Nominal Schemas for Integrating Rules and Ontologies

Technical Report, Kno.e.sis Center, Wright State University, 2010

Frederick Maier, Adila A. Krisnadhi, Pascal Hitzler
Kno.e.sis Center, Wright State University, Dayton, OH 45435, U.S.A.
{fred,adila,pascal}@knoesis.org

This is a very early report on the topic. It has subsequently been revised and substantially extended, leading to the publication [15], available from http://korrekt.org/page/A_Better_Uncle_For_OWL

ABSTRACT
We propose a description-logic style extension of OWL DL, which includes DL-safe variable SWRL and seamlessly integrates datalog rules. Our language also sports a tractable fragment, which we call ELP 2, covering OWL EL, OWL RL, most of OWL QL, and variable restricted datalog.

Categories and Subject Descriptors
I.2.4 [Knowledge Representation Formalisms and Methods]: Representation languages; F.4.1 [Mathematical Logic]: Computational logic

General Terms
Languages, Complexity, Algorithms

Keywords
Web Ontology Language, Description Logic, SROIQ, Semantic Web Rule Language, Datalog, tractability

1. INTRODUCTION
Despite significant recent progress, the quest for a satisfactory integration of Ontologies and Rules for the Semantic Web is still ongoing [7]. In the aftermath of the 2004 W3C Recommendation for the Web Ontology Language OWL [17], the problem was discussed, in a nutshell, by referring to the uncle rule, expressible in datalog as

\[ \text{brotherOf}(x, y) \land \text{parentOf}(y, z) \rightarrow \text{uncleOf}(x, z), \]

which could not be modeled at all in that version of OWL. From the perspective of OWL design criteria, a core difficulty in allowing unrestricted datalog rules along with OWL axioms is that it leads to undecidability of reasoning in the resulting combined language.

Subsequently, a significant body of work has developed, investigating the integration of description logics (DLs) [1], which form the basis for OWL, and rule languages (typically datalog). Conceptually, one can distinguish two approaches. On the one hand, description logics have been extended with additional “description-logic-style” expressive features which make it possible to express certain types of rules. OWL 2 [21], the 2009 revision of the OWL W3C Recommendation, in fact indirectly incorporates datalog rules of a certain form. This covers the uncle rule mentioned above, and, more generally, many rules with a tree-shaped body [13]. Decidability is nevertheless retained. However, many rules, such as

\[ \text{parentOf}(x, z) \land \text{parentOf}(y, z) \land \text{married}(x, y) \rightarrow C(z), \]

which defines a class \( C \) of children whose parents are married, are still not expressible.

On the other hand, there are approaches that are of a hybrid nature, in the sense that both OWL axioms and rules are syntactically allowed in ontologies, and a combined formal semantics defines how the hybrid language is to be understood. As already mentioned, such a combination generally leads to undecidability.\(^1\) This is the case for the Semantic Web Rule Language SWRL, [9, 8] which is the most straightforward rule extension of OWL, and which implicitly covers all of first-order predicate logic.\(^2\) In order to retain decidability, the most prominently discussed idea\(^3\) is to restrict the applicability of rules to known individuals in the knowledge base, i.e. to explicitly mentioned logical constants. Rules that are understood in this sense are called DL-safe [8, 20], and the combination of OWL DL and DL-safe SWRL is indeed decidable.\(^4\)

\(^1\)In the case of nonmonotonic semantics for rules, this may lead to languages which are not even semi-decidable, because some of these rule languages are not semi-decidable. We will not deal with nonmonotonic semantics very much in this paper, but we will briefly discuss them in Section 5.

\(^2\)The combination of RIF rule bases and OWL DL ontologies is also undecidable [4], even for RIF Core [2].

\(^3\)Other approaches, such as [5], achieve decidability by limiting the semantic interaction between the DL part and the rules part of the language, i.e., they provide a loose integration. While this is useful in some contexts, it seems evident that a tight integration would be preferred in general. A conceptually different series of work based on the idea of extending datalog with existential quantifiers uses syntactic restrictions to achieve decidability and favorable computational complexities [3].

\(^4\)Hybrid MKNF [10, 19] is also essentially based on the same idea, although the semantics is spelled out differently.
A generalization of DL-safe rules has been introduced in [14].\(^5\) in the form of *DL-safe variables* as part of the definition of the tractable ELP language. Rather than restricting all variables in a (DL-safe) rule such that they can bind only to known individuals, DL-safe variables endow the ontology engineer with the means to explicitly specify the variables to be treated this way.

In this paper, we expand on the above idea and improve on it in several ways. The key technical innovation is the introduction of *nominal schemas* as new elements of the DL syntax. While the semantic intuition behind nominal schemas is the same as that behind DL-safe variables, the key difference lies in the fact that DL-safe variables are tied to rule languages, while nominal schemas integrate seamlessly with DL syntax. As a consequence, the language which we propose encompasses DL-safe variable SWRL *while staying within the DL syntax paradigm*. It thus achieves both within the DL syntax paradigm what has hitherto only been achieved by hybrid approaches.

To give an initial example, consider again the rule (1) extended by the axioms (∃married.∃parentOf.{mary})(john) and parentOf(john, mary). First-order logic semantics (i.e., SWRL semantics) would yield, that Mary belongs to the class C. Interpreting the rule (1) as DL-safe, however, does not allow this conclusion. If we impose that the variable \(z\) in rule (1) is DL-safe while \(x\) and \(y\) are not, then the desired conclusion follows, but the rule (1) interpreted as such cannot be easily expressed in DL syntax,\(^6\) and so the knowledge base remains hybrid.

In contrast, using nominal schemas, rule (1) with \(z\) DL-safe can be expressed as

\[
\exists \text{parentOf} \{z\} \land \exists \text{married} \exists \text{parentOf} \{z\} \subseteq U, \{(z) \cap C\}, (2)
\]

where \(U\) is the universal role, and the desired conclusion again follows. \(\{z\}\) is a *nominal schema*, which is to be read as a *variable nominal* that can only bind to true nominals (i.e., \(z\) binds to known individuals), and the binding is the same for all occurrences of the nominal schema in the axiom.

The main contribution of this paper is the introduction of nominal schemas into DL syntax and the corresponding definition of a DL called \(\text{SROIQB}_s(\times, \{v\})\), which extends OWL DL and possesses the following very pleasing features:

1. It is a description logic (and thus can be defined concisely without reference to rules). (2) It is decidable (worst-case computational complexity of reasoning is in \(\text{N3ExpTime}\)).
2. It encompasses DL-safe SWRL and thus DL-safe datalog. (4) It sports a tractable (polynomial time) fragment ELP 2, which contains OWL EL (i.e. \(\text{EL}^{++}\)), OWL RL (i.e. DLP 2), most of OWL QL (i.e. DL-Lite\(s\); inverse roles are not completely covered), DL-safe datalog (and thus DL-safe RIF-Core), and 5-variable restricted (non DL-safe) datalog.

The work presented herein is heavily inspired by the work on ELP [14], which also explains why we call the tractable sublanguage ELP 2. We also incorporate the rather neat alternative definition of ELP given in [11] together with notational choices from [12]. The key difference between ELP 2 and ELP as presented in these publications is that ELP is defined as a *hybrid* language composed of DL axioms and rule axioms, and determining whether a knowledge base is in ELP can, in general, only be made by performing syntactic transformations and subsequently checking whether some standing conditions are met. In contrast, ELP 2 (and, in fact, \(\text{SROIQB}_s(\times, \{v\})\)) can be defined without resorting to rules or such transformations. Indeed, we urge the reader to verify, from the definition given in [14], that rule (1) (with \(z\) DL-safe) is not in ELP, which requires the checking of several rather involved conditions. In contrast, it follows easily from our definitions (given later) that axiom (2) is in ELP 2.

