Towards an Efficient Algorithm to Reason over Description Logics extended with Nominal Schemas

David Carral, Cong Wang, and Pascal Hitzler

Kno.e.sis Center, Wright State University, Dayton, OH, U.S.A.

Abstract. Extending description logics with so-called nominal schemas has been shown to be a major step towards integrating description logics with rules paradigms. However, establishing efficient algorithms for reasoning with nominal schemas has so far been a challenge. In this paper, we present an algorithm to reason with the description logic fragment $\mathcal{ELROV}_n$, a fragment that extends $\mathcal{EL}^{++}$ with nominal schemas. We also report on an implementation and experimental evaluation of the algorithm, which shows that our approach is indeed rather efficient.

1 Introduction

Nominal schemas have been introduced in [18] based on prelminary ideas in [15,17] and, essentially, as a generalization of the idea of DL-safety for rules [5,19]. Essentially, nominal schemas are a kind of variable nominals, i.e. they are variables which can be bound to known individuals only. A typical example for the use of nominal schemas (taken from [14]) would be

$$\begin{aligned}&\exists \text{hasReviewAssignment}.((\{x\} \sqcap \exists \text{hasAuthor}.\{y\}) \sqcap (\{x\} \sqcap \exists \text{atVenue}.\{z\}))\\ &\quad \sqcap \exists \text{hasSubmittedPaper}.(\exists \text{hasAuthor}.\{y\} \sqcap \exists \text{atVenue}.\{z\})\\ &\quad \sqsubseteq \exists \text{hasConflictingAssignedPaper}.\{x\}.\end{aligned}$$

In this case, think of the three nominal schemas $\{x\}$, $\{y\}$ and $\{z\}$ as placeholders for nominals—in fact this axiom can be translated into n^3 axioms without nominal schemas by *fully grounding* the axiom, where n is the number of known individuals in the knowledge base (see [14]). Full grounding eliminates nominal schemas and thus can be used, in principle, for reasoning over nominal-schema-extended knowledge bases. However, as the example indicates, fully grounding an axiom with k nominal schemas results in n^k new axioms without nominal schemas, i.e. the size of the *input* knowledge base to a reasoning algorithm becomes unmanageable for current algorithms (see [4] and Section 4 below).

The rationale for introducing nominal schemas lies in bridging the gap between description-logic-based and rule-based approaches for ontology modeling

[5,14]. Indeed, the example above arises from the rule

$$\begin{aligned}&\text{hasReviewAssignment}(v, x) \wedge \text{hasAuthor}(x, y) \wedge \text{atVenue}(x, z)\\ &\wedge \text{hasSubmittedPaper}(v, u) \wedge \text{hasAuthor}(u, y) \wedge \text{atVenue}(u, z)\\ &\qquad \rightarrow \text{hasConflictingAssignedPaper}(v, x)\end{aligned}$$

if x, y and z are considered to be *DL-safe variables*,[1] i.e., they bind only to constants present in the knowledge base. In [18] it was shown that DL-safe binary Datalog is completely subsumed by nominal-schema-extended description logics, and in [12] this was lifted to n-ary DL-safe Datalog. This means that nominal schemas allow for an incorporation of DL-safe SWRL [8,19] into the description logic paradigm. It was also shown in [12] that the use of nominal schemas together with autoepistemic operators yields a description logic which encompasses most of the major paradigms in non-monotonic logic programming and in local-closed-world-extended description logics (see also [11]), thus consituting a major step towards establishing a unifying logic for major Semantic Web languages around the W3C standards OWL [6] and RIF [10].

It was shown in [18] that extending $\mathcal{SROIQ}$ with nominal schemas does not result in an increase of worst-case complexity, and it was also shown that a tractable fragment can be obtained which encompasses both OWL EL and the DL-safe version of OWL RL [6] (and, more generally, Datalog under the Herbrand semantics provided there is a global bound on the number of variables per rule). However, despite this, first attempts to arrive at an efficient algorithmization of reasoning with nominal schemas have had limited success: [13] reported on a corresponding extension of tableaux algorithms, while [21] reported on a resolution-based algorithm for the tractable fragment—but neither of these algorithms looks promising enough in terms of scalability to even attempt an implementation.

In this paper, we therefore present an algorithm for OWL EL (more precisely, for $\mathcal{ELROV}_n$) based on an algorithm for OWL EL presented in [16] which uses a transformation into Datalog. We also report on an implementation and on corresponding experimental evaluations based on the IRIS Datalog reasoner [3], which show that our approach is feasible in terms of scalability.

The plan of the paper is as follows. In Section 2, we recall preliminaries on the description logic $\mathcal{ELROV}_n$. In Section 3 we describe our algorithm. In Section 4 we present our implementation and evaluation. We conclude in Section 5.

Acknowledgements. This work was supported by the National Science Foundation under award 1017225 *III: Small: TROn – Tractable Reasoning with Ontologies.*

2 The Logic $\mathcal{ELROV}_n$

In this section we define the syntax and semantics of $\mathcal{ELROV}_n$, which extends $\mathcal{EL}^{++}$ [2] with nominal schemas and subsumes OWL EL [6]. We assume that

[1] This notion was introduced in [17].

Name	**Syntax**	**Semantics**
Concept	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
Role	R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
Individual	a	$a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
Variable	v	$\mathcal{Z}(x) \in \Delta^{\mathcal{I}}$

Concept Constructor	**Syntax**	**Semantics**
Concept Conjunction	$C \sqcap D$	$C^{\mathcal{I},\mathcal{Z}} \cap D^{\mathcal{I},\mathcal{Z}}$
Existential Restriction	$\exists R.C$	$\{x \mid y \in \Delta^{\mathcal{I}} \text{ s.t. } (x,y) \in R^{\mathcal{I},\mathcal{Z}}, y \in C^{\mathcal{I},\mathcal{Z}}\}$
Self Restriction	$\exists R.\text{Self}$	$\{x \mid (x,x) \in R^{\mathcal{I},\mathcal{Z}}\}$
Nominal (Schema)	$\{t\}$	$\{t^{\mathcal{I},\mathcal{Z}}\}$
Top	$\top$	$\Delta^{\mathcal{I}}$
Bottom	$\bot$	$\emptyset$

Axiom	**Syntax**	**Semantics**
Concept Assertion	$E(a)$	$a^{\mathcal{I},\mathcal{Z}} \in E^{\mathcal{I},\mathcal{Z}}$
Role Assertion	$R(a,b)$	$(a^{\mathcal{I},\mathcal{Z}}, b^{\mathcal{I},\mathcal{Z}}) \in R^{\mathcal{I},\mathcal{Z}}$
General Concept Inclusion	$C \sqsubseteq D$	$C^{\mathcal{I},\mathcal{Z}} \subseteq D^{\mathcal{I},\mathcal{Z}}$
Role Inclusion Axiom	$R \sqsubseteq S$	$R^{\mathcal{I},\mathcal{Z}} \subseteq S^{\mathcal{I},\mathcal{Z}}$
Role Chain Axiom	$R \circ S \sqsubseteq T$	$\{(x,z) \mid (x,y) \in R^{\mathcal{I},\mathcal{Z}}, (y,z) \in S^{\mathcal{I},\mathcal{Z}}\} \subseteq T^{\mathcal{I},\mathcal{Z}}$
Concept Product	$R \sqsubseteq C \times D$	$\{R^{\mathcal{I},\mathcal{Z}} \subseteq C^{\mathcal{I},\mathcal{Z}} \times D^{\mathcal{I},\mathcal{Z}}\}$
	$C \times D \sqsubseteq R$	$\{C^{\mathcal{I},\mathcal{Z}} \times D^{\mathcal{I},\mathcal{Z}} \subseteq R^{\mathcal{I},\mathcal{Z}}\}$

where C, D are concept expressions, $E \in N_C$, $R, S \in N_R$, $a, b \in N_I$, $v \in N_v$, and $x, y \in \Delta^{\mathcal{I}}$

Fig. 1. Syntax and semantics of DL constructors

the reader is familiar with basic description logic (DL) notation and results, and refer to [1,7] for background reading.

