A Resolution Procedure for Description Logics with Nominal Schema

Cong Wang and Pascal Hitzler

Kno.e.sis Center, Wright State University, Dayton OH 45435, USA
{cong,pascal}@knoesis.org

Abstract. We present a polynomial resolution-based decision procedure for the recently introduced description logic $\mathcal{ELHOV}_n(\sqcap)$, which features nominal schemas as new language construct. Our algorithm is based on ordered resolution and positive superposition, together with a lifting lemma. In contrast to previous work on resolution for description logics, we have to overcome the fact that $\mathcal{ELHOV}_n(\sqcap)$ does not allow for a normalization resulting in clauses of globally limited size.

1 Introduction

Description Logic (DL) and rule-based formalism are two prominent paradigms for Knowledge Representation. Although both paradigms are based in classical logic, they provide different expressivity and neither of them contains the other one, i.e. there exist axioms in DL which are not expressible in the rules paradigm and viceversa. Despite significant research efforts [4,5,17], many integrations of the two paradigms lead to undecidability (see [14] for a survey).

Currently, the most notable language in DLs family is the W3C recommendation Web Ontology Language (OWL[1]). OWL can express many rules (see [14]), but it cannot express many others, such as

$$hasParent(x, z) \wedge hasParent(x, y) \wedge married(y, z) \rightarrow C(x) \quad (1)$$

which defines a class C of children whose parents are married.

One idea for retaining decidability is to restrict the applicability of rules to *named individuals*. Rules that are understood in this sense are called *DL-safe rules*, and the combination between OWL DL and DL-safe rules is decidable [21].

Very recently, this idea is found to be able to carry further to description logic paradigm. *Nominal schemas,* as a new element of description logic syntax construct, was introduced in this sense [16]. It does not only further generalize the notion of DL-safety, but also enables to express the DL-safe rules within the description logic syntax. Using nominal schemas, rule (1) could be represented as:

$$\exists hasParent.\{z\} \sqcap \exists hasParent.\exists married.\{z\} \sqsubseteq C \quad (2)$$

[1] http://www.w3.org/TR/owl2-overview/

The expression $\{z\}$ in (2) is a nominal schema, which is a variable that only binds with known individuals in a knowledge base and the binding is the same for all occurrences of the same nominal schema in an axiom.

Consequently, a new description logic $\mathcal{SROIQV}_n$ was introduced, which indeed is a extension of OWL 2 DL $\mathcal{SROIQ}$ with nominal schemas $\mathcal{V}_n$. It is decidable and has the same worst-case complexity as $\mathcal{SROIQ}$ [16]. $\mathcal{SROELV}_n$ is a tractable fragment of $\mathcal{SROIQ}$. And extending $\mathcal{SROELV}_n$ with role conjunction on simple roles and concept product would not change its worst-case complexity [24]. The importance of $\mathcal{SROELV}_n(\sqcap_s, \times)$ is that it can incorporate OWL EL and OWL RL (two tractable fragments of OWL 2), and to allow restricted semantic interaction between the two. Also, it is more easy for ontology modelers to write rules in OWL syntax.

Although reasoning for description logic with nominal schema is theoretically feasible, the simple experiment in [19] shows that the naive approach, based on *full grounding* nominal schemas to all known individuals, is extremely slow. Therefore, it is really necessary to design a *smarter* algorithm. One idea is to ground nominal schemas in a more intelligent way, e.g. intelligent grounding, which is quite well known in *Answer Set Programming* (ASP) field [23]. In [13], the authors applied this strategy on $\mathcal{ALCO}$ with nominal schema, but it needs a very good heuristics for grounding choices.

Another idea is to find a procedure that could do reasoning without grounding. We apply this idea in the paper by using resolution procedure with a lifting lemma. In this paper, we restrict $\mathcal{SROELV}_n(\sqcap_s, \times)$ to $\mathcal{ELHOV}_n(\sqcap)$, which disallows self role, complex role inclusion (role chain) and concept product, but allows role conjunction even for complex roles. The reason of this restriction will be discussed in Section 6. We provide a tractable resolution procedure for $\mathcal{ELHO}(\sqcap)$, then show that the algorithm can also apply for $\mathcal{ELHOV}_n(\sqcap)$ via a lifting lemma.

The structure of this paper is as follows. Section 2 describes some preliminaries of description logic and resolution procedure. Section 3 presents a tractable, sound and complete resolution procedure for $\mathcal{ELHO}(\sqcap)$. Section 4 extends the algorithm to deal with $\mathcal{ELHOV}_n(\sqcap)$. We will provide an example to illustrate the resolution procedure in Section 5 and then briefly discuss some related works in Section 6. Finally we conclude.

2 Preliminaries

2.1 Description Logic

We start by introducing the description logic $\mathcal{ELHOV}_n(\sqcap)$.

A signature of $\mathcal{ELHOV}_n(\sqcap)$ is a tuple $\Sigma = \langle \mathsf{N}_I, \mathsf{N}_C, \mathsf{N}_R, \mathsf{N}_V \rangle$ of mutually disjoint finite sets of *individual names*, *concept names*, *role names*, and *variables*.

Table 1. Semantics of $\mathcal{ELHOV}_n(\sqcap)$

Name	Syntax	Semantics
concept name	A	$A^{\mathcal{I}} \subseteq \triangle^{\mathcal{I}}$
role name	R	$R^{\mathcal{I}} \subseteq \triangle^{\mathcal{I}} \times \triangle^{\mathcal{I}}$
individual name	a	$a^{\mathcal{I}} \in \triangle^{\mathcal{I}}$
variable	x	$\mathcal{Z}(x) \in \triangle^{\mathcal{I}}$
top	$\top$	$\triangle^{\mathcal{I}}$
bottom	$\bot$	$\emptyset$
nominal(schema)	$\{t\}$	$\{t^{\mathcal{I},\mathcal{Z}}\}$
existential restriction	$\exists R.C$	$\{\delta \mid$ there is ϵ with $\langle\delta,\epsilon\rangle \in R^{\mathcal{I},\mathcal{Z}}$ and $\epsilon \in C^{\mathcal{I},\mathcal{Z}}\}$
concept conjunction	$C \sqcap D$	$C^{\mathcal{I},\mathcal{Z}} \cap D^{\mathcal{I},\mathcal{Z}}$
role conjunction	$R \sqcap S$	$R^{\mathcal{I}} \cap S^{\mathcal{I}}$
concept assertion(Abox)	$A(t)$	$t^{\mathcal{I},\mathcal{Z}} \in A^{\mathcal{I},\mathcal{Z}}$
role assertion(ABox)	$R(t,u)$	$\langle t^{\mathcal{I},\mathcal{Z}}, u^{\mathcal{I},\mathcal{Z}}\rangle \in R^{\mathcal{I},\mathcal{Z}}$
TBox axiom	$C \sqsubseteq D$	$C^{\mathcal{I},\mathcal{Z}} \subseteq D^{\mathcal{I},\mathcal{Z}}$
RBox axiom	$R \sqsubseteq S$	$R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$