The paper is structured as follows. In Section 2 we introduce the syntax and semantics of \(\text{SROIQB}_s(\times, \{v\})\) and give a complexity analysis. We introduce ELP 2 in Section 3 and show its polynomiality and that it encompasses OWL EL. In Section 4 we clarify the relation of datalog and SWRL to \(\text{SROIQB}_s(\times, \{v\})\) and ELP 2, and that the latter contains OWL RL and most of OWL QL. Section 5 contains conclusions and a discussion of future research.

The paper is relatively self-contained, but due to the depth of the discussion, we assume some knowledge by the reader of different notations for logics around OWL and rules, as well as background on DLs and their relationships to OWL. This background material can be found, e.g., in [6].

## 2. NOMINAL SCHEMAS FOR OWL

In [23] it was shown that the description logic \(\text{SROIQ}\), the logic underlying OWL 2 DL, can be extended with boolean constructors (\(\land, \lor, \neg\)) on simple roles without thereby increasing the complexity of the standard reasoning tasks. The resulting logic was called \(\text{SROIQB}_s\). Concept products \(C \times D\) and a universal role \(U\) (which can in turn be expressed as \(\top \times \top\), where \(\top\) is the universal concept) were also briefly discussed: while not formally part of \(\text{SROIQB}_s\), expressions involving both can be reduced to those using only the basic role constructors. A logic (which we’ll call \(\text{SROIQB}_s(\times, \{v\})\)) involving all features was described in [11]. Again, including all of these features does not increase the worst-case complexity of reasoning.

Here, we extend \(\text{SROIQB}_s\) with nominal schemas, forming the logic \(\text{SROIQB}_s(\times, \{v\})\). We mainly follow and extend the presentation found in [11] and give a complete account of the syntax and semantics of \(\text{SROIQB}_s(\times, \{v\})\).

A signature \(\Sigma\) is a tuple \(\langle \text{Nt}, \text{NC}, \text{Nr}, \text{NV}\rangle\), where \(\text{Nt}, \text{NC}, \text{Nt},\) and \(\text{NV}\) are finite and pairwise disjoint sets of *individual names*, *concept names*, *role names*, and *variables*. The set of terms \(\text{T}\) of \(\Sigma\) is \(\text{Nt} \cup \text{NV}\). The set \(\text{Nt}\) is partitioned into disjoint sets \(\text{Nt}_s\) of *simple role names* and \(\text{Nt}_n\) of *non-simple role names*. We also specify a universal role \(\top\), not in \(\text{Nt}\).

For each \(a \in \text{Nt}_s\), \(a\) is a *nominal*, and for each \(v \in \text{NV}, \{v\}\) is a *nominal schema*. \(O\) denotes the set of nominals, and \(V\) denotes the set of nominal schemas.

From now on, we assume a given signature \(\Sigma\) and so omit further references to it.

\(^5\)See also [11].

\(^6\)Arguments like this are always difficult, since languages of the same computational complexity are always translatable into each other. As usual, we appeal to an intuitively “simple” notion of “expressibility” in this context.
A simple atomic role is any element of $N^0 \cup \{ R^- \mid R \in N^0 \}$. A non-simple atomic role is any element of $N^0 \cup \{ R^- \mid R \in N^0 \}$. The set $R_o$ of non-simple role expressions is the smallest set that contains every non-simple atomic role and every concept product $C \times D$, where $C, D$ are nominal-schema free elements of the set $C$, as defined in Definition 1. The set $R_r$ of simple role expressions is the smallest set that contains $U$, every simple atomic role, every expression of the form $\neg R, R \cap S$ and $R \cap S$ for $R, S \in R_o$, and every concept product. The set of role expressions is the union of $R_o$ and $R_r$. The set $R_{C \times D} \subseteq R$ is the set of all concept products.

We define Inv as a function from roles to roles, as follows: For each $R, S \in N_R$, define $Inv(R) = R^-, \hat{Inv}(R) = R$, $\hat{Inv}(R \cap S) = \hat{Inv}(R) \cap \hat{Inv}(S)$, $\hat{Inv}(R \cup S) = \hat{Inv}(R) \cup \hat{Inv}(S)$, $\hat{Inv}(\neg R) = \neg \hat{Inv}(R)$; $Inv(U) = U$; for each concept product $C \times D$, $Inv(C \times D) = D \times C$.

**Definition 1.** The set $C$ of $SROIQB_\times(\times, \{v\})$ concept descriptions (or, concepts) is the smallest set such that

1. $\{ (\top, \bot) \cup N_C \cup O \cup V \subseteq C$;
2. if $C, D \in C$ and $k$ is an nonnegative integer, then the following are also $SROIQB_\times(\times, \{v\})$ concepts: $\neg C$, $\forall R.C \sqsubseteq D$, $\exists R.C \sqsubseteq D$, $\forall R.C \sqsubseteq \forall R.C$, $\exists S.\text{Self}$, $\leq k.S.C$ and $\geq k.S.C$, where $R \in R$ and $S \in R$.

A concept description $C$ is nominal-schema free if no occurrence of any element of $V$ appears in $C$.

A generalized role inclusion axiom (RIA) is any statement of the form $S_1 \circ \cdots \circ S_k \subseteq R$ where $R$ and each $S_j$ are in $R$ and if $R \notin R_a$, then $k = 1$ and $S_1 \in R_a$. An RBox axiom is either an RIA, or a statement of the form $\text{Ref}(R)$ (reflexive role), $\text{Asy}(R)$ (asymmetric role), $\text{Dis}(R_1, R_2)$ (role disjointness), where $R_{(i)} \in R$. A TBox axiom (general concept inclusion axiom, GCI) is any expression $C \subseteq D$, where $C, D \in C$. A TBox axiom is nominal-schema free if no nominal schema occurs in it. An ABox axiom is any expression of the form $C(a), R(a, b), \neg S(a, b)$, and $a \neq b$, where $C$ is a nominal-schema free concept description in $C$, $R \in R$, $S \in R$ or $R^C \times D$, and $a, b \in N_I$. A knowledge base is a set of RBox, TBox, and ABox axioms. A knowledge base is nominal-schema free if all of its TBox axioms are so.

An RBox is regular if there is a strict partial order $\prec$ on $R$ such that

- if $R \notin \{S, \text{Inv}(S)\}$, then $S \prec R$ iff $\text{Inv}(S) \prec R$; and
- every RIA is of one of the forms $R \circ R \subseteq R$, $\text{Inv}(R) \subseteq R$, $R \circ S_1 \cdots S_k \subseteq R$, $S_1 \circ \cdots \circ S_k \subseteq R$, or $S_1 \circ \cdots \circ S_k \subseteq R$, where $R$ and each $S_i$ is in $R$, and $S_i \prec R$ for each $S_i$.

A knowledge base is a $SROIQB_\times(\times, \{v\})$ knowledge base iff its RBox is regular, and for each statement $\text{Irr}(S)$, $\text{Asy}(S)$, $\text{Dis}(S, T)$ in the RBox, $S, T \in N_I$.

Axiom (2) from the previous section is an example of a $SROIQB_\times(\times, \{v\})$ axiom. The $\{z\}$ appearing there is a nominal schema. Intuitively, each nominal schema appearing in an axiom is universally quantified, and in any interpretation, ranges only over named individuals in the domain of discourse. Importantly, the semantics described below assumes that axioms do not share nominal schemas; i.e., if a schema appears in one axiom, then it appears in no other. This simplifies the presentation somewhat.

