D \in \mathbf{L}_H^{\mathcal{A}}$	$\mathsf{dlg}_H^{\mathcal{A}}(X \sqsubseteq D \sqcup E) \cup \{X(a)\}$
$D_1 \sqcap D_2$	$\mathsf{dlg}_a^{\mathcal{A}}(D_1(a), E) \cup \mathsf{dlg}_a^{\mathcal{A}}(D_2(a), E)$
$D_1 \sqcup D_2 \in (\mathbf{L}_a^{\mathcal{A}} \sqcup \mathbf{L}_B^{\mathcal{A}})$	$\mathsf{dlg}_B^{\mathcal{A}}(\neg X \sqsubseteq D_2) \cup \mathsf{dlg}_a^{\mathcal{A}}(D_1(a), E \sqcup \neg X)$
$\exists R.D$	$\mathsf{dlg}_B^{\mathcal{A}}(\neg X \sqsubseteq E) \cup \mathsf{dlg}_a^{\mathcal{A}}(D(b), \neg Y) \cup \{X(a) \to R(a,b), X(a) \to Y(b)\}$
$E \in \mathbf{L}_B^{\mathcal{A}}$, X, Y fresh concept names, b a fresh constant	

Fig. 3. Transforming axioms $C(a)$ with $C \in \mathbf{L}_a^{\mathcal{A}}$ to datalog

We still need to show that $\mathcal{DLP}_{\mathcal{ALC}}$ is indeed the largest DLP fragment of $\mathcal{ALC}$. We first introduce two transformations – etb and qe –, and make some basic observations that allow us to use these transformations for showing maximality of $\mathcal{DLP}_{\mathcal{ALC}}$.

Definition 6. *Let C be an arbitrary $\mathcal{ALC}$ concept expression. The expression etb(C) (***e**liminate **t**op and **b**ottom*) is obtained from C by elimination of top and bottom symbols, achieved by applying exhaustively the following rewrite rules:*

$$\begin{array}{lllll} \top \sqcap D \mapsto D & \bot \sqcup D \mapsto D & \top \sqcup D \mapsto \top & \bot \sqcap D \mapsto \bot & \forall \mathbf{R}.\top \mapsto \top \\ D \sqcap \top \mapsto D & D \sqcup \bot \mapsto D & D \sqcup \top \mapsto \top & D \sqcap \bot \mapsto \bot & \exists \mathbf{R}.\bot \mapsto \bot \end{array}$$

Note, that etb(C) may still contain subexpressions of the form $\forall \mathbf{R}.\bot$ and $\exists \mathbf{R}.\top$

The next lemma summarises some easy observations on etb.

Lemma 5. *For any $\mathcal{ALC}$ concept expression C*

1. *etb(C) is logically equivalent to C, i.e., for any interpretation $\langle \Delta^{\mathcal{I}}, \mathcal{I} \rangle$ and any $a \in \Delta^{\mathcal{I}}$, we have $a \in C^{\mathcal{I}}$ iff $a \in \mathit{etb}(C)^{\mathcal{I}}$.*
2. *$C \in \mathbf{L}_\top^{\mathcal{A}}$ iff $\mathit{etb}(C) \in \mathbf{L}_\top^{\mathcal{A}}$ $\quad C \in \mathbf{L}_\bot^{\mathcal{A}}$ iff $\mathit{etb}(C) \in \mathbf{L}_\bot^{\mathcal{A}}$ $\quad C \in \mathbf{L}_a^{\mathcal{A}}$ iff $\mathit{etb}(C) \in \mathbf{L}_a^{\mathcal{A}}$*
   $C \in \mathbf{L}_B^{\mathcal{A}}$ iff $\mathit{etb}(C) \in \mathbf{L}_B^{\mathcal{A}}$ $\quad C \in \mathbf{L}_H^{\mathcal{A}}$ iff $\mathit{etb}(C) \in \mathbf{L}_H^{\mathcal{A}}$
3. *If C does not contain subexpressions of the form $\forall \mathbf{R}.\bot$ or $\exists \mathbf{R}.\top$ then $\mathit{etb}(C) = \bot$, or $\mathit{etb}(C) = \top$, or $\mathit{etb}(C)$ does neither contain $\bot$ nor $\top$.*