Definition 1. *The role sets* $\mathbf{R}$ *of* $\mathcal{ELHOV}_n(\sqcap)$ *and the concept sets* $\mathbf{C}$ *of* $\mathcal{ELHOV}_n(\sqcap)$ *are defined by the following grammar:*

$$\mathbf{R} ::= \mathbf{R} \mid \mathbf{R} \sqcap \mathbf{R}$$
$$\mathbf{C} ::= \top \mid \bot \mid \mathsf{N}_C \mid \{\mathsf{N}_I\} \mid \{\mathsf{N}_V\} \mid \mathbf{C} \sqcap \mathbf{C} \mid \exists \mathbf{R}.\mathbf{C}$$

Definition 2. *An interpretation* $\mathcal{I} = (\triangle^{\mathcal{I}}, \cdot^{\mathcal{I}})$ *consists of a domain of discourse* $\triangle^{\mathcal{I}} \neq \emptyset$ *and a function* $\cdot^{\mathcal{I}}$ *which maps* $\mathsf{N}_C, \mathsf{N}_R$, *and* N_I *to elements, sets and relations of* $\triangle^{\mathcal{I}}$ as shown in Table 1. A variable assignment $\mathcal{Z}$ for an interpretation $\mathcal{I}$ is a function $\mathcal{Z} : \mathsf{N}_V \to \triangle^{\mathcal{I}}$ such that for each $v \in \mathsf{N}_V$, $\mathcal{Z}(v) = a^{\mathcal{I}}$ for some $a \in \mathsf{N}_I$. For any interpretation $\mathcal{I}$, assignments $\mathcal{Z}$, and $C_{(i)} \in C$, $R_{(i)} \in \mathsf{N}_R$, $t_{(i)} \in T$, the function $\cdot^{\mathcal{I},\mathcal{Z}}$ is defined as shown in Table 1.

$\mathcal{I}$ and $\mathcal{Z}$ satisfy a $\mathcal{ELHOV}_n(\sqcap)$ axiom α, written $\mathcal{I}, \mathcal{Z} \models \alpha$, if the corresponding condition shown in Table 1 holds. $\mathcal{I}$ satisfies α, written $\mathcal{I} \models \alpha$, if $\mathcal{I}, \mathcal{Z} \models \alpha$ for all variable assignments $\mathcal{Z}$ for $\mathcal{I}$. $\mathcal{I}$ satisfies a $\mathcal{ELHOV}_n(\sqcap)$ knowledge base KB, written $\mathcal{I} \models KB$, if $\mathcal{I} \models \alpha$ for all $\alpha \in KB$, and KB is satisfiable if such an $\mathcal{I}$ exists. The axiom α is entailed by KB, written $KB \models \alpha$, if all models of KB are also models of α.

If α is a $\mathcal{ELHOV}_n(\sqcap)$ axiom, we call $\mathsf{ground}(\alpha)$ as the set of all axioms that can be obtained by uniformly replacing nominal schemas in α with individuals in N_I. Given a $\mathcal{ELHOV}_n(\sqcap)$ knowledge base KB, $\mathsf{ground}(KB) := \bigcup_{\alpha \in KB} \mathsf{ground}(\alpha)$.

Example 1. If a $\mathcal{ELHOV}_n(\sqcap)$ KB contains $\exists R.\{z_1\} \sqcap A \sqsubseteq \exists S.\{z_2\}$ and known individuals a and b. Then, $\mathsf{ground}(KB) = \{\exists R.\{a\} \sqcap A \sqsubseteq \exists S.\{a\}, \exists R.\{a\} \sqcap A \sqsubseteq \exists S.\{b\}, \exists R.\{b\} \sqcap A \sqsubseteq \exists S.\{a\}, \exists R.\{b\} \sqcap A \sqsubseteq \exists S.\{b\}\}$.

Table 2. Translating $\mathcal{ELHOV}_n(\sqcap)$ into First Order Logic

Translating Concepts into FOL
$\pi_x(\bot) = \bot$
$\pi_x(\top) = \top$
$\pi_x(A) = A(x)$
$\pi_x(C \sqcap D) = \pi_x(C) \wedge \pi_x(D)$
$\pi_x(\exists R.C) = \exists y.[R(x,y) \wedge \pi_y(C)]$
$\pi_x(\{z\}) = x \approx z$
Translating TBox Axioms without nominal schemas into FOL
$\pi(C \sqsubseteq D) = \forall x : [\pi_x(C) \rightarrow \pi_x(D)]$
$\pi(R \sqsubseteq S) = \forall x \forall y : [R(x,y) \rightarrow S(x,y)]$
$\pi(R \sqcap S \sqsubseteq T) = \forall x \forall y : [R(x,y) \wedge S(x,y) \rightarrow T(x,y)]$
Translating ABox Axioms without nominal schemas into FOL
$\pi(C(a)) = C(a)$
$\pi(R(a,b)) = R(a,b)$
Translating Axioms containing nominal schemas into FOL
$\pi(C \sqsubseteq D) = \forall x \forall z_1 \ldots \forall z_n : [O(z_1) \wedge \ldots \wedge O(z_n) \rightarrow [\pi_x(C) \rightarrow \pi_x(D)]]$
where $z_1, \ldots, z_n$ are variables in C or D (*)
Translating KB into FOL
$\pi(KB) = \bigwedge_{\alpha \in KB_T \cup KB_A \cup KB_R} \pi(\alpha)$

2.2 Translating $\mathcal{ELHOV}_n(\sqcap)$ into First Order Logic

The first step in deciding satisfiability by resolution procedure is to transform KB into a set of clauses in first order logic. We apply the well-known FOL translation for DLs in [6], and also show how to translate nominal schemas into first order logic.

Definition 3. *Let $O(x) = x \approx a_1 \vee x \approx a_2 \vee \ldots \vee x \approx a_n$ be a first order logic predicate symbol, where $a_k (1 \leq j \leq n) \in \mathsf{N}_I$. The $\approx$ symbol refers to the first order logic equality.*

Table 2 shows such DL-to-FOL translation. The translation from DL to FOL is straightforward based on the semantics of DL. We just make it clear for axioms containing nominal schemas, which the formula (*) in Table 2 states.