An interpretation $I = (\Delta^I, \hat{\alpha})$ for a signature $\Sigma$ consists of a domain of discourse $\Delta^I \neq \emptyset$ and a function $\hat{\alpha}$ which maps $N_C, N_R, N_I, \top, \bot$ to elements, sets, and relations of $\Delta^I$ as shown in Table 1. A variable assignment $\hat{Z}$ for an interpretation $I$ is a function from $N_V$ to $\Delta^I$ such that for each $v \in N_V$, $Z(v) = a^\hat{\alpha}$ for some $a \in N_I$. For any interpretation $I$, assignment $\hat{Z}$, and $C(i) \in C$, $R(i) \in N_R$, $t(i) \in T$, the function $\alpha^{I,\hat{Z}}$ is defined as shown in Table 1. For an $\hat{A}$ a $SROIQB_\times(\times, \{v\})$ axiom, and $\hat{Z}$ a set of such axioms, $I$ and $\hat{Z}$ satisfy $\hat{A}$, written $I, \hat{Z} \models \hat{A}$, iff the corresponding condition shown in Table 1 holds. $I$ and $Z$ satisfy $S$ ($I, Z \models S$) iff $I, Z \models \hat{A}$ for each $\hat{A}$ in $S$. $I$ is a model of $S$ if ($I, Z \models S$) for each assignment $\hat{Z}$ for $I$.

The logic $SROIQB_\times$ is obtained from $SROIQB_\times(\times, \{v\})$ by disallowing nominal schemas; concept products are already covered by $SROIQB_\times$ since they can be simulated using role negations [23]. The logic $SROIQ$ is in turn obtained from $SROIQB_\times$ by disallowing Boolean role constructors. In Section 4, we show that $SROIQB_\times(\times, \{v\})$ is also expressive enough to encompass DL-safe datalog (and thus DL-safe SWRL and DL-safe RIF-Core).

We note that it is very straightforward to introduce nominal schemas into the normative RDF syntax for OWL 2 [22]. One way to do this would be to provide URIs for variables in the OWL namespace, used instead of individuals in owl:oneOf-statements (which are used for the RDF syntax for nominals in OWL 2).

### 2.1 Complexity of $SROIQB_\times(\times, \{v\})$

As shown here, every $SROIQB_\times(\times, \{v\})$ knowledge base can be converted into an equisatisfiable $SROIQB_\times$ knowledge base simply by grounding the nominal schemas—i.e., by uniformly replacing nominal schemas with elements of $O$. Since reasoning in $SROIQB_\times$ is NP2ExpTime-complete [11, 23], it follows that it is decidable in $SROIQB_\times(\times, \{v\})$.

Let $C$ be a concept containing $m \geq 1$ nominal schemas $\{v_1, \ldots, v_m\}$ and $\{a_1, \ldots, a_m\}$ be a tuple of $m$ individual names. A grounding of $C$ w.r.t. $\{a_1, \ldots, a_m\}$ (denoted using $C\theta$, where $\theta = [v_i/a_1, \ldots, v_m/a_m]$) is the concept that is obtained from $C$ by substituting all occurrences of nominal schema $\{v_i\}$ with nominal schema $\{a_i\}$ for every $i \in \{1, \ldots, m\}$. If $A$ is an axiom, then $A\theta$ is an axiom obtained from $A$ by an analogous substitution. If $A$ is free of nominal schemas, then $A = A\theta$. $A'$ is the set of all groundings of $A$ constructible from $N_I$. If $S$ is a set of axioms, then let $S'$ be $\bigcup\{A' \mid A \in S\}$. Observe that $S'$ is expressible in $SROIQB_\times$.

Grounding provides an alternative way of looking at nominal schemas, namely as a type of syntactic sugar or macro for the set of corresponding grounds. In general, however, there should be smarter algorithmizations than through a naive grounding, a matter to which we return in Section 3.
We show that there are the form $A = A \subseteq \Delta^2$ and $\Sigma = \Sigma \cap \Delta^2$, then $\text{Theorem}$.

Proof. Let $S$ be a $\text{SROIQB}(x, \{v\})$ knowledge base. We show $S$ is satisfiable if and only if $S'$ is.

(\text{LR}) Let $I$ be a model of $S$, and suppose $A \subseteq S'$. If $A \subseteq S$, then obviously $I$ is a model of $S$. So suppose $A \notin S$. Then $A$ must be of the form $(C \subseteq D)\theta$ where (i) $C \subseteq D \in S$; (ii) there are $m$ nominal schemas $\{v_1, \ldots, v_m\}$ in $C \subseteq D$, $m \geq 1$; (iii) $\theta = [a_1, \ldots, a_m]$ for some individual names $a_1, \ldots, a_m$ in $N_I$. Let $Z$ be an assignment for $I$ such that $Z(v_i) = a_i$ for each $i = 1, \ldots, m$. Then $(C\theta)^{1,2} = C^{1,2}$ and $(D\theta)^{1,2} = D^{1,2}$. Because $I$ is a model of $S$, $C^{1,2} \subseteq D^{1,2}$; so $(C\theta)^{1,2} \subseteq (D\theta)^{1,2}$, i.e., $I, Z \models A$. Since $C\theta$ and $D\theta$ are nominal-schema free, $(C\theta)^{1,2} = (C\theta)^{1,2}$ and $(D\theta)^{1,2} = (D\theta)^{1,2}$ for every assignment $W$ for $I$, and so $W, I \models A$ for every assignment $W$ for $I$, i.e., $I$ is a model of $A$. Generalizing on $A$, $I$ is a model of $S'$.

(\text{RL}) Let $I$ be a model of $S'$, and suppose $A \subseteq S$. If $A \subseteq S$, then $I$ models $A$. So suppose $A \notin S$. $A$ must have the form $C \subseteq D$ and must contain nominal schemas, $\{v_1, \ldots, v_m\}$ for some $m \geq 1$. Let $Z$ be an assignment for $I$. Then there exist individual names $a_1, \ldots, a_m$ in $N_I$ such that $Z(v_i) = a_i$ for each $i = 1, \ldots, m$. The assignment $Z$ guarantees that $C^{1,2} = (C\theta)^{1,2}$ and $D^{1,2} = (D\theta)^{1,2}$, where $\theta = [v_1, a_1, \ldots, v_m, a_m]$. $A\theta$ is a valid grounding of $A$ and so appears in $S'$. Since $I$ models $S'$, $I, Z \models A\theta$, and so $(C\theta)^{1,2} \subseteq (D\theta)^{1,2}$. I.e., $I, Z \models A$. Generalizing on $Z$ and $A$, we conclude that $I$ is a model of $S$. \hfill \Box

Theorem 1. $\text{SROIQB}(x, \{v\})$ is decidable.

The naive grounding of a $\text{SROIQB}(x, \{v\})$ knowledge base will yield an exponentially larger $\text{SROIQB}(x, \{v\})$ knowledge base that is free of nominal schemas if the number of occurrences of different nominal schemas in the axioms of the knowledge base is unbounded. However, if we have a fixed global bound of the number of occurrences of different nominal schemas in each axiom, i.e., by imposing such a restriction as a part of the language, then the naive grounding only yields a polynomially larger knowledge base. Given that reasoning in $\text{SROIQB}$ is $N2ExpTime$-complete, we obtain the following corollary.

Theorem 2. Reasoning for $\text{SROIQB}(x, \{v\})$ can be done in $N3ExpTime$, and it is $N2ExpTime$-hard. Moreover, let $\text{SROIQB}(x, \{v\})$ be the logic that is obtained from $\text{SROIQB}(x, \{v\})$ by requiring that every axiom in the knowledge base may only have at most $n$ different nominal schemas where $n$ is a fixed positive integer. Then reasoning in $\text{SROIQB}(x, \{v\})$ is $N2ExpTime$-complete.