Every $\mathcal{ELROV}_n$ knowledge base KB is defined over a signature $\mathcal{L}$ composed of four mutually disjoint finite[2] sets of *concept names* N_C, *individual names* N_I, *role names* N_R, and *variable names* N_V. Given a signature, the set $\mathbf{C}$ of $\mathcal{ELROV}_n$ concept expressions is defined inductively to contain the expressions in the upper part of Figure 1. The set of $\mathcal{ELROV}_n$ axioms is then defined in the lower part of Figure 1. As usual in DL, we distinguish between axioms of ABox (assertional axioms), TBox (terminological axioms or general concept inclusions), and RBox (role related axioms).

For a set KB of $\mathcal{ELROV}_n$ axioms to qualify as an $\mathcal{ELROV}_n$ knowledge base, further syntactic restrictions need to be satisfied. We continue by introducing some preliminary definitions that allow us to declare these restrictions.

Let π be a function that maps every DL axiom α into a first order logic axiom $\pi(\alpha)$ as defined in Figure 2. We recursively define the set N_R^C of role names with respect to a set KB of $\mathcal{ELROV}_n$ axioms to contain all roles T such that $R \circ S \sqsubseteq T \in KB$ or $R \sqsubseteq T \in KB$ where $R \in N_R^C$. We define the set

[2] but large enough

Concept and Role Expressions

$$\pi_x(\bot) = \bot$$
$$\pi_x(\top) = \top$$
$$\pi_x(A) = A(x)$$
$$\pi_x(C \sqcap D) = \pi_x(C) \land \pi_x(D)$$
$$\pi_x(\exists R.C) = \exists y[R(x,y) \land \pi_y(C)]$$
$$\pi_x(\{a\}) = x \approx a$$

Axioms

$$\pi(C \sqsubseteq D) = \forall x[\pi_x(C) \to \pi_x(D)]$$
$$\pi(R \sqsubseteq S) = \forall x \forall y[R(x,y) \to S(x,y)]$$
$$\pi(R \circ S \sqsubseteq T) = \forall x \forall y \forall z[R(x,y) \land S(y,z) \to T(x,z)]$$
$$\pi(R \sqsubseteq C \times D) = \forall x \forall y[R(x,y) \to \pi_x(C) \land \pi_y(D)]$$
$$\pi(C \times D \sqsubseteq R) = \forall x \forall y[\pi_x(C) \land \pi_y(D) \to R(x,y)]$$
$$\pi(C(a)) = \to C(a)$$
$$\pi(R(a,b)) = \to R(a,b)$$

where C, D are $\mathcal{ELROV}_n$ concept expressions, $R, S \in N_R$, $a \in N_I \cup N_V$, and x, y are fresh new first-order predicate logic variables

Fig. 2. Translating $\mathcal{ELROV}_n$ into first-order predicate logic

N_R^S of role names as $N_R^S = N_R / N_R^C$. We call the roles contained in the set N_R^C (resp. N_R^S) complex (resp. simple) roles with respect to a set *KB* of $\mathcal{ELROV}_n$ axioms. We frequently drop the "with respect to a set *KB* of $\mathcal{ELROV}_n$ axioms" as this is clear from the context. Furthermore, we define $ran(R)$ where $R \in N_R$ as the set of $\mathcal{ELROV}_n$ concept expressions containing all concepts D such that $R \sqsubseteq S_1, \ldots, S_{n-1} \sqsubseteq S_n \in KB$ and $S_n \sqsubseteq C \times D \in KB$ for some $S_1, \ldots, S_n \in N_R$ and $n \geq 0$.

Definition 1 **($\mathcal{ELROV}_n$ Restrictions).** *An* $\mathcal{ELROV}_n$ knowledge base *is a set KB of* $\mathcal{ELROV}_n$ *axioms which satisfies all of the following conditions:*

1. *All roles appearing in expressions of the form $\exists R.Self$ in KB are simple.*
2. *For every axiom of the form $R \circ S \sqsubseteq T \in KB$ we have that $ran(T) \subseteq ran(S)$.*
3. *For every $\mathcal{ELROV}_n$ axiom α containing nominal schemas we have that $\pi(\alpha)$ does not contain more than n different free variables and α does not contain more than n different nominal schemas.*
4. *Axioms of the form $R \sqsubseteq C \times D$ and $C \times D \sqsubseteq R$ do not contain nominal schemas and axioms containing nominal schemas of the form $C \sqsubseteq D$, where C and D are $\mathcal{ELROV}_n$ concept expressions, do not contain occurrences of the top $\top$ and bottom $\bot$ concepts in C.*
5. *If $\alpha \in KB$ is an axiom containing a nominal schema, then for any class-subexpression of the form $\exists R.D$ on the right hand side of the general class inclusion, we have that D is of the form $\{x\} \sqcap C$, where $\{x\}$ is a nominal schema and C is a class expression. We refer to this nominal schema $\{x\}$ as guard of subconcept D*

Some explanations are in order. The first two restrictions are inherited from $\mathcal{EL}^{++}$. The third is required for obtaining tractability (however, our algorithm does not need to know what n is). The fourth condition can actually be relaxed, however this would make our exhibition more involved, and we decided to go for the simpler variant as we have not been able to come up with good examples which would use nominal schemas in this types of axioms. The last condition forbids the occurrence of axioms that contain both nominal schemas and unrestricted existentially quantified variables on the right hand side. This is the only of the five restrictions which really impacts on the language—as shown in [18] it is not required for obtaining tractability. However we need it for our algorithm to work. We conjecture that a modification of our algorithm would be able to avoid this restriction, however details remain to be worked out.

Note that we do not include the role regularity restriction that applies to the OWL profile languages, as defined in [9] and that our defined restrictions are equivalent to the ones defined for $\mathcal{EL}^{++}$ if a knowledge base KB does not include nominal schemas.

The semantics of $\mathcal{ELROV}_n$ is specified by defining an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ where $\Delta^{\mathcal{I}}$ is a non-empty set, and $\cdot^{\mathcal{I}}$ is an interpretation function that maps individual, concept, and role names as shown in Figure 1. A variable assignment $\mathcal{Z}$ for an interpretation $\mathcal{I}$ is a function $\mathcal{Z} : N_V \to \Delta^{\mathcal{I}}$ such that for each $v \in N_V$, $\mathcal{Z}(v) = a^{\mathcal{I}}$ for some $a \in N_I$. Interpretations and assignments are extended to concept expressions as shown in Figure 1.