Definition 7. *Let C be an arbitrary $\mathcal{ALC}$ concept expression. The expression qe(C) is obtained from C by* quantifier elimination*, i.e., every subexpression of C of the form $\forall R.D$ or $\exists R.D$ is simply replaced by D.*

Lemma 6. *Let $\langle \mathbf{I}, \mathbf{A}, \mathbf{R} \rangle$ be a signature and fix a domain Δ. There is an interpretation $\mathcal{I}_1$ on Δ of the role symbols in $\mathbf{R}$ such that for any interpretation $\mathcal{I}_0$ on Δ of the signature $(\mathbf{I}, \mathbf{A}, \emptyset)$, and for any concept C of $\langle \mathbf{I}, \mathbf{A}, \mathbf{R} \rangle$, we find $C^{\mathcal{I}} = \mathit{qe}(C)^{\mathcal{I}_0}$ with $\mathcal{I} = \mathcal{I}_0 \cup \mathcal{I}_1$.*

Proof. Setting $\mathcal{I}_1(R) = \{\langle a, a \rangle \mid a \in \Delta\}$ for all $R \in \mathbf{R}$, we obtain:

$(\forall R.D)^{\mathcal{I}} = \{a \in \Delta \mid b \in D^{\mathcal{I}_0} \text{ for all } \langle a, b \rangle \in R^{\mathcal{I}_1}\} = \{a \in \Delta \mid a \in D^{\mathcal{I}_0}\} = D^{\mathcal{I}_0}$,
$(\exists R.D)^{\mathcal{I}} = \{a \in \Delta \mid \text{there is } \langle a, b \rangle \in R^{\mathcal{I}_1} \text{ with } b \in D^{\mathcal{I}_0}\} = \{a \in \Delta \mid a \in D^{\mathcal{I}_0}\} = D^{\mathcal{I}_0}$. □

Note, that Lemma 6 is true for arbitrary $\mathcal{ALC}$ concept expressions, they need neither belong to $\mathcal{DLP}_{\mathcal{ALC}}$ nor be name-separated.

Lemma 7. *Let C be an arbitrary $\mathcal{ALC}$ concept expression. Then*

$$\begin{array}{lll} C \in \mathbf{L}_B^{\mathcal{A}} & \textit{iff} & qe(C) \in \mathbf{L}_B^{\mathcal{A}}, \\ C \in \mathbf{L}_H^{\mathcal{A}} & \textit{iff} & qe(C) \in \mathbf{L}_H^{\mathcal{A}}, \\ C \in \mathbf{L}_a^{\mathcal{A}} & \textit{iff} & qe(C) \in \mathbf{L}_a^{\mathcal{A}}. \end{array}$$

Proof. Here is a sample from the inductive proof for the first equivalence. The goal in this case is to show that $(\forall R.D) \in \mathbf{L}_B^{\mathcal{A}}$ iff $D \in \mathbf{L}_B^{\mathcal{A}}$.

The "if" direction is directly covered by a grammar rule. For the "only if" direction, we observe that there are only two grammar rules that can produce a formula of the form $(\forall R.D)$. The first is $\forall \mathbf{R}.\mathbf{L}_B^{\mathcal{A}}$, for which we directly find that $(\forall R.D) \in \mathbf{L}_B^{\mathcal{A}}$ implies $D \in \mathbf{L}_B^{\mathcal{A}}$. The second rule is $\forall R.\mathbf{L}_\top^{\mathcal{A}}$. Thus $(\forall R.D) \in \mathbf{L}_B^{\mathcal{A}}$ implies $D \in \mathbf{L}_\top^{\mathcal{A}}$, which suffices since $\mathbf{L}_\top^{\mathcal{A}} \subseteq \mathbf{L}_B^{\mathcal{A}}$. □

Theorem 2. *$\mathcal{DLP}_{\mathcal{ALC}}$ is the largest DLP fragment of $\mathcal{ALC}$.*