3. A TRACTABLE FRAGMENT
The introduction of nominal schemas does not only give rise to an integration of OWL and rules. It also provides us with the possibility of defining a tractable fragment, i.e., a sublanguage having polynomial time complexity. It is rather pleasing that this tractable fragment completely contains two of the three designated tractable profiles of OWL 2 [18] as well as variable restricted datalog. The third tractable OWL 2 profile is also almost completely contained.

This fragment is defined here, and subsequent sections show the desired containment relationships. The definition is very heavily inspired by the work on ELP [11, 14]. The essential differences have already been discussed in Section 1.

If \( n \) is an arbitrary but fixed positive integer, then the logic \( \text{SROEL}(\pi, \times, \{V\}_n) \), defined below, is a tractable fragment of \( \text{SROIQB}(\pi, \times, \{V\}) \)—the integer restricts the number of occurrences of nominal schemas. Every \( \text{SROEL}(\pi, \times, \{V\}) \) signature is a \( \text{SROEL}(\pi, \times, \{V\}_n) \) signature, and the sets \( N_C, N_R, N_I \), etc., are defined as they were previously. Concepts and roles for \( \text{SROEL}(\pi, \times, \{V\}_n) \) are defined below.

**Definition 2.** The set \( C \) of \( \text{SROEL}(\pi, \times, \{V\}_n) \) concept descriptions is the smallest set such that

1. \( \{\top, \bot\} \cup N_C \cup O \cup V \subseteq C \),
2. if \( C, D \in C, R \in N_R \), then \( C \cap D, \exists R.C \) and \( \exists R.Self \) are in \( C \).

A \( \text{SROEL}(\pi, \times, \{V\}_n) \) RBox axiom is any expression having one of the forms:

1. \( R_1 \circ R_2 \subseteq R \), where \( R \) and each \( R_i \in N_R \).
2. \( R_1 \cap R_2 \subseteq R \), where \( R \) and each \( R_i \in N_R \).
3. \( C \times D \subseteq R \), where \( C, D \in C \) and \( R \in N_R \).
4. \( R \subseteq C \times D \), where \( C, D \in C \) and \( R \in N_R \).

A \( \text{SROEL}(\pi, \times, \{V\}_n) \) TBox axiom is any expression \( C \subseteq D \), where \( C, D \in C \). A \( \text{SROEL}(\pi, \times, \{V\}_n) \) ABox axiom is any expression \( C(a) \) or \( R(a, b) \), where \( C \in N_C, R \in N_R, a, b \in N_I \).

In order to arrive at a polynomial time fragment, we have to avoid the exponential blowup caused by grounding of nominal schema as in Theorem 2. The following notion allows us to do this in a non-trivial way.

**Definition 3.** An occurrence of a nominal schema \( \{x\} \in V \) in an axiom \( A \) is called safe, if this occurrence is within a concept description \( D \) of \( A \) such that

1. \( D \) contains no other occurrence of any nominal schema,
2. the occurrence of \( D \) is within a subformula of the form \( \{a\} \cap \exists R.D \), with \( \{a\} \) a nominal.

We write \( S(a, \exists R.D, x) \) to denote the subformula of the form \( \{a\} \cap \exists R.D \) which contains a safe occurrence of a nominal schema \( \{x\} \) as just described. We call \( S(a, \exists R.D, x) \) a safe environment for this occurrence of the nominal schema \( \{x\} \).

**Definition 4.** If \( KB \) is a knowledge base and \( R \) a role name, then \( \text{ran}(R) \) is the set of concept descriptions \( D \in C \) such that \( \{R_1 \subseteq R_2, \ldots, R_n \subseteq R_{n+1}\} \subseteq KB \), where \( n \geq 1 \), \( R_1 = R \), and \( R_{n+1} \subseteq C \times D \in KB \).

A \( \text{SROEL}(\pi, \times, \{V\}_n) \) knowledge base is a knowledge base which satisfies the following restrictions:

1. if \( \exists S.Self \) appears in \( KB \), then \( S \in N_B \);
2. if \( R_1 \cap R_2 \subseteq R \), then \( R_1, R_2 \in N_B \);
3. if \( R_1 \circ R_2 \subseteq S \) is in \( KB \), \( \text{ran}(S) \subseteq \text{ran}(R_2) \);
4. if \( R_1 \cap R_2 \subseteq S \) is in \( KB \), \( \text{ran}(S) \subseteq \text{ran}(R_1) \cup \text{ran}(R_2) \);
5. no nominal schema appears in any role expression;
6. in any axiom \( B \subseteq C \), there are at most \( n \) nominal schemas appearing more than once in non-safe form, and all remaining nominal schemas appear only in \( B \).

Item 6 merits some further explanations, as this condition is central to the expressive power of \( \text{SROEL}(\pi, \times, \{V\}_n) \) and also to the question why it is of polynomial time complexity. Item 6 in fact entails the following: There are at most \( n \) nominal schemas in \( B \subseteq C \) such that, after replacing all occurrences of these by nominals (through grounding), the resulting axiom \( B' \subseteq C \) contains nominal schemas only in \( B' \).

Furthermore, for each nominal schema in \( B' \), all occurrences, except possibly one occurrence, are safe.

Another perspective on item 6 can be given as follows. In order to check whether an axiom \( B \subseteq C \) satisfies item 6, do the following. (1) Erase all safe occurrences in \( B \) (e.g., by substituting an arbitrary atomic class name), resulting in an axiom \( B' \subseteq C \). (2) Subsequently, erase all nominal schemas in \( B' \) which occur only once, resulting in an axiom \( B'' \subseteq C \). (3) \( B \subseteq C \) satisfies item 6 if, and only if, \( B'' \subseteq C \) contains at most \( n \) nominal schemas (possibly occurring several times).

It is very easily verified that axiom (2) can be expressed in \( \text{SROEL}(\pi, \times, \{V\}_1) \). This also shows that the language \( \text{SROEL}(\pi, \times, \{V\}_1) \) is not contained in ELP.

**Definition 5.** \( \text{SROEL}(\pi, \times, \{V\}_3) \) knowledge bases are also called ELP 2 knowledge bases.

The choice for the number 5 as the global bound lies in the fact that OWL 2 RL can be embedded in the 5-variable fragment of datalog. We return to this topic in Section 4.2. Below, we show that ELP 2 contains OWL 2 EL.

**Theorem 3.** \( \text{SROEL}(\pi, \times) \) (and, thus, essentially OWL 2 EL) is a fragment of \( \text{SROEL}(\pi, \times, \{V\}_3) \).

**Proof.** Syntactically, every valid \( \text{SROEL}(\pi, \times) \) expression is a \( \text{SROEL}(\pi, \times, \{V\}_3) \) expression. This can be seen by comparing the presentation here with that found in [12]. Similarly, variable assignments become irrelevant, and so the semantics for \( \text{SROEL}(\pi, \times, \{V\}_3) \) collapses to that of \( \text{SROEL}(\pi, \times) \). \( \square \)

\(^7\)This requirement can be relaxed somewhat. However the definition then becomes more awkward, which is why we decided to stick with the presented one. We will further comment on this below.
3.1 A polynomial algorithm

We show here that $\mathcal{SROEL}(\forall, \times, \{v\}_n)$, for any fixed $n$, is indeed a tractable logic.

Let $K$ be a $\mathcal{SROEL}(\forall, \times, \{v\}_n)$ knowledge base. We will transform $K$ into a $\mathcal{SROEL}(\forall, \times)$ knowledge base $K'$. This will be done in three subsequent steps.

1. We know, that in any axiom $A \in K$, there are at most $n$ nominal schemas $\{x_1, \ldots, x_n\}$ which appear more than once in non-safe form. The first step of the transformation is obtained by inserting into $K'$ all of the groundings of each axiom $A \in K$, as described in Section 2.1—however, the groundings are done only with respect to the nominal schemas $\{x_1, \ldots, x_n\}$; all other nominal schemas are untouched. The number of new axioms introduced this way is bounded from above by $y^{n+1}$, where $y$ is the maximum of the number of axioms and the number of individuals in $K$. Hence, this grounding is polynomial.