From Definition 2, an interpretation $\mathcal{I}$ *satisfies* α, if $\mathcal{I}, \mathcal{Z} \models \alpha$ for all variable assignments $\mathcal{Z}$ for $\mathcal{I}$. That is to say, if $\mathcal{I}$ satisfies an axiom α which contains nominal schema $z_1, \ldots, z_n$, $\mathcal{I}$ must satisfy $\mathsf{ground}(\alpha)$. Suppose $a_j \in \mathsf{N}_I (1 \leqslant j \leqslant m)$ in a KB. $\mathcal{I}$ must satisfy $\bigwedge_{i=1}^{n} \bigwedge_{j=1}^{m} ((z_i \approx a_j) \rightarrow \pi(\alpha))$. From Definition 3, $\bigwedge_{j=1}^{m} [z_i \approx a_j \rightarrow \pi(\alpha)] = O(z_i) \rightarrow \pi(\alpha)$. Therefore, $\bigwedge_{i=1}^{n} (O(z_i) \rightarrow \pi(\alpha)) = O(z_1) \wedge \ldots \wedge O(z_n) \rightarrow \pi(\alpha)$. So, we get the formula (*) $\pi(C \sqsubseteq D) = \forall x \forall z_1 \ldots \forall z_n : (O(z_1) \wedge \ldots \wedge O(z_n) \rightarrow (\pi_x(C) \rightarrow \pi_x(D)))$, where $z_1, \ldots, z_n$ are in C or D.

Example 2. Given two DL axioms containing three nominal schemas z_1, z_2 and z_3, $\alpha = \exists R.\{z\} \sqsubseteq C$, $\beta = \exists R.\{z_1\} \sqcap \exists S.\{z_2\} \sqsubseteq \exists T.\{z_3\}$. The corresponding first order logic translations of α and β according to Table 2 are, $\pi(\alpha) = O(z) \rightarrow$

$[R(x,z) \rightarrow C(x)]$, $\pi(\beta) = [O(z_1) \wedge O(z_2) \wedge O(z_3)] \rightarrow [(R(x,z_1) \wedge S(x,z_2)) \rightarrow T(x,z_3)]$.

2.3 Ordered Resolution

Ordered resolution [3] is a widely used calculus for theorem proving in first order logic. The calculus has two parameters, an admissible ordering $\succ$ on literals and a selection function.

An ordering $\succ$ on literals is admissible if (1) it is well-founded, stable under substitutions, and total on ground literals; (2) $\neg A \succ A$ for all ground atoms A; and (3) $B \succ A$ implies $B \succ \neg A$ for all atoms A and B. A literal L is (strictly) maximal with respect to a clause C if there is no other literal $L^{'} \in C$ such that $(L^{'} \succeq L)L^{'} \succ L$. A literal $L \in C$ is (strictly) maximal in C if and only if L is (strictly) maximal with respect to $C \backslash L$. [6]

A *selection function* S assigns to each clause C a subset of negative literals of C (empty possibly); the literals are said to be *selected* if they are in $S(C)$. No other restrictions are imposed on the selection function, i.e., any arbitrary functions mapping to negative literals are allowed.

With $\mathcal{R}$ we denote the ordered resolution calculus, consisting of the following inference rules, where $D \vee \neg B$ is called the main premise. $C \vee A$ is called the side premise, and $C\sigma \vee D\sigma$ is called conclusion:

Ordered Resolution: $$\frac{C \vee A \quad D \vee \neg B}{C\sigma \vee D\sigma}$$

where (1) $\sigma = mgu(A, B)$, (2) $A\sigma$ is strictly maximal with respect to $C\sigma$, and no literal is selected in $C\sigma \vee A\sigma$, (3) $\neg B\sigma$ is either selected in $D\sigma \vee \neg B\sigma$, or it is maximal with respect to $D\sigma$ and no literal is selected in $D\sigma \vee \neg B\sigma$.

For general FOL, there is another rule needed, called Positive factoring. It resolves two positive literals in one clause. However, since the target logic language in the paper is a Horn logic, such that this rule is not required any more.

2.4 Superposition

Translation $\mathcal{ELHOV}_n(\sqcap)$ into FOL will produce equality symbol. In order to deal with equality, we also need superposition, a calculus for equational theorem proving.

Positive superposition: $$\frac{(C \vee s \approx t) \cdot p \quad (D \vee w \approx v) \cdot p}{(C \vee D \vee w[t]_p \approx v) \cdot \theta}$$

where (i) $\sigma = mgu(sp, wp|_p)$ and $\theta = p\sigma$, (ii) $t\theta \not\preceq s\theta$ and $v\theta \not\succeq w\theta$, (iii) $(s \approx t)\cdot\theta$ is strictly eligible for superposition in $(C \vee s \approx t) \cdot \theta$, (iv) $(w \approx v) \cdot \theta$ is strictly eligible for superposition in $(D \vee w \approx v) \cdot \theta$, (v) $s\theta \approx t\theta \not\succeq w\theta \approx v\theta$, (vi) $w|_p$ is not a variable.

Superposition [20] contains 4 rules, positive superposition, negative superposition, reflexivity resolution and equality factoring. However, due to the preprocess in Section 3, only positive superposition is needed. Since negative superposition and reflexivity resolution need clauses containing $\not\approx$, which will not occur in $\mathcal{ELHOV}_n(\sqcap)$ clauses. Also, since $\mathcal{ELHOV}_n(\sqcap)$ is a Horn logic, therefore,

Table 3. normal forms of $\mathcal{ELHO}(\sqcap)$ axioms

$A \sqsubseteq \bot$	$\bot \sqsubseteq C$	$A \sqsubseteq C$	$A \sqcap B \sqsubseteq C$	$\exists R.A \sqsubseteq C$	$A \sqsubseteq \exists R.B$
$\exists R.\{a\} \sqsubseteq C$	$A \sqsubseteq \exists R.\{a\}$	$A \sqsubseteq \{a\}$	$\{a\} \sqsubseteq A$	$R \sqsubseteq T$	$R \sqcap S \sqsubseteq T$

equality factoring, which requires two positive literals in the premise, cannot be applied.

Ordered resolution and superposition are sound and complete algorithms for first order logic [3]. But, with different settings of the order and selection function, the procedure may be terminated or not.

3 Deciding Satisfiability of $\mathcal{ELHO}(\sqcap)$ by Resolution Procedure

In order to make the resolution procedure simpler, we first eliminate some equality literals such that the clauses contains only positive equality literals. Then we use the well-known structure transformation [20] to get $\mathcal{ELHO}(\sqcap)$ normal forms.

3.1 Eliminating Equality Literals

Superposition rules are designed to deal with equality in saturating first order logic clauses. However, some superposition inferences often make the resolution procedure very complicated. Directly translating of DL to FOL may contain negative equality literals. We use the following equivalent translation to eliminate negative equality literals, such that only positive superposition can be applied.

For DL concepts containing a nominal $\{a\}$, $\pi_x(\exists R.\{a\}) = \exists y.[R(x,y) \wedge y \approx a] = R(x,a)$, $\pi_x(\{a\} \sqcap C) = x \approx a \wedge \pi_x(C) = \pi_a(C)$. For DL axioms containing a nominal $\{a\}$, $\pi_x(\{a\} \sqsubseteq C) = C(a)$. For $C \sqsubseteq \{a\}$, we still directly translate it into FOL, i.e., $\neg C(x) \vee x \approx a$.