Proof. For a contradiction, suppose that there is a DLP fragment $\mathcal{F}$ of $\mathcal{ALC}$ that is strictly larger than $\mathcal{DLP}_{\mathcal{ALC}}$. Then there is some GCI $C' \sqsubseteq D'$ in $\mathcal{F}$ but not in $\mathcal{DLP}_{\mathcal{ALC}}$. The other possibility that there is an ABox axiom $C'(a) \in \mathcal{F}$ with $C'(a) \notin \mathbf{L}_a^{\mathcal{A}}$ is completely analogous. By Definition 4, any name-separated variant $C \sqsubseteq D$ of $C' \sqsubseteq D'$ is still in $\mathcal{F}$. Since $\mathcal{DLP}_{\mathcal{ALC}}$ is closed under variants, $C \sqsubseteq D$ is not in $\mathcal{DLP}_{\mathcal{ALC}}$. By Definition 5 this means that the negation normal form E of $\neg C \sqcup D$ is not in $\mathbf{L}_H^{\mathcal{A}}$. By Lemmas 5 and 7 also etb(qe(E)) is not in $\mathbf{L}_H^{\mathcal{A}}$. Let E^{cnf} be a conjunctive normal form of etb(qe(E)). Thus $E^{cnf} = Con_1 \sqcap \ldots \sqcap Con_k$ with $Con_i = L_{i,1} \sqcup \ldots \sqcup L_{i,n_i}$ where each $L_{i,j}$ is a concept name or the negation of a concept name. Again, it can be verified that $E \in \mathbf{L}_H^{\mathcal{A}}$ iff $E^{cnf} \in \mathbf{L}_H^{\mathcal{A}}$. Furthermore, for one i, $1 \leq i \leq k$ there are two unnegated concept names among $\{L_{i,1}, \ldots, L_{i,n_i}\}$. Otherwise, we could show $E^{cnf} \in \mathbf{L}_H^{\mathcal{A}}$. For this we need the extended grammar of $\mathbf{L}_H^{\mathcal{A}}$. Without loss of generality let $i = 1$ and $L_{1,1} = A_1$, $L_{1,2} = A_2$ positive. The name separation of E may have been lost by building the transformation to conjunctive normal form E^{cnf}, but we still have the following:

1. For any atom A, if A occurs in E^{cnf} then $\neg A$ does not occur in E^{cnf}, and vice versa.
2. For any two different conjuncts Con_i and Con_j of E^{cnf}, there is a literal l occurring in Con_i and not in Con_j (and by symmetry also a literal l' occurring in Con_j and not in Con_i).

Claim 1 can be easily seen since the transformation from E to E^{cnf} is effected by repeated application of the rewriting rule $(C_1 \sqcap C_2) \sqcup C_3 \mapsto (C_1 \sqcup C_3) \sqcap (C_2 \sqcup C_3)$.
Claim 2 can be proven by induction on the structural complexity of E. In the simplest case E already is a conjunctive normal from. Then name separation of E even implies that different conjuncts Con_i are disjoint. Next assume that $E = E_1 \sqcup \ldots \sqcup E_n$ and by induction hypothesis each E_i has a conjunctive normal form $E_i = Con_{i,1} \sqcap \ldots \sqcap Con_{i,m_i}$, such that for $j \neq k$ the conjunct $Con_{i,j}$ contains a literal, that does not occur in $Con_{i,k}$. Furthermore, name separation of E tells us that different E_{i_1}, E_{i_2} do not share a literal. By elementary computation we have

$$E^{cnf} = \bigsqcap_{1 \leq i_1 \leq m_1} \ldots \bigsqcap_{1 \leq i_n \leq m_n} (Con_{1,i_1} \sqcup \ldots \sqcup Con_{n,i_n})$$

Lets look at two different conjuncts in E^{cnf}. Typically we may consider $Con_{1,1} \sqcup C_r$ and $Con_{1,2} \sqcup C_r$ with $C_r = Con_{2,i_2} \sqcup \ldots \sqcup Con_{n,i_n}$. By induction hypothesis there is a literal l in $Con_{1,1}$ that does not occur in $Con_{1,2}$. Under the present assumptions l occurs in $Con_{1,1} \sqcup C_r$ and not in $Con_{1,2} \sqcup C_r$. This completes our proof of claim 2. Returning to our main line of reasoning we define interpretations $\mathcal{I}_1$ and $\mathcal{I}_2$ on a universe Δ by

$$A_1^{\mathcal{I}_1} = \Delta \qquad L_{1,j}^{\mathcal{I}_1} = \emptyset \text{ for all } 2 \leq j \leq n_1$$
$$A_2^{\mathcal{I}_2} = \Delta \qquad L_{1,j}^{\mathcal{I}_2} = \emptyset \text{ for all } 1 \leq j \leq n_1, j \neq 2$$