We are now left with nominal schemas in axioms $B \subseteq H$ which occur only in $B$ and which occur at most once in non-safe form in $B$.

2. Introduce a new class name $O$. For all individuals $b$ occurring in the knowledge base, add axiom \{ $b \subseteq O$ to $K'$. For any axiom of $K'$ and any nominal schema $\{x\}$ occurring only once in this axiom, replace $\{x\}$ by $O$. Note that these nominal schemas can occur only on left-hand sides of axioms of the form $B \subseteq H$. This transformation is obviously polynomial in the size of the knowledge base.

3. Let $A \subseteq B \subseteq H$ be an axiom of $K'$ containing a safe occurrence of a nominal schema $\{x\}$. Let $S_i(a_i, \exists R_i, D_i, x)$ be all the safe environments for all safe occurrences of $\{x\}$ in $B$, and let $C_{x,A}$ be a new atomic concept. Do the following.

1. For each individual $b$ in $K'$, add to $K'$ the axiom

$$\prod_i (\exists U.S_i(a_i, \exists R_i, D_i, b)) \subseteq \exists U.(\{b\} \cap C_{x,A}).$$

2. In $B$, first add $\exists U.C_{x,A}$ as a conjunct, resulting in $B \sqcup \exists U.C_{x,A}$. Then replace each $S_i(a_i, \exists R_i, D_i, x)$ by $\{a_i\}$. Finally, replace the (single) remaining (non-safe) occurrence of $\{x\}$, if there is any, by $C_{x,A}$.

The number of new axioms added by this step is bounded from above by $y^2$, where $y$ is the maximum of the number of axioms and the number of individuals in the original knowledge base. Hence, the step is polynomial.

The knowledge base resulting from steps (1) to (3) is nominal schema-free and thus a $\mathcal{SROEL}(\forall, \times)$ knowledge base. The arguments just given also prove Theorem 4 below. Theorem 5, a key result, will be proven in Section 4.1, after some preparations.

---

**Theorem 4.** The transformation is polynomial wrt the size of the knowledge base.

**Theorem 5.** The transformation preserves satisfiability.

**Theorem 6.** Reasoning in $\mathcal{SROEL}(\forall, \times, \{v\}_n)$, for any fixed $n$, is possible in polynomial time in the size of the knowledge base. This encompasses satisfiability checking, instance retrieval, and computing class subsumptions.

**Proof.** This follows from the facts that the transformation yields a $\mathcal{SROEL}(\forall, \times)$ knowledge base polynomial in size relative to the size of the input (Theorem 4), the transformation is satisfiability preserving (Theorem 5), and reasoning in $\mathcal{SROEL}(\forall, \times)$ is polynomial. \Box

4. DL-SAFE DATLOG

We now turn to the issue of expressing DL-safe datalog rules in our new languages. We first recall definitions for datalog.

$\Sigma = \langle N_1, N_P, N_V \rangle$ is a datalog signature, where $N_1$, $N_P$, and $N_V$ are finite and pairwise disjoint sets of individual names, predicates, and variables. The set of terms $T$ of $\Sigma$ is $N_1 \cup N_V$. A datalog atom is any expression of the form $P(t_1, \ldots, t_n)$, where $P \in N_P$ and each $t_i \in T$.

As before, we often assume a single signature $\Sigma$ and omit explicit references to it. We also assume that each predicate has an associated arity, and that $\top$ and $\bot$ are unary. Below, we assume only unary and binary predicates appear in $N_P$, and so $N_P$ can be viewed as consisting of disjoint sets of concept and role predicates.

If $B$ is a finite and nonempty set of atoms and $H$ is an atom, then $B \rightarrow H$ is a datalog rule. A datalog rule base is a set $RB$ of datalog rules. For a (fixed) positive integer $n$, an $n$-variable datalog rule base is a datalog rule base where each rule contains at most $n$ distinct variables.

**Definition 6.** Interpretations ($I$), assignments ($Z$), and the function $x^I, z^I$ are defined for datalog terms and atoms as they are in $\mathcal{SROEL}(\forall, \times, \{v\}_n)$. The interpretation of $\top$ and $\bot$ is fixed, as it is in any DL. Satisfaction for atoms, rules, and rule bases is defined below. Let $I$ be an interpretation and $Z$ a variable assignment.

1. If $P(t_1, \ldots, t_n)$ is an atom,
   - $I, Z \models P(t_1, \ldots, t_n)$ iff $(t_1^I, \ldots, t_n^I) \in P^I$; otherwise $I, Z \not\models P(t_1, \ldots, t_n)$.

2. If $S$ is a set of atoms,
   - $I, Z \models S$ iff $I, Z \models A$ for each $A \in S$.
   - $I, Z \not\models S$ iff $I, Z \not\models A$ for at least one $A \in S$.

3. If $B \rightarrow H$ is a datalog rule, $I \models B \rightarrow H$, iff $I, Z \models H$ or $I, Z \not\models B$ for each variable assignment $Z$.

4. If $RB$ is a datalog rule base, then $I$ satisfies $RB$ iff it satisfies each rule in $RB$.

$I$ is a model of an atom, rule, or rule base if it satisfies it.
Definition 6 defines a first-order logic semantics for datalog. It should be noted that a ground atom is entailed by a datalog rule base under this first-order logic semantics if, and only if, it is entailed by it under the Herbrand semantics [16]. This means, in turn, that the set of ground atoms entailed by a datalog rule base is the same whether we use the first-order logic semantics, the Herbrand semantics, or the semantics resulting from interpreting all rules in the rule-base as DL-safe. If we look at other than ground entailments, this is no longer the case. However, for datalog, usually only ground entailments are considered.

4.1 Embedding into $\text{SROEL}(\pi, x, \{v\}_n)$

If $RB$ is a datalog rule base defined over a signature $\Sigma = (N_1, N_P, N_V)$, then $RB$ can be embedded into an equisatisfiable $\text{SROEL}(\pi, x, \{v\}_n)$ knowledge base $\text{trans}(RB)$ over a signature $\Sigma'$. The embedding obviously also works for $\text{SROIQB}(x, \{v\})$. Below, let $N_{P,i}$, $i \in \{1, 2\}$ be the unary and binary predicates of $N_P$.

Definition 7. If $\Sigma = (N_1, N_P, N_V)$:

1. $\Sigma' = (N_1, N_C, N_R, N_V)$, where
   (a) $N_C = N_{P,1}$;
   (b) $N_R = N_{P,2} \cup \{U\}$ (where $U$ is the universal role).
2. for each unary or binary atom $A$ over $\Sigma$,
   (a) if $A$ is $R(t, u)$, where $t, u \in T$, then $\text{trans}(A)$ is $\exists A U (\{t\} \cap \exists R (\{u\})$.
   (b) if $A$ is $C(t)$, where $t \in T$, then $\text{trans}(A)$ is $\exists A U (\{t\} \cap C)$.
3. for each set $X = \{A_1, \ldots, A_n\}$ of binary and unary atoms, $\text{trans}(X) = \text{trans}(A_1) \cap \ldots \cap \text{trans}(A_n)$.
4. for each $B \rightarrow H \in RB$, $\text{trans}(r) = \text{trans}(B) \rightarrow \text{trans}(H)$.
5. $\text{trans}(RB) = \{\text{trans}(r)|r \in RB\}$.

Observe that no role inclusion axioms appear in $\text{trans}(RB)$, and so all roles can be considered simple. Since this is so, $\text{trans}(RB)$ is a $\text{SROEL}(\pi, x, \{v\}_n)$ knowledge base, where $n$ is the maximum number of nominal schemas appearing in any rule of $\text{trans}(RB)$.