An $\mathcal{ELROV}_n$ axiom α is *satisfied* by $\mathcal{I}$ and $\mathcal{Z}$, written $\mathcal{I}, \mathcal{Z} \models \alpha$, if the conditions defined by the lower part of Figure 1 hold. An interpretation $\mathcal{I}$ satisfies an axiom α, written $\mathcal{I} \models \alpha$, if $\mathcal{I}, \mathcal{Z} \models \alpha$ for all possible variable assignments $\mathcal{Z}$ for $\mathcal{I}$. An interpretation $\mathcal{I}$ is a model for a knowledge base if $\mathcal{I}$ satisfies all of its axioms. We say a knowledge base is *satisfiable* if such a model exists. We say that a knowledge base KB entails an axiom α, written as $KB \models \alpha$, if all models of KB satisfy α.

To improve the clarity and understandability of the paper, as well as to simplify our implementation, we make use of a normalization algorithm from [16] and extend it with some new mappings to normalize axioms containing nominal schemas.

Definition 2. *An $\mathcal{ELROV}_n$ knowledge base is* in normal form *if it contains the axiom $\top \times \top \sqsubseteq U$ where U is the universal role, every axiom α not containing a nominal schema is of one of the forms as described in Figure 3, and for every axiom β of the form $C \sqsubseteq D$ containing nominal schemas we have that D is of one of the forms as descriptied in Figure 4.*

Note that we normalize only the right-hand sides of general class inclusion axioms. Since a nominal schema may occur in many places within one axiom, a normalization of axioms following the usual approach of replacing subclasses by new class names and adding additional axioms (this is called *folding* in logic programming terminology) is not possible in general, unless nominal schemas were first grounded to nominals. However, as discussed in the introduction, such up-front grounding results in the general case in a significant increase of the size of the knowledge base, which cannot be handled by existing reasoners.

$$
\begin{array}{rr}
C \sqsubseteq D & C \sqsubseteq \{a\} \\
C \sqcap D \sqsubseteq E & C \sqsubseteq \bot \\
C \sqsubseteq \exists R.D & \top \sqsubseteq C \\
\exists R.C \sqsubseteq D & R \sqsubseteq S \\
C \sqsubseteq \exists R.\text{Self} & R \circ S \sqsubseteq T \\
C \sqsubseteq \exists R.\text{Self} & R \circ S \sqsubseteq T \\
\exists R.\text{Self} \sqsubseteq C & R \sqsubseteq C \times D \\
\{a\} \sqsubseteq C & C \times D \sqsubseteq R
\end{array}
$$

where $\{C, D, E\} \subseteq N_C, \{R, S, T\} \subseteq N_R$, and $a \in N_I$.

Fig. 3. $\mathcal{ELROV}_n$ normal form for nominal-schema-free axioms

In fact, the unavailability of a folding-based normalization procedure is one of the main obstacles in adapting DL reasoning algorithms to nominal schemas, see [13,21]. Our approach presented below works without such a normalization as the underlying DL algorithm is based on Datalog. Our partial normalization of the right-hand-sides of general class inclusions, in fact, is not really required, it just makes our approach easier to read and simplifies correctness arguments.

Proposition 1. *For every $\mathcal{ELROV}_n$ knowledge base KB, an $\mathcal{ELROV}_n$ knowledge base KB' over an extended signature can be computed in linear time such that all axioms in KB' are in normal form, and, for all $\mathcal{ELROV}_n$ axioms α that only use signature symbols from KB, we find that KB $\models \alpha$ if and only if $KB' \models \alpha$.*

Proof. We make use of a normalization algorithm from [16] to normalize all axioms not containing nominal schemas in *KB*. Then, we add the axiom $\top \times \top \sqsubseteq U$ to *KB* and we exhaustively apply the mappings described in Figure 5 to the set of axioms containing nominal schemas in *KB*. Note that by restriction 5 in Definition 1 we have that all axioms of the form $\exists R.D$ appearing on the right-hand side of a a general class inclusion containing nominal schemas contain a nominal schema $\{x\}$ in D, and thus this normalization is always possible.

$$
\begin{array}{rr}
C & \exists R.\{x\} \\
\{y\} & \exists U.(\{x\} \sqcap \{y\}) \\
\exists U.(\{x\} \sqcap C) & \exists U.(\{x\} \sqcap \exists R.\{y\}))
\end{array}
$$

where $C \subseteq N_C, R \subseteq N_R$, and $\{x, y\} \in N_V$.

Fig. 4. $\mathcal{ELROV}_n$ normal form for axioms with nominal schemas

$$A \sqsubseteq C \sqcap D \mapsto \{A \sqsubseteq D, A \sqsubseteq C\}$$
$$A \sqsubseteq E \mapsto \{A \sqsubseteq C_x, C_x \sqsubseteq E\}$$
$$A \sqsubseteq \top \mapsto \emptyset$$
$$A \sqsubseteq \exists R.(\{x\} \sqcap C) \mapsto \{A \sqsubseteq \exists R.\{x\}, A \sqsubseteq \exists U.(\{x\} \sqcap C)\}$$
$$A \sqsubseteq \exists U.(\{x\} \sqcap E) \mapsto \{A \sqsubseteq \exists U.(\{x\} \sqcap C_x), C_x \sqsubseteq E\}$$
$$A \sqsubseteq \exists U.(\{x\} \sqcap \top) \mapsto \emptyset$$
$$A \sqsubseteq \exists U.(\{x\} \sqcap C \sqcap D) \mapsto \{A \sqsubseteq \exists U.(\{x\} \sqcap C), A \sqsubseteq \exists U.(\{x\} \sqcap D)\}$$
$$A \sqsubseteq \exists U.(\{x\} \sqcap \exists R.(\{y\} \sqcap C)) \mapsto \{A \sqsubseteq \exists U.(\{x\} \sqcap \exists R.\{y\}), A \sqsubseteq \exists U.(\{y\} \sqcap C)\}$$

where A, C, D are concept expressions, E is a nominal or expression of the form $\exists R.\mathrm{Self}$ or $\bot$, $R, U \in N_R$ and U is the universal role, $x, y \in N_V$, $c \in N_I$ and C_x is a freshly introduced concept name.

Fig. 5. Normalization of axioms containing nominal schemas

Without loss of generality we assume that all knowledge bases appearing throughout the rest of the paper are in normal form.

3 An Algorithm for $\mathcal{ELROV}_n$

As previously mentioned, our algorithm is based on the materialization calculus K_{inst} presented in [16]. Following this approach, for every $\mathcal{ELROV}_n$ knowledge base KB we will construct a Datalog program P_{KB} that can be regarded as an instance retrieval procedure over KB. The Datalog program P_{KB} consists of two sets of rules P and P^{ns} and a set of facts I, produced as follows.