Thus

$$Con_1^{\mathcal{I}_1} = Con_1^{\mathcal{I}_2} = \Delta \quad \text{and} \quad Con_1^{\mathcal{I}_1 \cap \mathcal{I}_2} = \emptyset$$

By property 2 it is possible to extend the interpretations $\mathcal{I}_i$ such that $Con_j^{\mathcal{I}_i} = \Delta$ for $i \in \{i, 1\}$ and $2 \leq j \leq k$. In total we have

$$(E^{cnf})^{\mathcal{I}_1} = (E^{cnf})^{\mathcal{I}_2} = \Delta \quad \text{and} \quad (E^{cnf})^{\mathcal{I}_1 \cap \mathcal{I}_2} = \emptyset$$

Since the normal form and the etb transformation preserve logical equivalence, we also have $\text{qe}(E)^{\mathcal{I}_i} = \Delta$ for $i = 1, 2$ and $\text{qe}(E)^{\mathcal{I}_1 \cap \mathcal{I}_2} = \emptyset$. By Lemma 6 there are expansions $\mathcal{I}_i^*$ of $\mathcal{I}_i$ such that $E^{\mathcal{I}_i^*} = \text{qe}(E)^{\mathcal{I}_i} = \Delta$ for $i \in \{1, 2\}$ and $E^{\mathcal{I}_1{}^* \cap \mathcal{I}_2{}^*} = E^{(\mathcal{I}_1 \cap \mathcal{I}_2)^*} = \text{qe}(E)^{\mathcal{I}_1 \cap \mathcal{I}_2} = \emptyset$. By Lemma 1, this contradicts the possibility that $\pi(E)$ can be emulated by a datalog formula. □

5 The Datalog Fragment of $\mathcal{SROIQ}$

The previous section showed that syntactic descriptions tend to become rather complex when maximising languages in a canonical way, but the situation is substantially more intricate when considering $\mathcal{SROIQ}^{\text{free}}$ instead of $\mathcal{ALC}$ as an underlying DL. Here, we summarise the conclusions that have been obtained in [19] for this case. There, a maximal DLP fragment of $\mathcal{SROIQ}^{\text{free}}$ has been developed under the additional requirement of closure under disjunctive normal forms (DNF):

Theorem 3. *The largest DL fragment of $\mathcal{SROIQ}^{\text{free}}$ that is also closed under DNF exists, and it can be characterised by a parametrised set of grammar productions. We call this DL $\mathcal{DLP}$.*

Disjunctive normal forms here are mainly required to curtail the syntactic complexity of the obtained fragment, and we conjecture that a maximal DLP fragment of $\mathcal{SROIQ}^{\text{free}}$ that does not have this property also exists. Rather than in the concrete description of this fragment, we are interested here in the general insights that are obtained from proofs of such results. The above result consists of three parts: (1) specifying an explicit syntactic characterisation, (2) showing that all $\mathcal{DLP}$ axioms can be **FOL**$_=$-emulated in datalog, (3) showing that $\mathcal{DLP}$ is the largest such DL. Here we give an overview of the main methods that are used in each step.

Syntactic Characterisation The main challenge here is to reduce the presentational complexity as far as possible. A *DLP normal form* is introduced that incorporates DNF

and an improved form of NNF, and which ignores concepts that, like $\mathbf{L}_{\top}^{\mathcal{A}}/\mathbf{L}_{\bot}^{\mathcal{A}}$ above, are always equivalent to $\top/\bot$. The syntax of $\mathcal{DLP}$ in normal form is still very complex due to the interplay of number restrictions and nominals that is possible even in name-separated axioms.

Datalog Emulation A recursive datalog transformation as in the case of $\mathcal{DLP}_{\mathcal{ALC}}$ above is provided. The individual steps are substantially more involved, and even lead to exponentially large datalog programs in various cases, although these programs are very regular and can be constructed in a single pass without complex computations. We conjecture that this blow-up is unavoidable but this issue has not been investigated further.