Every interpretation for $RB$ or its translation can be taken as one for its counterpart, provided $U$ is interpreted in the usual fashion. As this is so, we assume a common domain of discourse $\Delta^Z$.

Lemma 1. Let $A$ be a binary or unary atom in $RB$. For any interpretation $I$ and assignment $Z$,

1. $I, Z \models A \iff \text{trans}(A)^{I, Z} = \Delta^Z$ and
2. $I, Z \not\models A \iff \text{trans}(A)^{I, Z} = \emptyset$.

Proof. For part (1), suppose $I, Z \models A$ and let $d \in \Delta^Z$. If $A = C(t)$, then $t^Z \in C^Z$. Since $(d, t^Z) \in U^Z$ and $t^Z \in \{t\}^Z$, it follows that $d \in \exists A (\{t\} \cap C)^Z$. If instead $A = R(t, u)$, then $(u^Z, t^Z) \in U^Z$. Since $u^Z \in \{u\}^Z$, it must be that $t^Z \in \exists R (\{u\})^Z$. Since $t^Z \in \{t\}^Z$ and $(d, t^Z) \in U^Z$, it follows that $d \in \exists A (\{t\} \cap \exists R (\{u\}))^Z$. Generalizing on $d$, $\text{trans}(A)^{I, Z} = \Delta^Z$.

Now suppose $\text{trans}(A)^{I, Z} = \Delta^Z$. If $\text{trans}(A)$ is $\exists U (\{t\} \cap C)$, then for any $d \in \Delta^Z$, it follows that $(d, t^Z) \in U^Z$ and $t^Z \in \{t\}^Z$. And so $I, Z \models C(t)$. If $\text{trans}(A)$ is $\exists U (\{t\} \cap \exists R (\{u\})$, then it follows that for any $d \in \Delta^Z$, $(d, t^Z) \in U^Z$ and $t^Z \in \{t\}^Z$. From this, $(u^Z, t^Z) \in \emptyset$, and so $I, Z \models R(t, u)$.

For (2), suppose $I, Z \not\models A$. Suppose for a proof by contradiction that there is a $d \in \Delta^Z$ such that $d \in \text{trans}(A)^{I, Z}$. If $A = C(t)$, then $t^Z \notin C^Z$. Since $d \in \text{trans}(A)^{I, Z}$, it follows that $(d, t^Z) \in U^Z$ and $t^Z \in \{t\}^Z$. If instead $A = R(t, u)$, then $(u^Z, t^Z) \notin \emptyset$. Since $d \in \text{trans}(A)^{I, Z}$, it follows that $(d, t^Z) \in U^Z$ and $(u^Z, t^Z) \in \emptyset$. Either way, a contradiction occurs, and $\text{trans}(A)^{I, Z} = \emptyset$.

Now suppose $\text{trans}(A)^{I, Z} = \emptyset$. If $I, Z \models A$, then from the first part of the proof, $\text{trans}(A)^{I, Z} = \emptyset$. And so it must be that $I, Z \not\models A$. ☐

The following is the key result of the embedding.

Theorem 7. $RB$ is satisfiable iff $\text{trans}(RB)$ is.

Proof. Let $I$ be a model of $RB$ and $Z$ an assignment for $I$. Suppose $B \rightarrow H \in RB$. If $I, Z \models H$, then $\text{trans}(H)^{I, Z} = \Delta^Z$ by Lemma 1. Similarly, if $I, Z \not\models B$, then there exists a $B_j \in B$ such that $I, Z \not\models B_j$. Again by Lemma 1, $\text{trans}(B_j)^{I, Z} = \emptyset$, and so $\text{trans}(B)^{I, Z} = \emptyset$. Either way, $I, Z \models \text{trans}(B \rightarrow H)$. Generalizing on $Z$ and $B \rightarrow H$, $I$ models $\text{trans}(RB)$.

Now let $I$ be a model of $\text{trans}(RB)$ and $Z$ an assignment for $I$. Suppose $B \rightarrow H \in RB$ and $I, Z \models B$. For each $A \in B$, $I, Z \models A$. By Lemma 1, $\text{trans}(A)^{I, Z} = \emptyset$ for each $A \in B$. From this, it follows that $\text{trans}(B)^{I, Z} = \emptyset$. Since $I, Z \models \text{trans}(B \rightarrow H)$, it must be that $\text{trans}(H)^{I, Z} = \emptyset$, and so $I, Z \models H$ by Lemma 1. As such, $I, Z \models B \rightarrow H$. Generalizing on $Z$ and $B \rightarrow H$, $I$ models $RB$. ☐

The key insight from Theorem 7 is that $RB$ and $\text{trans}(RB)$ yield the same ground (i.e., ABox) entailments: $RB$ entails a ground atom $A$ if $RB \cup \{A \rightarrow \bot\}$ is unsatisfiable.

The following theorem is more tricky. While it shows that $\text{SROEL}(\pi, x, \{v\}_n)$ knowledge bases can be combined with $n$-variable datalog rule bases without jeopardizing polymodality, it is important to realize that the datalog rules are semantically interpreted in a DL-safe way, through the use of nominal schemas in the transformation. Hence, the set of entailments of such a combined knowledge base differs from the set of entailments which the hybrid knowledge base would have under a straightforward first-order logic semantics. Entailment under such a first-order semantics, however, would be undecidable.

Theorem 8. If $KB$ is a $\text{SROEL}(\pi, x, \{v\}_n)$ knowledge base and $RB$ an $n$-variable datalog rule base, then $KB' = KB \cup \text{trans}(RB)$ is a $\text{SROEL}(\pi, x, \{v\}_n)$ knowledge base.
obtained by defining $K$ to be $\langle N_1 \rangle^M$. Given a model of $K'$, a model $M'$ of $K'$ is obtained by defining $O^M$ to be $\langle N_1 \rangle^M$. Given a model of $K'$, a model for $K$ is obtained by restricting the signature (i.e., removing $O$).

For step (3), again let $K$ be the knowledge base before the transformation, and let $K'$ be the knowledge base after the transformation. Now note that the axiom

$$\prod_i (\exists U.S_i(a_i, \exists R_i.D_i, b)) \subseteq \exists U.\{(b) \cap C_{x,a}\}$$

as given in step (3) of the transformation is similar in structure to the results of embedding datalog rules as discussed above. This insight guides the rest of the proof.

For any interpretation $I$ for $K$ and any concept $C_{x,a}$, we extend $I$ to $I'$ by defining $C^*_z$ to be the set of $b^z$ ($b \in N_I$) such that $\prod_i (\exists U.S_i(a_i, \exists R_i.D_i, b^z))$ is not empty. Below, $C_j(A, x, b)$ is shorthand for such a conjunction. We assume a common domain of discourse $\Delta$ for both $I$ and $I'$.

(LR) Assume $M$ models $K$ and $A \subseteq K$. If $A$ contains no nominal schemata (this holds for all RIA and ABox assertion), $M$ is a model of $A$ iff $M'$ is. So suppose $A$ is the TBox axiom $B \subseteq H$ such that $S_i(a_i, \exists R_i.D_i, v) \ldots S_n(a_n, \exists R_n.D_n, v)$ are the safe environments for $\{v\}$ in $B$. Let $Z$ be a variable assignment. The transformation produces axioms of the below two forms:

1. $\prod_i (\exists U.S_i(a_i, \exists R_i.D_i, b)) \subseteq \exists U.\{(b) \cap C_{y,A}\}$, $b \in N_I$.
2. $\exists U.C_{y,A} \cap B^z \subseteq H$

where $C_{y,A}$ is a new atomic concept and $B^z$ is $B$ with each $S_i(a_i, \exists R_i.D_i, v)$ replaced with $\{a_i\}$, and the one non-safe occurrence (if present) of $\{v\}$ is replaced with $C_{y,A}$. No axiom of form 1 contains any nominal schema.