- P is the set of rules listed in Figure 6—this is independent of the input knowledge base KB.
- $I(KB)$ is the set of facts $I(\alpha)$ produced according to Figure 7 for each α which is a class name, a role name, an individual name, or a nominal-schema-free axiom occurring in KB.
- $P^{ns}(KB)$ is the set of all rules $P^{ns}(\gamma)$ generated from each axiom γ containing nominal schemas. The definition of P^{ns} is given below.

In order to define P^{ns}, we first define the partial functions b and h that map first-order logic axioms to sets of unary and binary predicates. Let α be a general concept inclusion axiom in $\mathcal{ELROV}_n$. Then it is easy to see that $\pi(\alpha)$ can be normalized into an axiom of the form

$$\forall x[\exists y_1 \ldots \exists y_n (\bigwedge b_i) \to \exists z_1 \ldots \exists z_n (\bigwedge h_i)], \tag{29}$$

where all h_i and b_i are unary and binary predicates of the form $R(x, y)$ or $C(x)$ with $R \in N_R$, $C \in N_C$, and x, y are first-order logic variables. Then let $b(\pi(\alpha))$ (respectively, $h(\pi(\alpha))$) be the set of all unary and binary predicates contained in $\bigwedge b_i$ (respectively $\bigwedge h_i$), called the *body* (respectively, *head*) of $\pi(\alpha)$.

$$\mathrm{nom}(x) \mapsto \mathrm{inst}(x, x) \tag{1}$$
$$\mathrm{nom}(x) \wedge \mathrm{triple}(x, v, x) \mapsto \mathrm{self}(x, v) \tag{2}$$
$$\mathrm{top}(z) \wedge \mathrm{inst}(x, z') \mapsto \mathrm{inst}(x, z) \tag{3}$$
$$\mathrm{bot}(z) \wedge \mathrm{inst}(u, z) \wedge \mathrm{inst}(x, z') \wedge \mathrm{cls}(y) \mapsto \mathrm{inst}(x, y) \tag{4}$$
$$\mathrm{subClass}(y, z) \wedge \mathrm{inst}(x, y) \mapsto \mathrm{inst}(x, z) \tag{5}$$
$$\mathrm{subConj}(y_1, y_2, z) \wedge \mathrm{inst}(x, y_1) \wedge \mathrm{inst}(x, y_2) \mapsto \mathrm{inst}(x, z) \tag{6}$$
$$\mathrm{subEx}(v, y, z) \wedge \mathrm{triple}(x, v, x') \wedge \mathrm{inst}(x', y) \mapsto \mathrm{inst}(x, z) \tag{7}$$
$$\mathrm{subEx}(v, y, z) \wedge \mathrm{self}(x, v) \wedge \mathrm{inst}(x, y) \mapsto \mathrm{inst}(x, z) \tag{8}$$
$$\mathrm{supEx}(y, v, z, x') \wedge \mathrm{inst}(x, y) \mapsto \mathrm{triple}(x, v, x') \tag{9}$$
$$\mathrm{supEx}(y, v, z, x') \wedge \mathrm{inst}(x, y) \mapsto \mathrm{inst}(x', z) \tag{10}$$
$$\mathrm{subSelf}(v, z) \wedge \mathrm{self}(x, v) \mapsto \mathrm{inst}(x, z) \tag{11}$$
$$\mathrm{supSelf}(y, v) \wedge \mathrm{inst}(x, y) \mapsto \mathrm{self}(x, v) \tag{12}$$
$$\mathrm{subRole}(v, w) \wedge \mathrm{triple}(x, v, x') \mapsto \mathrm{triple}(x, w, x') \tag{13}$$
$$\mathrm{subRole}(v, w) \wedge \mathrm{self}(x, v) \mapsto \mathrm{self}(x, w) \tag{14}$$
$$\mathrm{subRChain}(u, v, w) \wedge \mathrm{triple}(x, u, x') \wedge \mathrm{triple}(x', v, x'') \mapsto \mathrm{triple}(x, w, x'') \tag{15}$$
$$\mathrm{subRChain}(u, v, w) \wedge \mathrm{self}(x, y) \wedge \mathrm{triple}(x, v, x') \mapsto \mathrm{triple}(x, w, x') \tag{16}$$
$$\mathrm{subRChain}(u, v, w) \wedge \mathrm{triple}(x, u, x') \wedge \mathrm{self}(x', v) \mapsto \mathrm{triple}(x, w, x') \tag{17}$$
$$\mathrm{subRChain}(u, v, w) \wedge \mathrm{self}(x, u) \wedge \mathrm{self}(x, v) \mapsto \mathrm{triple}(x, w, x) \tag{18}$$
$$\mathrm{subProd}(y_1, y_2, w) \wedge \mathrm{inst}(x, y_1) \wedge \mathrm{inst}(x', y_2) \mapsto \mathrm{triple}(x, w, x') \tag{19}$$
$$\mathrm{subProd}(y_1, y_2, w) \wedge \mathrm{inst}(x, y_1) \wedge \mathrm{inst}(x, y_2) \mapsto \mathrm{self}(x, w) \tag{20}$$
$$\mathrm{supProd}(v, z_1, z_2) \wedge \mathrm{triple}(x, v, x') \mapsto \mathrm{inst}(x, z_1) \tag{21}$$
$$\mathrm{supProd}(v, z_1, z_2) \wedge \mathrm{self}(x, v) \mapsto \mathrm{inst}(x, z_1) \tag{22}$$
$$\mathrm{supProd}(v, z_1, z_2) \wedge \mathrm{triple}(x, v, x') \mapsto \mathrm{inst}(x', z_2) \tag{23}$$
$$\mathrm{supProd}(v, z_1, z_2) \wedge \mathrm{self}(x, v) \mapsto \mathrm{inst}(x, z_2) \tag{24}$$
$$\mathrm{inst}(x, y) \wedge \mathrm{nom}(y) \wedge \mathrm{inst}(x, z) \mapsto \mathrm{inst}(y, z) \tag{25}$$
$$\mathrm{inst}(x, y) \wedge \mathrm{nom}(y) \wedge \mathrm{inst}(y, z) \mapsto \mathrm{inst}(x, z) \tag{26}$$
$$\mathrm{inst}(x, y) \wedge \mathrm{nom}(y) \wedge \mathrm{triple}(z, u, x) \mapsto \mathrm{triple}(z, u, y) \tag{27}$$
$$\mathrm{self}(x, y) \mapsto \mathrm{triple}(x, y, x) \tag{28}$$