Similarly, for DL concepts containing a nominal schema $\{z\}$, $\pi_z(\exists R.\{z\}) = R(x,z)$, $\pi_x(\{z\} \sqcap C) = \pi_z(C)$. For DL axioms containing a nominal schema $\{z\}$, $\pi_x(\{z\} \sqsubseteq C) = O(z) \rightarrow (x \approx z \rightarrow \pi_x(C)) = (O(z) \wedge x \approx z) \rightarrow \pi_x(C) = O(x) \rightarrow \pi_x(C)$. For $C \sqsubseteq \{z\}$, C is either empty or the subconcept of each individual. Without losing generality, we assume that no concept C can be subconcept of each individual. Therefore, C has to be empty, $\pi_x(C \sqsubseteq \{z\}) = \pi_x(C \sqsubseteq \bot)$.

After such transformation, all negative equality literals can be eliminated.

Example 3. For DL axiom $\alpha = \exists R.\{z\} \sqcap \{z\} \sqsubseteq C$, where $\{z\}$ is a nominal schema, $\pi(\alpha) = O(z) \rightarrow [\pi_x(\exists R.\{z\} \sqcap \{z\}) \rightarrow \pi_x(C)] = [O(z) \wedge (x \approx z) \wedge \pi_x(\exists R.\{z\})] \rightarrow \pi_x(C) = \neg O(x) \vee \neg R(x,x) \vee C(x)$.

3.2 Preprocessing

All the $\mathcal{ELHO}(\sqcap)$ axioms can be translated into normal forms in Table 3 in polynomial time using the structure transformation [20]. Table 4 shows all possible

clause types appearing in $\mathcal{ELHO}(\sqcap)$ saturation. We first give the definition of $\Xi(KB)$, which denotes the FOL clause set of a $\mathcal{ELHO}(\sqcap)$ KB.

Definition 4. *The set of clauses $\Xi(KB)$, encoding an $\mathcal{ELHO}(\sqcap)$ knowledge base KB in FOL, is defined as follows:*

- For each ABox or RBox axiom α in KB , $\pi(\alpha) \in \Xi(KB)$.
- For each TBox axiom $C \sqsubseteq D$ in KB , $\pi(C \sqsubseteq D) \in \Xi(KB)$.
- For each TBox axiom $C \equiv D$ in KB , $\pi(C \sqsubseteq D) \in \Xi(KB)$ and $\pi(D \sqsubseteq C) \subseteq \Xi(KB)$.

Theorem 1. *Let KB be an $\mathcal{ELHO}(\sqcap)$ knowledge base. Then, the following claims hold:*

- KB is satisfiable if and only if $\Xi(KB)$ is satisfiable.
- $\Xi(KB)$ can be computed in polynomial time in the $|KB|$.

Proof. From Definition 4, equisatisfiability of KB and $\Xi(KB)$ is trivial to check [6]. All the $\mathcal{ELHO}(\sqcap)$ axioms can be translated into the normal forms of Table 3 in polynomial time, and translating from DL normal forms into first order logic clauses is in polynomial time. Therefore, $\Xi(KB)$ can be computed in time polynomial in the $|KB|$.

3.3 Deciding $\mathcal{ELHO}(\sqcap)$

Now we are ready to show that the resolution procedure for $\mathcal{ELHO}(\sqcap)$ is in time polynomial in $|KB|$.

Definition 5. *Let $\mathcal{R}_{DL}$ denote the ordered resolution calculus $\mathcal{R}$ with positive superposition parameterized as follows:*

- The literal ordering is an admissible ordering $\succ$ such that $f \succ c \succ R \succ A$, for all function symbol f, constant symbol c, binary predicate symbol R and unary predicate symbol A.
- The selection function selects every negative maximal binary literal in each clause.

Next, we enumerate all $\mathcal{R}_{DL}$ inferences between clauses and show that every conclusion is one of clause types of Table 4. With [n, m] $\rightsquigarrow$ [k] we denote an inference between clause type n and m resulting in clause type k, where n, m, k are integers.

Lemma 1. *Each $\mathcal{R}_{DL}$ inference, when applied to $\mathcal{ELHO}(\sqcap)$ -clauses, produces a $\mathcal{ELHO}(\sqcap)$ -clause type in Table 4. The maximum length of each clause is 3. And the number of clauses different up to variable renaming is polynomial in $|KB|$.*

Table 4. $\mathcal{ELHO}(\sqcap)$ -clause types

(1) $\neg A(x)$	(11) $\neg A(x) \vee f(x) \approx a$
(2) $C(x)$	(12) $a \approx b$
(3) $\neg A(x) \vee C(x)$	(13) $\neg R(x,y) \vee S(x,y)$
(4) $\neg A(x) \vee \neg B(x) \vee C(x)$	(14) $\neg R(x,y) \vee \neg S(x,y) \vee T(x,y)$
(5) $\neg R(x,y) \vee \neg A(y) \vee C(x)$	(15) $(\neg)A(a)$
(6) $\neg A(x) \vee R(x, f(x))$	(16) $(\neg)R(a,b)$
(7) $\neg A(x) \vee B(f(x))$	(17) $\neg A(x) \vee \neg R(x, f(x)) \vee S(x, f(x))$
(8) $\neg A(x) \vee R(x,a)$	(18) $\neg A(x) \vee \neg B(f(x)) \vee C(f(x))$
(9) $\neg R(x,a) \vee A(x)$	(19) $\neg A(x) \vee \neg B(f(x)) \vee C(x)$
(10) $\neg A(x) \vee x \approx a$	

Proof. The ordered resolution inferences are possible between the following clauses. $[2,3] \rightsquigarrow [2]$, $[2,4] \rightsquigarrow [3]$. $[6,5] \rightsquigarrow [19]$, $[6,13] \rightsquigarrow [6]$, $[6,14] \rightsquigarrow [17]$. $[7,3] \rightsquigarrow [7]$,$[7,4] \rightsquigarrow [18]$, $[7,10] \rightsquigarrow [11]$, $[7,15] \rightsquigarrow [1]$. $[8,5] \rightsquigarrow [3]$ with unifying x to a, $[8,9] \rightsquigarrow [3]$, $[8,13] \rightsquigarrow [8]$, $[8,14] \rightsquigarrow [17]$ with unifying x to a, $[8,16] \rightsquigarrow [15]$. $[15,1] \rightsquigarrow \bot$, $[15,3] \rightsquigarrow [15]$, $[15,4] \rightsquigarrow [3]$ with unifying x to a, $[15,15] \rightsquigarrow \bot$. $[16,5] \rightsquigarrow [3]$ with unifying x to a and y to b, $[16,9] \rightsquigarrow [15]$, $[16,16] \rightsquigarrow [15]$.