Let $P$ be a form 1 axiom (with $b \in N_I$ replacing $v$). Suppose $x \in C_j(A, v, b)^{M',z}$. According to the specification of

Suppose $P$ has form 2 and $x \in (\exists U.C_{y,A} \cap B^*')^{M',z}$. Then there is a $d \in N_I$ such that $(x, d^M) \in U^{M',z}$ and $d^M \in C_{y,A}^{M',z}$. From this, $C_j(A, v, d)^{M',z} \neq \emptyset$. If there is an occurrence of $C_{y,A}$ in $B^z$, then the following holds:

(*) There is an $e \in N_I$ with $e^M \in C_{y,A}^{M',z}$ such that $x \in (\exists U.C_{y,A} \cap B^*')^{M',z}$, where $B^*$ is obtained from $B^z$ by replacing $C_{y,A}$ by $\{e\}$.

To show (*), we distinguish two cases. (1) If $B^* = C_{y,A} \cap D$ for some concept description $D$, then $x \in C_{y,A}^{M',z}$. By definition of the extension of $C_{y,A}$ under $M'$, there must be an $e \in N_I$ with $x = e^M$, and so $x \in (\exists U.C_{y,A} \cap \{e\} \cap D)^{M',z}$, i.e., $x \in (\exists U.C_{y,A} \cap B^*')^{M',z}$. (2) Otherwise, note that $B^*$ is of the form $B^* = D_1 \cap \exists R_1.(D_2 \cap \exists R_2.\ldots(D_k \cap \exists R_k.(D_{k+1}(C_{y,A}))))$, where all $D_i$ are concept descriptions. Since we have $x \in (B^*)^{M',z}$, there must be a selection of elements $y_0, \ldots, y_k$ in $\Delta^M$ such that $x = y_0$ and $(y_{j-1}, y_j) \in R_j^{M',z}$ for all $j = 1, \ldots, k$. By definition of the extension of $C_{y,A}$ under $M'$, there must thus be an $e \in N_I$ with $y_k = e^M \in C_{y,A}^{M',z}$, and (*) follows in this case. This concludes the proof of (*).

Now note that $x \in (B^*)^{M',z}$ and thus $x \in (B^*')^{M',z}$. Let $Z'$ be an assignment such that $Z'(u) = Z(u)$ for all variables $u \neq v$ and such that $Z'(v) = e^M$ if $C_{y,A}$ occurs in $B^*$, and $Z'(v) = d^M$ otherwise. We induct on the depth of the expression tree of $B^*$, showing that for any sub-concept $C^*$ of $B^*$, and for any $u \in \Delta$, if $u \in (C^*)^{M,\Delta}$, then $u \in (C^*)^{M,\Delta}$, where $C$ is the subformula of $B$ from which $C^*$ is constructed.

For the base case (the leaves, where $\exists R Self$ is considered a leaf), if $C^* = C$, then $\{e\}$ does not appear in $C$, and so $(C^*)^{M,\Delta} = C^{M,\Delta}$. So suppose $C^* \neq C$. Since $C^*$ is a leaf, we must distinguish two cases. (i) If $C^* = \{e\}$ (coming from $C_{y,A}$ via (*)), then $u \in C^{M,\Delta}$ by definition of $Z'$. (ii) In the remaining case, the leaf has the form $\{a\}$, where $C$ is of the form $\{a\} \cap \exists R \{v\}$. As such, the portion of $C_j(A,v,c)$ corresponding to $C$ is just $\exists U.(\{a\} \cap \exists R \{v\})$. It follows that $u = a^M$. Therefore, $\exists U.(\{a\} \cap \exists R \{c\})^{M,\Delta} \neq \emptyset$ (because $C_j(A,v,c)^{M,\Delta} \neq \emptyset$ and $C_j(A,v,c)^{M,\Delta} = C(A,v,c)^{M,\Delta}$). Let $z \in U.(\{a\} \cap \exists R \{c\})^{M,\Delta}$, then $z = a^M \in U^{M,\Delta}$ and $a^M \in \exists R \{v\}^{M,\Delta}$. Since $u = a^M$, $u \in (\{a\} \cap \exists R \{v\})^{M,\Delta}$. It follows that $u \in (\{a\} \cap \exists R \{v\})^{M,\Delta}$, i.e., $u \in C^{M,\Delta}$.

For the induction, it must be that $C$ is of the form $C_1 \cap C_2$ or $\exists R.C_1$. If the former, then $u \in (C_1 \cap C_2)^{M,\Delta}$ and so $u \in (C_1)^{M,\Delta}$ and $u \in (C_2)^{M,\Delta}$. By inductive hypothesis, $u \in C_1^{M,\Delta}$ and $u \in C_2^{M,\Delta}$, and so $u \in (C_1 \cap C_2)^{M,\Delta}$. If the latter, then there exists a $w \in \Delta$ such that $(u, w) \in R^{M,\Delta}$.
and \( w \in (C_i^*)^{M,Z} \). By inductive hypothesis, \( w \in C_i^{M,Z'} \). From this, \( u \in \exists R.C_i^{M,Z'} \). Either way, \( u \in C^{M,Z'} \).

Given the above induction, \( x \in B^{M,Z'} \). Since \( M \) models \( B \subseteq H \), \( x \in H^{M,Z} \). Since \( H \) contains neither nominal schemas nor \( C,v,A \), \( H^{M,Z} = H^{M,Z'} \), and so \( x \in H^{M,Z} \).

Given this, \( M \) and \( Z \) satisfy \( P \).

Generalizing on \( Z \) and then \( A, M \) models \( K' \).

(RL) Let \( M' \) be a model of \( K' \) and \( M \) its restriction to the terms appearing in \( K \). Let \( A \in K \); we assume that \( A \) is a concept inclusion axiom of the form \( B \subseteq H \). Let \( Z \) be an assignment and suppose that \( x \in B^{M,Z} \). Observe that \( B^{M,Z} = B^{M,Z'} \) and furthermore that \( Z(v) = c^M \equiv c^M \) for some \( c \in N_j \). We may assume wlog that \( \models \) does not appear in \( B \), since otherwise \( A \) would be trivially modelled by every interpretation \( T \). Since this is so, and since \( x \in B^{M,Z} \) it must be that each \( \mathcal{S}_i(a_i \in R_i.D_i, c_i)^{M,Z} \) is nonempty. As such, there is a \( z \in \Delta \) such that \( z \in \mathcal{S}_i(a_i \in R_i.D_i, c_i)^{M,Z} \). It follows that \( x \in \mathcal{U}( \mathcal{S}_i(a_i \in R_i.D_i, c_i))^{M,Z} \). Generalizing on \( S_i \), \( x \in C_j(A, v, c)^{M,Z} \). From this, \( x \in \exists U.(\{c\} \cap C,v,A)^{M,Z} \), and so \( x \in \exists U.(\{c\} \cap C,v,A)^{M,Z} \).

We now induct on the degree of \( B^* \), showing that for any subconcept \( C^* \) of \( B^* \) and any \( u \in \Delta \), if \( u \in C^{M,Z} \), then \( u \in (C^*)^{M,Z} \). For the base case, if \( C^* = C \), then there's nothing to prove. For \( C^* \neq C \) we consider the two possible cases. If \( C^* \) is \( \{a\} \), corresponding to the safe environment \( C = \{a\} \cap \exists R.R \{v\} \), then \( u \in \{(a) \cap \exists R.R \{v\})^{M,Z} \), and so clearly \( u \in (C^*)^{M,Z} \).