Fig. 6. Deduction Rules P

$C(a) \mapsto \{\text{subClass}(a, D)\}$	$R(a, b) \mapsto \{\text{subEx}(a, R, b, b)\}$
$\top \sqsubseteq C \mapsto \{\text{top}(C)\}$	$A \sqsubseteq \bot \mapsto \{\text{bot}(A)\}$
$\{a\} \sqsubseteq C \mapsto \{\text{subClass}(a, C)\}$	$A \sqsubseteq \{c\} \mapsto \{\text{subClass(A, c)}\}$
$A \sqsubseteq C \mapsto \{\text{subclass}(A, C)\}$	$A \sqcap B \sqsubseteq C \mapsto \{\text{subConj}(A, B, C)\}$
$\exists R.\text{Self} \sqsubseteq C \mapsto \{\text{subSelf}(R, C)\}$	$A \sqsubseteq \exists R.\text{Self} \mapsto \{\text{supSelf}(A, R)\}$
$\exists R.A \sqsubseteq C \mapsto \{\text{subEx}(R, A, C)\}$	$A \sqsubseteq \exists R.C \mapsto \{\text{supEx}(A, R, B, aux_{A \sqsubseteq \exists R.C})\}$
$R \sqsubseteq T \mapsto \{\text{subRole}(R, T)\}$	$R \circ S \sqsubseteq T \mapsto \{\text{subRChain}(R, S, T)\}$
$R \sqsubseteq C \times D \mapsto \{\text{supProd}(R, C, D)\}$	$C \times D \sqsubseteq R \mapsto \{\text{subProd}(C, D, R)\}$
$A \in N_C \mapsto \{\text{cls}(A)\}$	$a \in N_I \mapsto \{\text{nom}(a)\}$
$R \in N_R \mapsto \{\text{rol}(R)\}$	

Fig. 7. Input Translation I

Definition 3. *Given an $\mathcal{ELROV}_n$ axiom $\alpha = A \sqsubseteq B$ (where A and B are concept expressions) that contains nominal schemas, we now define $P^{ns}(\alpha)$ as follows. Let B_α be the set of Datalog atoms containing*

- *$triple(x, R, y)$ for every $R(x, y) \in b(\pi(\alpha))$,*
- *$inst(x, C)$ for every $C(x) \in b(\pi(\alpha))$,*
- *$inst(x, t)$ for every $x \approx t \in b(\pi(\alpha))$ with $t \in N_I \cup N_V$,*
- *$nom(v)$ for every $x \approx v \in b(\pi(\alpha)) \cup h(\pi(\alpha))$ with $v \in N_V$.*

Furthermore, let H_α be the set of Datalog atoms containing

- *$inst(x, C)$ if $h(\alpha) = \{C(x)\}$,*
- *$triple(x, R, t)$ if $h(\alpha) = \{R(x, y), y \approx t\}$ and $t \in N_V$,*
- *$inst(x, t)$ if $h(\alpha) = \{x \approx t\}$,*
- *$inst(u, t)$ and $inst(t, u)$ if $h(\alpha) = \{U(x, y), y \approx t, y \approx u\}$,*
- *$inst(t, C)$ if $h(\alpha) = \{U(x, y), y \approx t, C(y)\}$, and*
- *$triple(t, R, u)$ if $h(\alpha) = \{U(x, y), y \approx t, R(y, z), z \approx u\}$.*

We define

$$P^{ns}(\alpha) = \bigwedge B_i \to \bigwedge H_i$$

for all Datalog atoms $B_i \in B_\alpha$ and $H_i \in H_\alpha$.

Note that for every nominal schema v in axiom α we include the Datalog atom $nom(v')$ in the body of the Datalog rule $P^{ns}(\alpha)$, which essentially restricts the variable to named individuals (see Figure 7). Note that this precisely corresponds to the semantics of nominal schemas, which may only represent named individuals.

We give an example of an $\mathcal{ELROV}_n$ axiom α and the corresponding Datalog rule $P^{ns}(\alpha)$. Let α be the axiom

$$\exists R.\{v\} \sqcap \exists S.\{v\} \sqsubseteq \exists T.\{v\},$$

where v is a nominal schema. Then we obtain

$$\begin{aligned}\pi(\alpha) =& \forall x[\pi_x(\exists R.\{v\} \sqcap \exists S.\{v\}) \to \pi_x(\exists T.\{v\})]\\ =& \forall x[\pi_x(\exists R.\{v\}) \wedge \pi_x(\exists S.\{v\}) \to \exists y[T(x,y) \wedge \pi_y(\{v\})]]\\ =& \forall x[\exists z[R(x,z) \wedge z \approx v] \wedge \exists w[R(x,w) \wedge w \approx v] \to \exists y[T(x,y) \wedge y \approx v)]]\end{aligned}$$

and thus

$$P^{ns}(\alpha) = \mathit{triple}(x,R,z) \wedge \mathit{inst}(z,v) \wedge \mathit{triple}(x,S,w) \wedge \mathit{inst}(w,v) \wedge \mathit{nom}(v) \mapsto \mathit{triple}(x,T,v)$$

Finally, we observe the following result.

Theorem 1 (Correctness). *Let KB be an $\mathcal{ELROV}_n$ knowledge base and let $P_{KB} = I(KB) \cup P \cup P^{ns}(KB)$. We have that $P_{KB} \models \mathit{inst}(a,C)$ if and only if $KB \models C(a)$ for all $C \in N_C$ and $a \in N_I$. Furthermore, execution of P_{KB} terminates in polynomial time with respect to the size of KB.*

Proof. The formal proof of Theorem 1 can be found in the appendix of an extended technical report available from
`http://www.pascal-hitzler.de/pub/elrov13.pdf`. The proof is in fact an adaptation of the arguments used in [16].

4 Implementation and Evaluation

In the technical report [4], we had already given a preliminary report on some experiments using full grounding (there called *naive* grounding), and we give a summary here. These experiments were performed by adding some axioms with nominal schemas to some ontologies from the TONES repository[3], some slightly modified. We then removed the nominal schemas through full grounding, and ran the resulting ontologies through Pellet [20]. This round of testing was performed using a 64-bit Windows 7 computer with an Intel(R) Core(TM) i5 CPU processor. A Java JDK 1.5 version was used allocating 3GB as the minimum for the Java heap and 3.5GB as the maximum for each experiment.

In order to understand the effect of several nominal schemas on the runtime, we added three different types of axioms to the ontologies, (1) an axiom with

[3] `http://owl.cs.manchester.ac.uk/repository/`
[4] `http://www.mindswap.org/ontologies/family.owl`
[5] `http://sweet.jpl.nasa.gov/1.1/data.owl`
[6] `http://www.ordnancesurvey.co.uk/ontology/BuildingsAndPlaces/v1.1/BuildingsAndPlaces.owl`
[7] `http://www.berkeleybop.org/ontologies/obo-all/worm_phenotype_xp/worm_phenotype_xp.obo`
[8] `http://reliant.teknowledge.com/DAML/Transportation.owl`
[9] `http://www.co-ode.org/roberts/family-tree.owl`
[10] `http://reliant.teknowledge.com/DAML/Economy.owl`

Table 1. Ontologies used in experiments for full grounding and full grounding experimental results. Ind: individuals, Ann: Annotation Properties, Data: Data Properties, Obj: Object Properties. For the remaining entries, the first listed number is load time, the second is reasoning time, both in seconds. OOM indicates *out of memory.*

Ont	Ind	Classes	Ann	Data	Obj	no ns		1 ns		2 ns		3 ns	
Fam[4]	5	4	0	1	11	0.01	0.00	0.01	0.00	0.01	0.00	0.04	0.02
Swe[5]	22	189	1	6	25	3.58	0.08	3.73	0.07	3.85	0.10	10.86	1.11
Bui[6]	42	686	15	0	24	1.70	0.16	1.50	0.15	2.75	0.26	74.00	6.68
Wor[7]	80	1842	6	0	31	0.11	0.04	0.12	0.05	1.10	0.55	11,832.00	315.00
Tra[8]	183	445	2	4	89	0.05	0.03	0.05	0.02	5.66	1.76	OOM	OOM
FTr[9]	368	22	2	6	52	0.03	4.28	0.05	5.32	35.53	42.73	OOM	OOM
Eco[10]	482	339	2	8	45	0.04	0.24	0.07	0.02	56.59	13.67	OOM	OOM