Maximality The least model property of datalog was used for showing maximality of $\mathcal{DLP}_{\mathcal{ALC}}$, but no extension of this direct approach to $\mathcal{DLP}$ has been found. Instead, additional model-theoretic properties of datalog were used that incorporate submodels and product models [6]. Using various inductive arguments, it has then been shown that any extension of $\mathcal{DLP}$ leads to axioms that cannot be $\mathbf{FOL}_{=}$-emulated in datalog.

We provide some examples to illustrate the issues that occur in the general case (datalog emulations are provided in parentheses). DLP expressions of the form $A \sqcap \exists R.B \sqsubseteq \forall S.C$ $(A(x) \wedge R(x,y) \wedge B(y) \wedge S(x,z) \rightarrow C(z))$ are well-known. The same is true for $A \sqsubseteq \exists R.\{c\}$ $(A(x) \rightarrow R(x,c))$ but hardly for $A \sqsubseteq \geqslant 2\, R.(\{c\} \sqcup \{d\})$ $(A(x) \rightarrow R(x,c)$, $A(x) \rightarrow R(x,d)$, $A(x) \wedge c \approx d \rightarrow \bot)$. Another unusual form of DLP axioms arises when Skolem constants (not functions) can be used as in the case $\{c\} \sqsubseteq \geqslant 2\, R.A$ $(R(c,s)$, $R(c,s')$, $A(s)$, $A(s')$, $s \approx s' \rightarrow \bot$ with fresh s, s') and $A \sqsubseteq \exists R.(\{c\} \sqcap \exists S.\top)$ $(A(x) \rightarrow R(x,c)$, $A(x) \rightarrow S(c,s)$ with fresh s). This is possible since semantic emulation is more general than semantic equivalence.

For a more complex $\mathcal{DLP}$ axiom, consider the GCI $\{c\} \sqsubseteq \geqslant 2\, R.(\neg\{a\} \sqcup A \sqcup B)$. It is semantically emulated by $\{R(c,s_1), R(c,s_2), a \approx s_1 \rightarrow A(s_1), a \approx s_2 \rightarrow A(s_2)\}$ where s_i are fresh constants. Note how equalities of fresh constants are used to simulate finite amounts of disjunctive behaviour. In contrast, $\{c\} \sqsubseteq \geqslant 2\, R.(\neg\{a\} \sqcup A \sqcup B \sqcup C)$ is not in $\mathcal{DLP}$.

Another complex example is $\{c\} \sqsubseteq \geqslant 4\, R.(A \sqcup \{a\} \sqcup (\{b\} \sqcap \leqslant 1\, S.(\{c\} \sqcup \{d\})))$ which is semantically emulated by a datalog program that contains about 30 rules. Interestingly, the axiom $\{c\} \sqsubseteq \geqslant 3\, R.(A \sqcup \{a\} \sqcup (\{b\} \sqcap \leqslant 1\, S.(\{c\} \sqcup \{d\})))$ which only differs by using 3 instead of 4 cannot be $\mathbf{FOL}_{=}$-emulated by any datalog program.

6 Conclusions and Outlook

DLP provides an interesting example of a general type of problem: given two KR formalisms that can be translated to first-order logic, how can we syntactically characterise all theories of the source formalism that can faithfully be represented in the target formalism? In this work, we proposed to interpret "faithful representation" by means of semantic emulation (a weaker notion of semantic equivalence), while "syntactic" has been realised by requiring closure under variants (non-uniform renamings of signature symbols). These two simple principles allowed us to show the existence of a largest DLP fragment for the DL $\mathcal{ALC}$. In this sense, we argue that our approach introduces a workable definition for the vague notion of the "intersection" of two KR formalisms.

Our rigorous definition of DLP fragments also clarifies the differences between DLP and the DLs $\mathcal{EL}$ and Horn-$\mathcal{SHIQ}$ which can both be expressed in terms of datalog as well. Neither $\mathcal{EL}$ nor Horn-$\mathcal{SHIQ}$ can be semantically emulated in datalog but both satisfy a weaker version of syntactic emulation that is obtained by restricting to variable-free formulae φ in Definition 2. Under such weaker requirements, a larger space of possible DL fragments is allowed, but it is unknown whether (finitely many) maximal languages exist in this case. There is clearly no largest such language, since both $\mathcal{EL}$ and $\mathcal{DLP}$ abide by the weakened principles whereas their (intractable) union does not (contradicting Proposition 2).