The positive superposition inferences are possible between the following clauses. $[6,11] \rightsquigarrow [8]$, $[7,11] \rightsquigarrow [3]$ with unifying x to a, $[8,12] \rightsquigarrow [8]$. $[10,12] \rightsquigarrow [10]$.

(18) $\neg A(x) \vee \neg B(f(x)) \vee C(f(x))$ can only resolve with clause (7) $\neg A(x) \vee B(f(x))$ or (2) $B(x)$, and produce clause (7) $\neg A(x) \vee C(f(x))$. Since ordered resolution only resolves on maximal literals, thus literal $\neg A(x)$ in clause type (7) can never participate. In addition, due to that every function symbol is unique after skolemization, there is no other clauses in clause type (7) containing $B(f(x))$. Since $\neg B(f(x))$ in (18) has to resolve with $B(f(x))$ or $B(x)$, (18) $\neg A(x) \vee \neg B(f(x)) \vee C(f(x))$ can only resolve with clause $\neg A(x) \vee B(f(x))$ or $B(x)$. Similarly, (17) $\neg A(x) \vee \neg S(x, f(x)) \vee T(x, f(x))$ can only resolve with (6) $\neg A(x) \vee S(x, f(x))$ producing (6) $\neg A(x) \vee T(x, f(x))$, (19) $\neg A(x) \vee \neg B(f(x)) \vee C(x)$ can only resolve with (2) and (3) producing (3).

Any other inferences are not applicable. Therefore, every clause is one of the clause types of Table 4, and the maximum length of clauses is 3. Let c be the number of unary predicates, r the number of binary predicates, f the number of unary function symbols, and i the number of constants in the signature of $\Xi(KB)$. Then, trivially c, r, f and i are linear in $|KB|$. Consider now the maximal $\mathcal{ELHO}(\sqcap)$ -clause of type 5 in Table 4. There are possibly at most rc^2 clauses of type 5. The number of clauses is polynomial in $|KB|$. For other $\mathcal{ELHO}(\sqcap)$ -clause types, the bounds on the length and on the number of clauses can be derived in an analogous way. Therefore, the number of $\mathcal{ELHO}(\sqcap)$ -clauses different up to variable renaming is polynomial in $|KB|$.

Theorem 2. *For an $\mathcal{ELHO}(\sqcap)$ knowledge base KB, saturating $\Xi(KB)$ by $\mathcal{R}_{DL}$ decides satisfiability of KB and runs in time polynomial in $|KB|$.*

Proof. The number of clauses by translating KB is polynomial in $|KB|$. By Lemma 1, the length of every clauses derivable by $\mathcal{R}_{DL}$ is at most 3. And each inference can be performed polynomially. Hence, the saturation terminates in polynomial time. Since $\mathcal{R}_{DL}$ is sound and complete [3], therefore $\mathcal{R}_{DL}$ decides satisfiability of $\Xi(KB)$ in time polynomial in $|KB|$.

4 Deciding Satisfiability of $\mathcal{ELHOV}_n(\sqcap)$ by Resolution Procedure

$\mathcal{ELHOV}_n(\sqcap)$ axioms may contain several nominal schemas or one nominal schema appearing in different positions of an axiom. In such situation, normalization of axioms becomes difficult. For example, $\exists R.(C \sqcap \exists S.\{z\}) \sqsubseteq \exists R.\{z\}$, since $\{z\}$ binds to the same variable, the axiom can not be normalized. $\exists R.(C \sqcap \exists S.\{z\}) \sqsubseteq \exists R.\{z\}$ has to be translated into first order logic directly, which is $\neg O(z) \vee \neg R(x,y) \vee \neg C(y) \vee \neg S(y,z) \vee R(x,z)$. Hence, there are possibly very complex clauses. In order to solve such issue, we use a lifting lemma to show show the resolution procedure for $\mathcal{ELHOV}_n(\sqcap)$ is still polynomial in $|KB|$.

In general, the lifting lemma states that reasoning on a $\mathcal{ELHOV}_n(\sqcap)$ KB without grounding nominal schemas takes fewer steps or produces fewer clauses than reasoning on the grounding KB. Since after grounding all the nominal schemas to nominals, $\mathcal{ELHOV}_n(\sqcap)$ KB becomes actually $\mathcal{ELHO}(\sqcap)$ KB. And since we already showed that the resolution procedure for $\mathcal{ELHO}(\sqcap)$ is polynomial in Theorem 2, therefore reasoning on a $\mathcal{ELHOV}_n(\sqcap)$ KB is still polynomial.

At first, we need to define *safe environment* and $\mathsf{ground}^+(KB)$. The intuition behind of *safe environment* is to restrict the KB with tree-shaped dependencies in order to avoid exponential blow-up (see details in [16]). Then, we show that the size of $\mathsf{ground}^+(KB)$ is polynomial in $|KB$.

Definition 6. *An occurrence of a nominal schema x in a concept C is safe if C has a sub-concept of the form $\{a\} \sqcap \exists R.D$ for some $a \in N_I$, such that D contains the occurrence of $\{x\}$ but no other occurrence of any nominal schema. In this case, $\{a\} \sqcap \exists R.D$ is a safe environment for this occurrence of $\{x\}$. $S(a,x)$ will sometimes be used to denote an expression of the form $\{a\} \sqcap \exists R.D$ within which $\{x\}$ occurs safe.*

Definition 7. *We define a $\mathcal{ELHOV}_n(\sqcap)$ knowledge base $\mathsf{ground}^+(KB)$ as follows. The RBox and ABox of $\mathsf{ground}^+(KB)$ are the same as the RBox and ABox of KB. For each TBox axiom $\alpha = C \sqsubseteq D \in KB$, the following axioms are added to $\mathsf{ground}^+(KB)$:*

1. *For each nominal schema $\{x\}$ safe for α, with safe occurrences in environments $S_i(a_i, x)$ for $i = 1, \ldots, l$, introduce a fresh concept name $O_{x,\alpha}$. For every individual $b \in N_I$ in KB, $\mathsf{ground}^+(KB)$ contains an axiom*

$$\bigcap_{i=1}^{l} \exists U.S_i(a_i, b) \sqsubseteq \exists U.(\{b\} \sqcap O_{x,\alpha}),$$

2. *A concept C' is obtained from C as follow. Initialize $C' := C$. For each nominal schema $\{x\}$ that is safe for α: (a) replace all safe occurrences $S(a, x)$ in C' by $\{a\}$; (b) replace the non-safe occurrence (if any) of $\{x\}$ in C' by $O_{x,\alpha}$; (c) set $C' := C' \sqcap \exists U.O_{x,\alpha}$. After these steps, C' contains only nominal schemas that are not safe for α, and neither for $C' \sqsubseteq D$.*

   Now add axioms $\mathsf{ground}(C' \sqsubseteq D)$ *to* $\mathsf{ground}^+(KB)$.