Table 2. More full grounding experimental results, the first listed number is load time, the second is reasoning time, both in seconds. OOM indicates *out of memory.*

Ontology	Individuals	no ns		20×1 ns		10×2 ns	
Fam	5	0.01	0.00	0.01	0.00	0.02	0.01
Swe	22	3.58	0.08	3.42	0.08	3.73	0.28
Bui	42	2.70	0.16	2.69	0.25	5.70	3.21
Wor	80	0.11	0.04	0.23	0.28	12.42	6.88
Tra	183	0.05	0.03	0.33	0.15	107.57	43.63
FTr	368	0.03	4.28	0.52	11.33	OOM	OOM
Eco	482	0.04	0.24	0.65	0.30	OOM	OOM

only one nominal schema, (2) an axiom with two different nominal schemas, and (3) an axiom with three different nominal schemas. An example for an added axiom is

$$\exists \text{prop1}.\{v1\} \sqcap \exists \text{prop2}.\{v1\} \sqcap \exists \text{prop3}.\{v2\} \sqcap \exists \text{prop4}.\{v2\} \sqsubseteq \text{Class1}.$$

Since the blow-up obtained from full grounding is exponential in the number of nominal schemas, this is already the limit we can manage with non-trivial ontologies—as can be seen from the results presented in Table 1.

We then investigated the impact of *several* axioms with nominal schemas on the performance, by adding 20 axioms with one nominal schema, respectively 10 axioms with 2 nominal schemas. The results can be found in Table 2.

The experiments just given indicate that full grounding is limited to a maximum of two or three nominal schemas per axiom, even for relatively small ontologies. This insight provides the baseline against which to evaluate our algorithm. Our goal is to show that axioms with more nominal schemas can be handled with reasonable efficiency.

In order to test our approach, we implemented it as front-end to the Java-based Datalog reasoner IRIS[11] [3]. We also used suitable ontologies from the TONES repository, see Table 3 for some basic metrics.

Table 3. Evaluation ontologies for our algorithm

Ontology	Classes	Annotation P.	Data P.	Object P.
Rex[12]	552	10	0	6
Spatial[13]	106	13	0	13
Xenopus[14]	710	19	0	5

Since these ontologies do not contain individuals, but the algorithm requires individuals to fire the rules, we created three different sets of dummy individuals of varying size (100, 1000, and 10000 individuals) which were randomly assigned to concepts and roles. We then added an axiom of the form

$$\prod_{1 \leq i \leq k} (\exists R_i.\{z_i\}) \sqsubseteq C,$$

where k ranged from 1 to 5 in different tests, to evaluate the effect of axioms with different numbers of nominal schemas.

To obtain a comparison with the full grounding approach, we ran each ontology through two tests. The first used our algorithm, with IRIS as underlying Datalog reasoner. The second test did first perform a full grounding, with subsequent processing by our algorithm. Note that in this case our algorithm essentially coincides with the one reported in [16], thus providing a fair comparison between our approach and the full grounding approach. In the second case, the final reasoning was also done using IRIS. We ran the experiments on a laptop with a 2.4GHz Intel CoreTM i7-3630QM processor and 8GB RAM operated by Windows 7 64-bit system with Java VM v.1.7.0. We set a time-out of 1 hour and a Java heap space of 1GB.

Results are listed in Table 4. First note that the full grounding approach performed similarly to the results reported above using Pellet, i.e., we hit a limit with 2 or 3 nominal schemas per axiom. Using our algorithm, however, the number of nominal schemas per axioms had almost no effect on the runtime, thus indicating that our approach performs very well indeed.

---

[11] `http://iris-reasoner.org/`

[12] `http://obo.cvs.sourceforge.net/checkout/obo/obo/ontology/physicochemical/rex.obo`

[13] `http://obo.cvs.sourceforge.net/checkout/obo/obo/ontology/anatomy/caro/spatial.obo`

[14] `http://obo.cvs.sourceforge.net/checkout/obo/obo/ontology/anatomy/gross_anatomy/animal_gross_anatomy/frog/xenopus_anatomy.obo`

Table 4. Evaluation, IRIS reasoning time listed only (no pre-processing, no load time), in ms. The "No ns" column refers to the running with no nominal schemas. Times in brackets are for full grounding, for comparison. If not listed, full grounding was OOM (Out of Memory)

Ontology	Individuals	no ns	1 ns	2 ns	3 ns	4 ns	5 ns
Rex (full ground.)	100	263	263 (321)	267 (972)	273	275	259
	1000	480	518 (1753)	537 (OOM)	538	545	552
	10000	2904	2901 (133179)	3120 (OOM)	3165	3192	3296
Spatial (full ground.)	100	22	191 (222)	201 (1163)	198	202	207
	1000	134	417 (1392)	415 (OOM)	421	431	432
	10000	1322	1792 (96437)	1817 (OOM)	1915	1888	1997
Xenopus (full ground.)	100	62	332 (383)	284 (1629)	311	288	280
	1000	193	538 (4751)	440 (OOM)	430	456	475
	10000	1771	2119 (319013)	1843 (OOM)	1886	2038	2102

5 Conclusions and Future Work

In this paper, we have introduced, for the first time, an algorithm for reasoning over a nominal-schema-extended description logic which scales well. We have obtained this result by modifying an existing algorithm for $\mathcal{EL}^{++}$. While the algorithm modification itself is not overly sophisticated, it has taken considerable time (namely three years since the introduction of nominal schemas in [18]) and several previous unsuccessful efforts (such as [13,21]) to come up with this first approach. The main contribution of this paper is thus to show that a reasonable algorithmization of nominal-schema-extended description logics is feasible at all.

Of course, we consider this work only to be a first step towards the development of algorithms for nominal-schema-extended description logics. It is reasonable to expect that the approach presented herein will in some way extend to other nominal-schema-extended Horn DLs, however major modifications will be required in order to step outside the $\mathcal{EL}$ family of description logics. It can also be expected that adaptations of tableaux or resolution-based algorithms are possible, although the initial efforts mentioned above were only of limited value. New ideas may be required for further advances.