Even when weakening the requirements of DLP fragments like this, Horn-$\mathcal{SHIQ}$ is still excluded by Proposition 2, which explains why Horn-$\mathcal{SHIQ}$ cannot be translated to datalog axiom-by-axiom. In the presence of transitivity, Horn-$\mathcal{SHIQ}$ also is not really closed under variants, but this problem could be overcome by using distinct signature sets for simple and non-simple roles. Again, it is open which results can be established for Horn-$\mathcal{SHIQ}$-like DLs based on the remaining weakened principles.

This work also explicitly introduces a notion of *emulation* which appears to be novel, though loosely related to conservative extensions. In essence, it requires that a theory can take the place of another theory in all logical contexts, based on a given syntactic interface. Examples given in this paper illustrate that this can be very different from semantic equivalence. Yet, emulation can be argued to define minimal requirements for preserving a theory's semantics even in combination with additional information, so it appears to be a natural tool for enabling information exchange in distributed knowledge systems. We think that the articulation of this notion is useful for studying the semantic interplay of heterogeneous logical formalisms in general.

Finally, the approach of this paper – seeking a logical fragment that is provably maximal under certain conditions – immediately leads to a number of further research questions. For example, what is the maximal fragment of SWRL ("datalog $\cup$ $\mathcal{SROIQ}$," see [14]) that can be expressed in $\mathcal{SROIQ}$? Clearly, this fragment would contain DL Rules [21] and maybe some form of DL-safe rules [26]. But also the maximal $\mathbf{FOL}_{=}$ fragment that can be expressed in a well-known subset such as the Guarded Fragment [1] or the two-variable fragment might be of general interest. We argue that ultimate answers to such questions can indeed be obtained based on similar definitions of *fragments* as used for DLP in this work. At the same time, our study of $\mathcal{SROIQ}$ indicates that the required definitions and arguments can become surprisingly complex when dealing with a syntactically rich formalism like description logic. The main reason for this is that constructs that are usually considered "syntactic sugar" have non-trivial semantic effects when considering logical fragments that are closed under variants.

References

1. Andréka, H., van Benthem, J.F.A.K., Németi, I.: Modal languages and bounded fragments of predicate logic. Journal of Philosophical Logic 27(3), 217–274 (1998)
2. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P. (eds.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press, second edn. (2007)