Theorem 3. *Given a $\mathcal{ELHOV}_n(\sqcap)$ knowledge base KB, the size of* $\mathsf{ground}^+(KB)$ *is exponential in n and polynomial in $|KB|$ [16].*

From Theorem 3, we know that deciding $\mathcal{ELHOV}_n(\sqcap)$ is in polynomial time. We also showed that resolution for $\mathcal{SROEL}(\sqcap_s, \times)$ is a polynomial algorithm in Section 3. Now, we are ready to bridge the gap by the lifting lemma. Before giving the lifting lemma, the ordered resolution parameters must be redefined.

Definition 8. *Let $\mathcal{R}^O_{DL}$ denote the resolution calculus $\mathcal{R}_{\mathcal{DL}}$ parameterized as follows:*

- The literal ordering is an admissible ordering $\succ$ such that $f \succ c \succ R \succ O \succ A$, for all function symbol f, constant symbol c, binary predicate symbol R, unary predicate symbol A and first order logic predicate O for nominal schemas.
- The selection function selects every negative maximal binary literal in each clause.

Lemma 2 (lifting lemma). *For a $\mathcal{ELHOV}_n(\sqcap)$ knowledge base KB, clause $C \in \Xi(KB)$ and $D \in \Xi(KB)$, if C can resolve with D, then there must exist at least one resolution inference between a clause $C' \in \mathsf{ground}(C)$ and $D' \in \mathsf{ground}(D)$.*

Proof. We show this lemma by proving contradiction, which is to show the statement that if there is no clause $C' \in \mathsf{ground}(C)$ can resolve with clause $D' \in \mathsf{ground}(D)$, then C can not resolve with D. Without losing generality, we assume C in clause type (n) and D in clause type (m). We denote $[n \not\sim m]$ by clause type (n) cannot resolve with clause type (m). There are two possibilities.

- Clause type (n) and clause type (m) can resolve, but the resolved literals are not on the same predicate name. So no matter the clauses are grounded or not, they cannot resolve.
- Clause type (n) and clause type (m) cannot resolve. So we need to show they cannot resolve even before grounding. We enumerate all the impossible resolution cases. For ordered resolution inference, $[2 \not\sim 5]$, because before grounding, $\neg A(x)$ is not selected and not the maximal literal, so $C(x)$ in (2) cannot resolve with $\neg A(x)$ in (5). Similarly, we have $[2 \not\sim 6, 7, 8, 10, 11]$ and $[7 \not\sim 5, 6, 7, 8, 9, 10, 11]$. For positive superposition, $[6 \not\sim 10, 12]$, because violate the condition of positive superposition before grounding. Similarly, $[7 \not\sim 10, 12]$ and also $[10 \not\sim 11]$.

Therefore, if there is no clause $C' \in \mathsf{ground}(C)$ which can resolve with clause $D' \in \mathsf{ground}(D)$, then C can not resolve with D. Hence, as for clause $C \in \Xi(KB)$ and $D \in \Xi(KB)$, if C can resolve with D, then there must exist at least one resolution inference between a clause $C' \in \mathsf{ground}(C)$ and $D' \in \mathsf{ground}(D)$.

Theorem 4. *Given an $\mathcal{ELHOV}_n(\sqcap)$ knowledge base KB, saturating $\Xi(KB)$ by $\mathcal{R}^O_{DL}$ decides satisfiability of KB and runs in time polynomial in $|KB|$.*

Proof. By the lifting lemma, reasoning on clauses before grounding take fewer steps than the clauses after grounding. By theorem 3, we know that reasoning on $\mathsf{ground}^+(KB)$ is in polynomial time in $|KB|$. We also know that the resolution procedure for $\mathcal{ELHO}(\sqcap)$ is in polynomial time by Theorem 2 and $\mathsf{ground}^+(KB)$ is a $\mathcal{ELHO}(\sqcap)$ KB. Therefore, the resolution procedure for $\mathcal{ELHOV}_n(\sqcap)$ KB takes fewer steps than $\mathsf{ground}^+(KB)$, and so $\mathcal{R}^O_{DL}$ decides satisfiability of $\Xi(KB)$ in time polynomial in $|KB|$.

The proof is closely relevant with the order and selection parameter of ordered resolution. If we change the setting of the parameters, the lifting lemma might not hold.

To the best of our knowledge, this parameter setting of the order and select function is best. If O has the highest order among all predicates, then the clauses which contain O cannot resolve with others unless $O(x)$ literals are resolved. Thus, it has no difference with *full grounding* method, because resolving $O(x)$ is actually grounding x with all known individuals. If $f \succ O \succ P$, where P denotes the DL predicate name, $\neg A(x) \vee R(x, f(x))$ and $\neg O(z) \vee \neg R(x, z) \vee S(x, z)$ cannot resolve. That is to say, we still need to ground $O(x)$ in some clauses. If we set the parameter as $f \succ R \succ O \succ A$, $\neg A(x) \vee R(x, f(x))$ and $\neg O(z) \vee \neg R(x, z) \vee S(x, z)$ can resolve. So it delays the grounding even later. However, if we force O to be the lowest order, there are undesired clauses violating termination. Therefore, we choose $f \succ R \succ O \succ A$ as the order.

There are several reasons that resolution procedure can be much more efficient than the naive *full grounding* method. First of all, the number of clauses translated from $\mathcal{ELHOV}_n(\sqcap)$ knowledge base KB is much fewer than the number of clauses of $\mathsf{ground}^+(KB)$. Secondly, some clauses cannot do any further resolution, such that they can be seen as redundant clauses. For example, resolving $\neg A(x) \vee R(x, f(x))$ and $\neg O(z) \vee \neg R(x, z) \vee B(x)$ produces a clause containing $\neg O(f(x))$. However, $\neg O(f(x))$ cannot resolve with any others, because there are only positive literal of $O(a)$ in KB and thus $\neg O(f(x))$ cannot unify with others. For the resolution procedure, we can even apply the powerful redundancy technique to reduce the number of clauses [6], e.g., all the tautologies can be removed directly in the resolution procedure.

After saturation, we can reduce all the clauses into a disjunctive datalog program $\mathsf{DD}(KB)$. The program $\mathsf{DD}(KB)$ entails the same set of ground facts as KB, Thus, instance checking in KB can be performed as query answering in $\mathsf{DD}(KB)$. Database systems usually contain Datalog reasoning and compute query answers in one pass efficiently, either bottom-up or top-down. Especially, when KB containing nominal schemas, the decision procedure needs to do a lot

of Instance Checking implicitly. So resolution approach is particularly suitable for nominal schemas. Comparatively, tableau algorithms might need to run for each individual in ABox. (see more details about disjunctive datalog program in [6])

5 Examples

We now present a rather simple example that points out how the resolution procedure works and why it's more efficient than *full grounding* approach in general. Intuitively, our approach delays grounding only when it's necessary to do so.