References

1. 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)
2. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope. In: Proc. 19th Int. Joint Conf. on Artificial Intelligence (IJCAI-05). pp. 364–369. Morgan-Kaufmann Publishers, Edinburgh, UK (2005)

3. Bishop, B., Fischer, F.: IRIS – Integrated Rule Inference System. In: van Harmelen, F., Herzig, A., Hitzler, P., Lin, Z., Piskac, R., Qi, G. (eds.) ARea2008, Workshop on Advancing Reasoning on the Web: Scalability and Commonsense. Proceedings of the Workshop on Advancing Reasoning on the Web: Scalability and Commonsense Tenerife, Spain, June 2, 2008. CEUR Wrokshop Proceedings, vol. 350. CEUR-WS.org (2008)
4. Carral Martínez, D., Krisnadhi, A., Maier, F., Sengupta, K., Hitzler, P.: Reconciling OWL and rules. Tech. rep., Kno.e.sis Center, Wright State University, Dayton, Ohio, U.S.A. (2011), available from http://www.pascal-hitzler.de/
5. Hitzler, P., Parsia, B.: Ontologies and rules. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, pp. 111–132. Springer, 2 edn. (2009)
6. Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S. (eds.): OWL 2 Web Ontology Language: Primer. W3C Recommendation (27 October 2009), available at http://www.w3.org/TR/owl2-primer/
7. Hitzler, P., Krötzsch, M., Rudolph, S.: Foundations of Semantic Web Technologies. Chapman & Hall/CRC (2009)
8. Horrocks, I., Patel-Schneider, P., Boley, H., Tabet, S., Grosof, B., Dean, M.: SWRL: A Semantic Web Rule Language. W3C Member Submission (21 May 2004), see http://www.w3.org/Submission/SWRL/
9. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2006). pp. 57–67. AAAI Press (2006)
10. Kifer, M., Boley, H. (eds.): RIF Overview (Second Edition). W3C Working Group Note (5 February 2013), available at http://www.w3.org/TR/rif-overview/
11. Knorr, M., Carral Martínez, D., Hitzler, P., Krisnadhi, A., Maier, F., Wang, C.: Recent advances in integrating OWL and rules (technical communication). In: Krötzsch, M., Straccia, U. (eds.) Web Reasoning and Rule Systems, 6th International Conference, RR2012, Vienna, Austria, September 10-12, 2012, Proceedings. Lecture Notes in Computer Science, vol. 7497, pp. 225–228. Springer, Heidelberg (2012)
12. Knorr, M., Hitzler, P., Maier, F.: Reconciling OWL and non-monotonic rules for the Semantic Web. In: De Raedt, L., Bessiere, C., Dubois, D., Doherty, P., Frasconi, P., Heintz, F., Lucas, P. (eds.) ECAI 2012, 20th European Conference on Artificial Intelligence, 27-31 August 2012, Montpellier, France. Frontiers in Artificial Intelligence and Applications, vol. 242, pp. 474–479. IOS Press, Amsterdam (2012)
13. Krisnadhi, A., Hitzler, P.: A tableau algorithm for description logics with nominal schema. In: Krötzsch, M., Straccia, U. (eds.) Web Reasoning and Rule Systems – 6th International Conference, RR 2012, Vienna, Austria, September 10-12, 2012. Proceedings. Lecture Notes in Computer Science, vol. 7497, pp. 234–237. Springer (2012)
14. Krisnadhi, A., Maier, F., Hitzler, P.: OWL and rules. In: Polleres, A., d'Amato, C., Arenas, M., Handschuh, S., Kroner, P., Ossowski, S., Patel-Schneider, P. (eds.) Reasoning Web. Semantic Technologies for the Web of Data. 7th International Summer School 2011, Galway, Ireland, August 23-27, 2011, Tutorial Lectures. Lecture Notes in Computer Science, vol. 6848, pp. 382–415. Springer, Heidelberg (2011)
15. Krötzsch, M.: Description Logic Rules, Studies on the Semantic Web, vol. 008. IOS Press/AKA (2010)
16. Krötzsch, M.: Efficient inferencing for OWL EL. In: Janhunen, T., Niemelä, I. (eds.) Proc. 12th European Conf. on Logics in Artificial Intelligence (JELIA'10). LNAI, vol. 6341, pp. 234–246. Springer (2010)

17. 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.) Proceedings of the 7th International Semantic Web Conference (ISWC 2008, Karlsruhe, Germany, October 26-30, 2008). Lecture Notes in Computer Science, vol. 5318, pp. 649–664. Springer (2008)
18. Krötzsch, M., Maier, F., Krisnadhi, A.A., Hitzler, P.: A better uncle for OWL: Nominal schemas for integrating rules and ontologies. In: Proceedings of the 20th International Conference on World Wide Web (WWW'11). pp. 645–654. ACM (2011)
19. Motik, B., Sattler, U., Studer, R.: Query answering for OWL DL with rules. J. of Web Semantics 3(1), 41–60 (2005)
20. Sirin, E., Parsia, B., Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Web Semantics: Science, Services and Agents on the World Wide Web 5(2), 51–53 (2007)
21. Wang, C., Hitzler, P.: A resolution procedure for description logics with nominal schemas. In: Takeda, H., Giu, Y., Mizoguchi, R., Kitamura, Y. (eds.) Semantic Technology, Second Joint International Conference, JIST 2012, Nara, Japan, December 2-4, 2012, Proceedings. Lecture Notes in Computer Science, vol. 7774, pp. 1–16. Springer, Heidelberg (2012)

Appendix

A Correctness

Lemma 1 (Soundness). *Let KB be an $\mathcal{ELROV}_n$ knowledge base and $P_{KB} = I(KB) \cup P \cup P^{ns}(KB)$. We have that $P \models inst(a, C)$ if $KB \models C(a)$ where $C \in N_C$ and $a \in N_I$.*

Proof. Our proof relies on the soundness proof presented in [16]. As in the previously mentioned argument, to interpret the atoms derived by P, we define a function k which maps every constant c in our program to a concept $k(c)$. Function k is defined as follows:

- if $c \in N_I$ then $k(c) := \{c\}$,
- if $c = aux_\alpha$ where $\alpha = C \sqsubseteq \exists R.D$, then $k(c) := D \sqcap \exists R^-.C$.

The concept used in the second case includes an inverse role, a constructor not included in $\mathcal{ELROV}_n$. The semantics of a concept of the form $(\exists R^-.C)$ are defined as follows $(\exists R^-.C)^\mathcal{I} = \{y | \langle x, y\rangle \in R^\mathcal{I}, x \in C^\mathcal{I}\}$. As in [16], we assign meaning to ground atoms of $P \cup P^{ns}$ as follows:

- inst(c, A) with $A \in N_C$: $KB \models k(c) \sqsubseteq A$,
- inst(c, d) with $d \in N_I$: $KB \models k(c) \sqsubseteq d$,
- triple(c, R, d) : $KB \models k(c) \sqsubseteq \exists R.k(d)$, and
- self(c, R) : $KB \models k(c) \sqsubseteq \exists R.$Self,

and in each case KB implies that $k(c)$ is necessarily non-empty. We have that the second term in any derived *inst* predicate must either in N_C or N_I, so the above covers all cases. We claim that an atom is entailed by P only if the corresponding semantic conditions are satised by *KB*. In particular, this proves the overall claim of soundness.

For most of the rules in P, it is easy to apply the induction hypothesis immediately to the body atoms to obtain the desired conclusion in combination with the axioms of *KB* that the involved EDB atoms encode. This covers all rules but (23). To prove soundness of rule (23) we extend the induction hypothesis argument presented in [16]. Note that this is necessary, as we have added a new rule (28) to the set P.