3. Baget, J.F., Leclere, M., Mugnier, M.L.: Walking the decidability line for rules with existential variables. In: Lin, F., Sattler, U., Truszczynski, M. (eds.) Proc. 12th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'10). pp. 466–476. AAAI Press (2010)
4. Baral, C., Gelfond, M.: Logic programming and knowledge representation. J. Log. Program. 19/20, 73–148 (1994)
5. Calì, A., Gottlob, G., Lukasiewicz, T.: Datalog$^{\pm}$: a unified approach to ontologies and integrity constraints. In: Fagin, R. (ed.) Proc. 12th Int. Conf. on Database Theory (ICDT'09). pp. 14–30. ACM (2009)
6. Chang, C., Keisler, H.J.: Model Theory, Studies in Logic and the Foundations of Mathematics, vol. 73. North Holland, 3rd edn. (1990)
7. Dantsin, E., Eiter, T., Gottlob, G., Voronkov, A.: Complexity and expressive power of logic programming. ACM Computing Surveys 33(3), 374–425 (2001)
8. Donini, F.M., Lenzerini, M., Nardi, D., Schaerf, A.: $\mathcal{AL}$-log: Integrating datalog and description logics. Journal of Intelligent and Cooperative Information Systems 10(3), 227–252 (1998)
9. Eiter, T., Ianni, G., Schindlauer, R., Tompits, H.: A uniform integration of higher-order reasoning and external evaluations in answer-set programming. In: Kaelbling and Saffiotti [16], pp. 90–96
10. Eiter, T., Lukasiewicz, T., Schindlauer, R., Tompits, H.: Combining answer set programming with description logics for the Semantic Web. In: Dubois, D., Welty, C.A., Williams, M.A. (eds.) Proc. 9th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'04). pp. 141–151. AAAI Press (2004)
11. Gasse, F., Sattler, U., Haarslev, V.: Rewriting rules into $\mathcal{SROIQ}$ axioms. In: Baader, F., Lutz, C., Motik, B. (eds.) Proc. 21st Int. Workshop on Description Logics (DL'08). CEUR Workshop Proceedings, vol. 353. CEUR-WS.org (2008)
12. Grosof, B.N., Horrocks, I., Volz, R., Decker, S.: Description logic programs: combining logic programs with description logic. In: Proc. 12th Int. Conf. on World Wide Web (WWW'03). pp. 48–57. ACM (2003)
13. Horrocks, I., Patel-Schneider, P.F.: A proposal for an OWL rules language. In: Feldman, S.I., Uretsky, M., Najork, M., Wills, C.E. (eds.) Proc. 13th Int. Conf. on World Wide Web (WWW'04). pp. 723–731. ACM (2004)
14. Horrocks, I., Patel-Schneider, P.F., Bechhofer, S., Tsarkov, D.: OWL Rules: A proposal and prototype implementation. Journal of Web Semantics 3(1), 23–40 (2005)
15. Hustadt, U., Motik, B., Sattler, U.: Data complexity of reasoning in very expressive description logics. In: Kaelbling and Saffiotti [16], pp. 466–471
16. Kaelbling, L., Saffiotti, A. (eds.): Proc. 19th Int. Joint Conf. on Artificial Intelligence (IJCAI'05). Professional Book Center (2005)
17. Kazakov, Y.: Saturation-Based Decision Procedures for Extensions of the Guarded Fragment. Ph.D. thesis, Universität des Saarlandes, Saarbrücken, Germany (2006)
18. Krötzsch, M.: Efficient inferencing for OWL EL. In: Proc. 12th European Conf. on Logics in Artificial Intelligence (JELIA'10). LNAI, Springer (2010), to appear
19. Krötzsch, M., Rudolph, S.: Finding the largest datalog fragment of description logic. Tech. Rep. 3002, Institute AIFB, Karlsruhe Institute of Technology (2009), available online at `http://www.aifb.kit.edu/web/Techreport3002`
20. Krötzsch, M., Rudolph, S., Hitzler, P.: Complexity boundaries for Horn description logics. In: Proc. 22nd AAAI Conf. on Artificial Intelligence (AAAI'07). pp. 452–457. AAAI Press (2007)
21. Krötzsch, M., Rudolph, S., Hitzler, P.: Description logic rules. In: Ghallab, M., Spyropoulos, C.D., Fakotakis, N., Avouris, N. (eds.) Proc. 18th European Conf. on Artificial Intelligence (ECAI'08). pp. 80–84. IOS Press (2008)

22. Krötzsch, M., Rudolph, S., Hitzler, P.: ELP: Tractable rules for OWL 2. In: Sheth, A., Staab, S., Dean, M., Paolucci, M., Maynard, D., Finin, T., Thirunarayan, K. (eds.) Proc. 7th Int. Semantic Web Conf. (ISWC'08). LNCS, vol. 5318, pp. 649–664. Springer (2008)
23. Levy, A.Y., Rousset, M.C.: Combining Horn rules and description logics in CARIN. Artificial Intelligence 104(1–2), 165–209 (1998)
24. Motik, B., Horrocks, I., Rosati, R., Sattler, U.: Can OWL and logic programming live together happily ever after? In: Cruz, I.F., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) Proc. 5th Int. Semantic Web Conf. (ISWC'06). LNCS, vol. 4273, pp. 501–514. Springer (2006)
25. Motik, B., Rosati, R.: A faithful integration of description logics with logic programming. In: Veloso, M.M. (ed.) Proc. 20th Int. Joint Conf. on Artificial Intelligence (IJCAI'07). pp. 477–482. IJCAI (2007)
26. Motik, B., Sattler, U., Studer, R.: Query answering for OWL DL with rules. Journal of Web Semantics 3(1), 41–60 (2005)
27. Rosati, R.: $\mathcal{DL}$+*log*: A tight integration of description logics and disjunctive datalog. In: Doherty, P., Mylopoulos, J., Welty, C.A. (eds.) Proc. 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'06). pp. 68–78. AAAI Press (2006)
28. Volz, R.: Web Ontology Reasoning with Logic Databases. Ph.D. thesis, Universität Karlsruhe (TH), Germany (2004)