Consider the following clearly unsatisfiable KB containing nominal schema.

$KB = \{$
$\exists hasParent.\{z\} \sqcap \exists hasParent.\exists married.\{z\} \sqcap Teen \sqsubseteq Child,$
$hasParent(john, mark),$
$hasParent(john, mary),$
$married(mary, mark),$
$Teen(john),$
$\neg Child(john)\}$

We first translate all the DL axioms into first order logic clauses.

$\Xi(KB) = \{$
(1) $\neg O(z) \vee \neg hasParent(x, z) \vee \neg hasParent(x, y) \vee \neg married(y, z) \vee \neg Teen(x) \vee Child(x)$
(2) $hasParent(john, mark),$
(3) $hasParent(john, mary),$
(4) $married(mary, mark),$
(5) $Teen(john),$
(6) $\neg Child(john)\}$

$\Xi(KB)$ also contains $O(john)$, $O(mary)$ and $O(mark)$ because they are known individuals. By saturating $\Xi(KB)$ we obtain the following clauses (the notation R(n,m) means that a clause is derived by resolving clauses n and m):

(7) $\neg O(mark) \vee \neg hasParent(john, y) \vee \neg married(y, mark) \vee \neg Teen(john) \vee Child(john)$ R(1,2)
(8) $\neg O(mark) \vee \neg married(mary, mark) \vee \neg Teen(john) \vee Child(john)$ R(7,3)
(9) $\neg O(mark) \vee \neg Teen(john) \vee Child(john)$ R(8,4)

Since O symbol has a higher order than unary predicate, (9) $\neg O(mark) \vee \neg Teen(john) \vee Child(john)$ will resolve with $O(mark)$ and produce (10) $\neg Teen(john) \vee Child(john)$.

(11) $Child(john)$ R(10,5)
(12) $\bot$ R(11,6)

Now, we can see that the O symbol literal is resolved at the very last. That is to say, the grounding of nominal schemas in such procedure has been delayed. However, if we use the *full grounding* approach, the KB will contain clause $\exists hasParent.\{john\} \sqcap \exists hasParent.\exists married.\{john\} \sqcap Teen \sqsubseteq Child$, which is

absolutely unnecessary for inference. Consider KB has even more irrelevant known individuals, it will generate even more useless clauses. Therefore, clearly, our approach is much better than the *full grounding* one.

6 Related Work and Discussion

There are several algorithms particularly for $\mathcal{EL}$ family, but none of them can be easily to extend to DLs containing nominal schemas. For $\mathcal{SROEL}(\sqcap_s, \times)$, the only reasoning approach was proposed in [15]. Instead of using traditional tableau method, all DL axioms are rewritten into a set of datalog rules. However, it is unclear how to translate nominal schemas into such rules. Since all of the rewriting rules apply on the normal forms, but axioms containing nominal schemas are not able to be normalized to the best of our knowledge. Also, we do not know how to extend completion-rule based algorithm [1] and the recently concurrent algorithm [9,8], because they also need to normalize axioms at first. In [13], the author tried to apply a selective and delayed grounding technique to decide $\mathcal{ALCO}$ extended with nominal schemas. The advantage is the technique can easily extend to more expressive logic, such as $\mathcal{SROIQV}_n$. But it is hardly to say such algorithm is suitable for DLs with nominal schemas, because one has to find a very good heuristics for grounding choices.

When we want to extend our resolution procedure to capture general role chain, it becomes much more difficult. The possible cyclic role chain axioms can do self-resolution, which prevent termination. For example, a transitive relation S satisfies $S(a,b) \wedge S(b,c) \rightarrow S(a,c)$ can resolve with itself to yield a new clause $S(a,b) \wedge S(b,c) \wedge S(c,d) \rightarrow S(a,d)$ and so on. In [10], the problem was partially solved by eliminating transitive role with another equisatisfiable axiom containing universal quantifier. In [2], the authors developed a so-called ordered chaining rule based on standard techniques from term rewriting. It can deal with binary relations with composition laws of the form $R \circ S \sqsubseteq U$ in the context of resolution-type theorem proving. However, the approach applies even more restricted order on role predicates than the acyclic role chains in $\mathcal{SROIQ}$. So it can not solve the problem of general role chain neither. To the best of our knowledge, there is no resolution procedure that can deal with general role chain. And hence, it becomes to our next goal.

The extension of our algorithm to deal with cross-products becomes intractable for conjunction of roles. The reason is that using extended role hierarchies, it is possible to express inclusion axioms with universal value restrictions of form $C \sqsubseteq \forall R.D$, or equivalently, inclusion axioms with inverse roles of form $\exists R^{-}.C \sqsubseteq D$ which were shown to cause intractability in [1]. Indeed, these axioms are expressible using three inclusion axioms: $C \times \top \sqsubseteq S$, $S \sqcap R \sqsubseteq H$ and $H \sqsubseteq \top \times D$, where S and H are fresh role names [7].

The extension of $\mathcal{ELHO}(\sqcap)$ with self role should not affect tractability, although it may cause clauses have longer length. For example, consider the following $KB = \{C \sqsubseteq \exists R.\mathsf{Self}, D \sqsubseteq \exists S.\mathsf{Self}, R \sqcap S \sqsubseteq T\}$. After saturation, $\Xi(KB)$ contains $\neg C(x) \vee \neg D(x) \vee T(x,x)$, which may resolve with other role conjunc-

tion axioms and so forth. So it is possible to have clauses with longer length, like $\neg C_1(x) \vee \ldots \vee \neg C_n(x) \vee R(x,x)$. However, we conjecture that the number n in $\neg C_1(x) \vee \ldots \vee \neg C_n(x) \vee R(x,x)$ is linear to the number of concept names in KB. So the resolution procedure for $\mathcal{ELHO}(\sqcap)$ with self role should be still in polynomial time. Due to that we want to keep this paper easier to read, we disallow self role constructor.

More problems will occur when extending to more expressive description logic $\mathcal{SROIQV}_n$. When translating $\mathcal{SROIQV}_n$ into first order logic, since one nominal schema can appear at different positions in an axiom, such that the number of corresponding FOL clauses can be exponential blow-up. For example, for such a DL axiom, $(\exists R_1.\{z\} \sqcup \exists S_1.\{z\}) \sqcap \ldots \sqcap (\exists R_n.\{z\} \sqcup \exists S_n.\{z\}) \sqsubseteq C$, since we cannot normalize it into smaller axioms, it has to be translated into a number of clauses in conjunctive normal form, and the number of such clauses is exponential blow-up.