Let us first consider the situation. We have that if $P \models$ supProd$(R, C, D) \wedge$ triple(c, R, d) then this implies that $KB \models k(d) \sqsubseteq D$. This conclusion is immediate if we have that $d = a$ for some $a \in N_I$ but we need to include an induction based argument to also prove that this is the case when $d = aux_\alpha$. Thus, we assume that $d = aux_{B \sqsubseteq \exists V.A}$ and hence $k(d) = \exists V^-.B \sqcap A$ and prove the following claim: if $KB \models R \sqsubseteq C \times D$ and $P \models$ triple(c, R, d) then $k(d) \sqsubseteq D$. We proceed by induction on the proof tree of $P \models$ triple(c, R, d). Most of the rules here are already considered in the soundness proof in [16] and hence, when some argument is repeated we just pointed to the aforementioned publication.

- Rule (9): analogous to rule (9) in [16].

- Rule (13): analogous to rule (13) in [16].
- Rule (15): analogous to rule (15) in [16].
- Rule (16): analogous to rule (16) in [16].
- Rule (17): analogous to rule (17) in [16].
- Rule (18): analogous to rule (18) in [16].
- Rule (19): analogous to rule (21) in [16]
- Rule (28): then we have that self(c, R) which by the induction hypothesis implies that $KB \models k(x) \sqsubseteq \exists R.\text{Self}$ and $d = c$. Hence, it is the case that $KB \models k(d) \sqsubseteq \exists R^{-}.\top$ and $KB \models k(d) \sqsubseteq D$.

Note that the induction argument presented in that publication to prove soundness of rule (23) does not need to be extended for the Datalog rules produced by $P^{ns}(KB)$. Let α be an $\mathcal{ELROV}_n$ axiom containing nominal schemas, by the definition of function $P^{ns}(KB)$ we have that for every Datalog atom triple(x, R, y) produced as a consequence of triggering a rule $P^{ns}(\alpha)$, where α is an $\mathcal{ELROV}_n$ axiom, $y = a$ for some $a \in N_I$.

We now proceed to show that inferences produced by Datalog rules of the form $P^{ns}(\alpha)$ are indeed sound. As defined by our normal form, rules $P^{ns}(\alpha)$ may be of six different types. We evaluate this six different types of productions that may appear in P and verify that all of them are indeed sound.

- Fact $\mathit{inst}(x, C)$ produced by datalog rule $P^{ns}(\alpha) = \bigwedge B_i \mapsto \mathit{inst}(x, C)$ where $\alpha = D \sqsubseteq C$: then, by the definition of P^{ns},$KB \models c(x) \sqsubseteq D$ for some grounding of the nominal schemas in D and hence $KB \models c(x) \sqsubseteq C$.
- Fact $\mathit{triple}(x, R, a)$ produced by $P^{ns}(\alpha) = \bigwedge B_i \mapsto \mathit{triple}(x, R, t)$: then $\alpha = D \sqsubseteq \exists R.\{t\} \in KB$ and $t \in N_V$. By the definition of P^{ns}, we have that $KB \models c(x) \sqsubseteq D$ if every occurrence of nominal schema $\{t\}$ is grounded to named individual a for some grounding of the rest of the nominal schemas in D. Henceforth we have that $KB \models c(x) \sqsubseteq \exists R.\{t\}$.
- Fact $\mathit{inst}(x, a)$ produced by $P^{ns}(\alpha) = \bigwedge B_i \mapsto \mathit{inst}(x, t)$ where $\alpha = D \sqsubseteq \{t\} \in KB$ and $t \in N_V$. By the definition of P^{ns}, we have that $KB \models c(x) \sqsubseteq D$ if every occurrence of nominal schema $\{t\}$ is grounded to named individual a for some grounding of the rest of the nominal schemas in D. Henceforth, we have that $KB \models c(x) \sqsubseteq \{a\}$.
- Facts $\mathit{inst}(a, b)$ and $\mathit{inst}(b, a)$ produced by $P^{ns}(\alpha) = \bigwedge B_i \mapsto \mathit{inst}(u, t) \wedge \mathit{inst}(t, u)$ where $\alpha = D \sqsubseteq \exists U.(\{t\} \sqcap \{u\})$ and $t, u \in N_V$. Then, by definition of P^{ns} we have that D is non-empty for some grounding of the nominal schemas where occurrences of $\{t\}$ and $\{u\}$ are grounded to named individuals a and b. Consequently, $KB \models \mathit{inst}(a, b), \mathit{inst}(b, a)$. Note that, by the argument made in [16], we have that $c(x)$ is always non-empty if we have that inst(x, C) or triple(x, R, y). This is the case, as every rule $P^{ns}(\alpha)$ contains at least some of these predicates in the body.
- Fact $\mathit{inst}(a, C)$ produced by $P^{ns}(\alpha) = \bigwedge B_i \mapsto \mathit{inst}(t, C)$ where $\alpha = D \sqsubseteq \exists U.(\{t\} \sqcap C) \in KB$ and $t \in N_V$. Then, by definition of P^{ns} we have that D is non-empty for some grounding of the nominal schemas where occurrences of $\{t\}$ are grounded to named individual a and hence $KB \models \mathit{inst}(a, C)$.

– Fact *triple*(a, R, b) produced by $P^{ns}(\alpha) = \bigwedge B_i \mapsto triple(t, R, u)$ where $\alpha = D \sqsubseteq \exists U.(\{t\} \sqcap \exists R.\{u\}) \in KB$ and $t \in N_V$. Then, by definition of P^{ns} imply that D is non-empty for some grounding where occurrences of $\{t\}$ and $\{u\}$ are grounded to named individuals $\{a\}$ and $\{b\}$ and thus $KB \models inst(a, C)$.

Theorem 2 (Completeness). *Let KB be an* $\mathcal{ELROV}_n$ *knowledge base and* $P_{KB} = I(KB) \cup P \cup P^{ns}(KB)$. *We have that it* $KB \models C(a)$ *then* $P \models inst(a, C)$ *where* $C \in N_C$ *and* $a \in N_I$.

Proof. The proof extends on the completeness argument used in [16]. More precisely, we extend the construction of the model $\mathcal{J}$ from [16], extending the bullet list presented therein.

– α. We have that $J \models P^{ns}(\alpha)$, where $\alpha = C \sqsubseteq D$ with C containing some nominal schemas. If $d^{\mathcal{I}} \in C$, then there is some grounding for the nominal schemas in C that will trigger the execution of rule $P^{ns}(\alpha)$ and produce a set of facts that imply $d^{\mathcal{I}} \in C$ for that specific grounding of the nominal schemas in the axiom.

Note that the addition of rule (28) to the program means that we only have to check for the *triple* predicates in rules of the form $P^{ns}(\alpha)$, without checking for combinations of the self predicate.

Lemma 2 (Termination). *Let KB be an* $\mathcal{ELROV}_n$ *knowledge base and* $P_{KB} = I(KB) \cup P \cup P^{ns}(KB)$. *Execution of P terminates in polynomial time with respect to the original size of KB.*

Proof. The number of nominal schemas per $\mathcal{ELROV}_n$axiom α and free variables in $P^{ns}(\alpha)$ is bounded by n, thus there is a global bound on the number of variables per rule in KB. Datalog is polynominal in this case.