If instead \( C^* \) is \( C,v,A \), corresponding to \( C = \{v\} \), then \( u = v^{M,Z} \equiv c^M \). Recall that \( x \in C_j(A, v, c)^{M,Z} \), i.e., \( C_j(A, v, c)^{M,Z} \) is not empty. As such, \( c^M \in C^*_v,A^* \), and so \( u \in (C^*)^{M,Z} \).

For the induction, we consider the two viable forms of the \( C \) corresponding to \( C^* \). If \( C \) has form \( C_1 \cap C_2 \), then \( u \in c^{M,Z} \) and \( u \in c^{M,Z} \), and so by inductive hypothesis \( u \in (C_1)^{M,Z} \) and \( u \in (C_2)^{M,Z} \). As such, \( u \in (C_1 \cap C_2)^{M,Z} \). If \( C \) is instead of the form \( \exists R.C_1 \), then there exists a \( w \in \Delta \) such that \( (u, w) \in R^{M,Z} \) and \( w \in (C_1)^{M,Z} \). By inductive hypothesis, \( w \in (\exists R.C_1)^{M,Z} \). From this, \( u \in (\exists R.C_1)^{M,Z} \).

Since \( x \in B^{M,Z} \), we conclude that \( x \in (B^*)^{M,Z} \), and therefore \( x \in \exists U.(C,v,A \cap B^*)^{M,Z} \). As \( M' \) models \( K' \), we may conclude \( x \in H^{M,Z} \), and since \( H^{M,Z} = H^{M,Z} \), \( x \in H^{M,Z} \).

And so \( M,Z \models A \). Generalizing on \( Z \), \( M \) models \( A \). Generalizing on \( A, M \) models \( K \).

4.2 Embedding DLP 2 and DL-Lite\(_R\)

For the following, note that DLP 2 [13] is essentially OWL 2 RL [18]. We show that it can be embedded into ELP 2.

A DLP role is any \( SROIQ \) role expression. A DLP body concept description is a \( SROIQ \) concept description constructed from elements of \( N_C \cup O \cup \{\top, \bot\} \) and \( \exists \) and \( \forall \). A DLP head concept description is a \( SROIQ \) concept description constructed from elements of \( N_C \cup O \cup \{\top, \bot\} \) and \( \forall, \cap \), and expressions of the form \( \leq 1R.C \), where \( R \) is a \( SROIQ \) role expression and \( C \) is a body concept description. A DLP RBox axiom is any expression of the form \( R \subseteq S \) or \( R \cap R \subseteq R \), where \( R,S \) are DLP roles. A DLP TBox axiom is any expression of the form \( C \subseteq D \), where \( C \) is a DLP body concept and \( D \) is a DLP head concept.

An DLP ABox axiom is any expression of the form \( D(a,b) \), where \( D \) is a DLP head concept, \( R \) is a DLP role, and \( a,b \in N_i \). A DLP knowledge base is a set of Rbox, TBox, and ABox axioms.

Let \( T \) be a set of terms as defined earlier. If \( C \) is a DLP concept description and \( t \in T \), then \( C(t) \) is a concept atom. If \( R \) is a DLP role and \( t \in T \), then \( R(t,u) \) is a role atom.

Given a set \( S \) of (concept and role) atoms, and \( t,u \in T \), a path from \( t \) to \( u \) in \( S \) is a nonempty sequence \( R_1(x_1,x_2), \ldots, R_i(x_n,x_{n+1}) \), where \( x_1 = t \), \( x_{n+1} = u \), and \( x_i \in N_V \) for all \( 2 \leq i \leq n \). A term \( t \) is initial in \( S \) if there is no path to \( t \) in \( S \); \( t \) is final in \( S \) if there is no path starting with \( t \) in \( S \).

A DL rule is any expression of the form \( B \rightarrow H \), where \( B \) and \( H \) are sets of atoms and:

1. for any term \( u \) not initial in \( B \), there is a path in \( B \) from exactly one initial term to \( u \);
2. for any terms \( t,u \), there is exactly one path from \( t \) to \( u \) in \( B \);
3. if \( C(t) \) or \( R(t,u) \) are in \( H \), then \( t \) is initial in \( B \).

A DL rule base is any set of DL rules. A DLP rule base is a DL rule base such that every concept atom in \( B \) is a body concept, and every head concept in \( H \) is a head concept. A DLP 2 knowledge base is a DLP knowledge base together with a DLP rule base. Axioms of the form \( Dis(R,S) \) and \( Assg(R) \) are also allowed, where \( R \) and \( S \) are DLP roles.

Theorem 9. Any DLP 2 knowledge base can be converted into an equisatisfiable \( SROEL(\wedge, \cdot, \{\cdot\}_\alpha) \) knowledge base in polynomial time.

Proof. As shown in [13], a DLP 2 knowledge base \( KB \) can be converted in polynomial time to a datalog rule base \( RB \) in which (i) each rule of \( RB \) has at most 5-variables, and (ii) \( KB \) is satisfiable if and only if \( RB \) is. Each predicate in the rule base \( RB \) is either unary or binary, and so the conversions of Section 4 apply. According to Theorem 7, \( RB \) is satisfiable if and only if \( \text{trans}(RB) \) is, where \( \text{trans}(RB) \) is the \( SROEL(\wedge, \cdot, \{\cdot\}_\alpha) \) translation of \( RB \). We conclude that \( KB \) and \( \text{trans}(RB) \) are equisatisfiable. □

DL-Lite\(_R\) is essentially OWL 2 QL [18]. From the definition of DL-Lite\(_R\) (which we do not repeat here), it is immediately clear that all of the language, except inverse roles, is covered by \( SROEL(\cap, \cdot, \{\cdot\}_\alpha) \). For other uses of inverse roles, see [18].

10D. Calvanese and R. Rosati, personal communications.
roles, we can approximate statements of the form $R \subseteq S^\ominus$ by the DL-safe rule $R(x, y) \rightarrow S(y, x)$, or more precisely by their corresponding DL axiom as described in Section 4.

We can thus say that ELP 2 covers most of OWL 2 QL.

5. CONCLUSIONS AND FUTURE WORK

We have provided a language which seamlessly integrates OWL DL and datalog (and thus SWRL and RIF-Core) rules. The language is firmly based on description logic traditions and adheres to the usual conceptual design decisions, including decidability of the language. We have also provided a sublanguage, called ELP 2, which is tractable and covers variable-restricted datalog and all major tractable profiles of OWL 2 DL (with the mentioned exception of some uses of inverse roles in OWL 2 QL).

The complexity analyses which we provided also yield key steps to a naive algorithmization of reasoning in these languages. However, there is much more work to be done in this respect, in particular regarding efficient algorithmizations and implementations.

For $SROIQB\{\times, \{\nu\}\}$, naive grounding together with an extension of existing algorithms for dealing with role constructors provides a first approach to algorithmization. It is expected that system performance will be the main bottleneck in such investigations.

For ELP 2, the transformation provided in the proof of Theorem 5 provides a starting point, possibly together with reasoning approaches laid out for $SROEL$ in [12]. However, we believe that there should be better solutions by developing algorithms which avoid the blow-up of the knowledge base through grounding, and we intend to investigate these in the near future.

Since datalog rules are expressible in ELP 2, ELP 2 may also provide a natural gateway to incorporating closed-world reasoning into description logics. This could be accomplished by applying the semantics of nonmonotonic rule-based reasoning approaches to the transformed rules.

Acknowledgements This work was supported by the National Science Foundation under award 1017225 “III: Small: TRON—Tractable Reasoning with Ontologies”. The second author is funded by a Fulbright Indonesia Presidential Scholarship PhD Grant 2010.

6. REFERENCES