Although theoretically optimal, the resolution procedure may not be scalable in practice. The reason seems to be that, despite optimizations, resolution still produces many unnecessary clauses (see discussion in [25]). Another algorithm, called hypertableau, seems to be very potential to efficiently deal with nominal schemas. Hypertableau algorithm takes unnormalized *DL-clauses* to infer based on the hypertableau rule. It can avoid unnecessary nondeterminism and the construction of large models, which are two primary sources of inefficiency in the tableau-based reasoning calculi [22]. We believe that the idea of the lift lemma can also work for hypertableau method, such that we may use the similar way to prove the feasibility of hypertableau for nominal schemas.

Nominal schemas have even more good properties. In [11,12], the author describes nominal schemas allow not only for a concise reconciliation of OWL and rules, but also that the integration can in fact be lifted to cover established closed world formalisms on both the OWL and the rules side. More precisely, they endow $\mathcal{SROIQ}$ with both nominal schemas and a generalized semantics based on the logic of minimal knowledge and negation as failure (MKNF). The latter is non-monotonic and captures both open and closed world modeling.

7 Conclusion and Future work

In this paper, we provide a polynomial resolution procedure for the description logic language $\mathcal{ELHOV}_n(\sqcap)$. We show that the algorithm is sound, complete and tractable. For future work, the main task is to implement the algorithm and compare it with the tableau approach with selective grounding strategy. We will also look into the hypertableau method to see if it can be extended. In general, we hope to develop a more efficient algorithm to be applicable for $\mathcal{SROELV}_n(\sqcap_s, \times)$, $\mathcal{SROIQV}_n(\sqcap_s, \times)$ and even more powerful DL languages.

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

References

1. Baader, F., Brandt, S., Lutz, C.: Pushing the el envelope. In: Kaelbling, L.P., Saffiotti, A. (eds.) IJCAI. pp. 364–369. Professional Book Center (2005)
2. Bachmair, L., Ganzinger, H.: Ordered chaining for total orderings. In: Bundy, A. (ed.) CADE. Lecture Notes in Computer Science, vol. 814, pp. 435–450. Springer (1994)
3. Bachmair, L., Ganzinger, H.: Resolution theorem proving. In: Robinson, J.A., Voronkov, A. (eds.) Handbook of Automated Reasoning, pp. 19–99. Elsevier and MIT Press (2001)
4. Boley, H., Hallmark, G., Kifer, M., Paschke, A., Polleres, A., Reynolds, D. (eds.): RIF Core Dialect. W3C Recommendation (22 June 2010), available at `http://www.w3.org/TR/rif-core/`
5. 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/`
6. Hustadt, U., Motik, B., Sattler, U.: Reasoning for Description Logics around SHIQ in a Resolution Framework. Tech. Rep. 3-8-04/04, FZI, Germany (2004)
7. Kazakov, Y.: Saturation-based decision procedures for extensions of the guarded fragment. Ph.D. thesis, Saarlndische Universitts- und Landesbibliothek, Postfach 151141, 66041 Saarbrcken (2005), `http://scidok.sulb.uni-saarland.de/volltexte/2007/1137`
8. Kazakov, Y., Krötzsch, M., Simančík, F.: Practical reasoning with nominals in the $\mathcal{EL}$ family of description logics. In: Brewka, G., Eiter, T., McIlraith, S.A. (eds.) Proceedings of the 13th International Conference on Principles of Knowledge Representation and Reasoning (KR'12). pp. 264–274. AAAI Press (2012)
9. Kazakov, Y., Krtzsch, M., Simank, F.: Concurrent classification of el ontologies (2011), to appear
10. Kazakov, Y., Motik, B.: A resolution-based decision procedure for *shoiq*. In: Furbach, U., Shankar, N. (eds.) IJCAR. Lecture Notes in Computer Science, vol. 4130, pp. 662–677. Springer (2006)
11. Knorr, M., Alferes, J.J., Hitzler, P.: Local closed world reasoning with description logics under the well-founded semantics. Artif. Intell. 175(9-10), 1528–1554 (2011)
12. Knorr, M., Martínez, D.C., Hitzler, P., Krisnadhi, A.A., Maier, F., Wang, C.: Recent advances in integrating owl and rules (technical communication). In: Krötzsch and Straccia [18], pp. 225–228
13. Krisnadhi, A., Hitzler, P.: A tableau algorithm for description logics with nominal schema. In: Krötzsch and Straccia [18], pp. 234–237
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.F. (eds.) Reasoning Web. Lecture Notes in Computer Science, vol. 6848, pp. 382–415. Springer (2011)
15. Krötzsch, M.: Efficient inferencing for OWL EL. In: Janhunen, T., Niemelä, I. (eds.) Proceedings of the 12th European Conference on Logics in Artificial Intelligence (JELIA'10). LNAI, vol. 6341, pp. 234–246. Springer (2010)
16. Krötzsch, M., Maier, F., Krisnadhi, A., Hitzler, P.: A better uncle for owl: nominal schemas for integrating rules and ontologies. In: Srinivasan, S., Ramamritham, K., Kumar, A., Ravindra, M.P., Bertino, E., Kumar, R. (eds.) WWW. pp. 645–654. ACM (2011)

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'08). LNCS, vol. 5318, pp. 649–664. Springer (2008)
18. 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. Springer (2012)
19. Martinezi, D.C., Krisnadhi, A., Maier, F., Sengupta, K., Hitzler, P.: Reconciling owl and rules. Tech. rep., Kno.e.sis Center, Wright State University, Dayton, OH, U.S.A. (2011), available from `http://www.pascal-hitzler.de/`
20. Motik, B.: Reasoning in Description Logics using Resolution and Deductive Databases. Ph.D. thesis, Univesität Karlsruhe (TH), Karlsruhe, Germany (January 2006)
21. Motik, B., Sattler, U., Studer, R.: Query answering for owl-dl with rules. In: McIlraith, S.A., Plexousakis, D., van Harmelen, F. (eds.) International Semantic Web Conference. Lecture Notes in Computer Science, vol. 3298, pp. 549–563. Springer (2004)
22. Motik, B., Shearer, R., Horrocks, I.: Hypertableau reasoning for description logics. J. Artif. Intell. Res. (JAIR) 36, 165–228 (2009)
23. Palù, A.D., Dovier, A., Pontelli, E., Rossi, G.: Answer set programming with constraints using lazy grounding. In: Hill, P.M., Warren, D.S. (eds.) ICLP. Lecture Notes in Computer Science, vol. 5649, pp. 115–129. Springer (2009)
24. Rudolph, S., Krötzsch, M., Hitzler, P.: Cheap boolean role constructors for description logics. In: Hölldobler, S., Lutz, C., Wansing, H. (eds.) JELIA. Lecture Notes in Computer Science, vol. 5293, pp. 362–374. Springer (2008)
25. Simancik, F., Kazakov, Y., Horrocks, I.: Consequence-based reasoning beyond horn ontologies. In: Walsh, T. (ed.) IJCAI. pp. 1093–1098. IJCAI/AAAI (2011)