5-1-2012

Extending Description Logic Rules

David Carral Martinez

Pascal Hitzler
pascal.hitzler@wright.edu

Extending Description Logic Rules*

David Carral Martínez and Pascal Hitzler

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

Abstract. Description Logics – the logics underpinning the Web Ontology Language OWL – and rules are currently the most prominent paradigms used for modeling knowledge for the Semantic Web. While both of these approaches are based on classical logic, the paradigms also differ significantly, so that naive combinations result in undesirable properties such as undecidability. Recent work has shown that many rules can in fact be expressed in OWL. In this paper we extend this work to include some types of rules previously excluded. We formally define a set of first order logic rules, C-Rules, which can be expressed within OWL extended with role conjunction. We also show that the use of nominal schemas results in even broader coverage.

1 Introduction

Several different paradigms have been devised to model ontologies for the Semantic Web [6]. Currently, the most prominent approaches for modeling this knowledge are description logics (DLs) [1] and rules based on the logic programming paradigm. Although both are based on classical logic, they differ significantly and the search for a satisfactory integration is still ongoing [4,11].

Even if the DL-based Web Ontology Language OWL [5], a W3C standard, is the main language for modeling ontologies in the Semantic Web, rule-based approaches have also proven very successful. Included in many commercial applications, rules continue to be pursued in parallel to OWL using the Rule Interchange Format RIF [2], also a W3C standard, as a rule exchange layer. Understanding the differences between both paradigms in order to come up with workable combinations has become a major effort in current research.

This paper extends on the work presented in [13] where it has been shown that, in fact, many rules can be expressed in OWL. We extend this work to include some types of rules previously excluded. We formally define C-Rules, a set of rules that can be embedded directly into OWL extended with role conjunction. We also discuss how our approach can be used in conjunction with previous weaker methods for embedding rules based on nominal schemas.

To express C-Rules in DL notation we employ the DL $\mathcal{SROIQ}(\sqcap_{\exists})$, an extension of $\mathcal{SROIQ}$ [8], which underlies OWL 2 DL. $\mathcal{SROIQ}(\sqcap_{\exists})$ encompasses $\mathcal{SROIQ}$ adding a restricted form of role conjunction.

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

To introduce our approach, consider the following rule R which cannot be readily expressed in $\mathcal{SROIQ}$ using known techniques. Although R may not have a single directly equivalent axiom in DL, we can transform it into a set of equisatisfiable statements in first-order predicate logic (FOL).

$$\text{hasFather}(x, y) \wedge \text{hasBrother}(y, z) \wedge \text{hasTeacher}(x, z) \rightarrow \text{TaughtByUncle}(x)$$

The example rule R can be represented as an equisatisfiable set of statements:

- $\text{hasFather}(x, y) \wedge \text{hasBrother}(y, z) \rightarrow \text{hasUncle}(x, z)$
- $\text{hasUncle}(x, z) \wedge \text{hasTeacher}(x, z) \rightarrow \text{hasTeacherAndUncle}(x, z)$
- $\text{hasTeacherAndUncle}(x, z) \rightarrow \text{TaughtByUncle}(x)$

This set of FOL statements can then be translated into

- $\text{hasFather} \circ \text{hasBrother} \sqsubseteq \text{hasUncle}$
- $\text{hasUncle} \sqcap \text{hasTeacher} \sqsubseteq \text{hasTeacherAndUncle}$
- $\exists \text{hasTeacherAndUncle}.\top \sqsubseteq \text{TaughtByUncle}$

and therefore the rule R can be expressed in DL notation.

Although some rules fall under our definition and are expressible using these combinations of role constructors, there are even more complex rules that cannot be simplified using the approach presented in this paper.

A prominently discussed idea for retaining decidability, and still be able to express complex rules, is to restrict the applicability of rules to named individuals. Rules with this kind of semantics are called DL-safe, and the combination of OWL and DL-safe rules is indeed decidable [7,16]. In this paper we also discuss the use of nominal schemas to express complex rules. Nominal schemas are a DL constructor presented in [15] described as "variable nominal classes." Restricted to stand only for named individuals as DL-safe variables, nominal schemas have the advantage of allowing us to express complex rules in native OWL notation.

The plan for the paper is as follows. After providing some preliminaries in Section 2, the paper continues in Section 3 with the formal definition of C-Rules with unary predicates in the head. Section 4 extends the approach presented in the previous section to rules with binary predicates in the head. Section 5 contains some examples. Section 6 contains the discussion about rules and nominal schemas. Section 7 includes the conclusions of the paper and future work.

2 Preliminaries

We introduce $\mathcal{SROIQ}(\sqcap_{\exists})$, a DL fragment that adds role conjunction, in a restricted way, to $\mathcal{SROIQ}$ [8]. Axioms of the form $R_1 \sqcap R_2 \sqsubseteq V$ are allowed in $\mathcal{SROIQ}(\sqcap_{\exists})$, where R_1 and R_2 are two (possibly complex) roles.[1]

Roles which appear on the right hand sides of axioms of the form $R_1 \sqcap R_2 \sqsubseteq V$ are restricted to only appear in concepts of the form $\exists V.C$. Although this

[1] In a sense, role conjunction was already implicit in [14], and is also used in [13], for a similar purpose.

precondition might look very restrictive, it suffices for the use of role conjunction for expressing rules, as discussed in this paper. As a technical note, in terms of regularity of RBoxes (required for decidability), we assume that for a role V appearing in an axiom $R_1 \sqcap R_2 \sqsubseteq V$ we have that both $R_1 \prec V$ and $R_2 \prec V$ ($\prec$ indicates the order in a regular role hierarchy).

$\mathcal{SROIQ}(\sqcap_\exists)$ bears the same semantics as $\mathcal{SROIQ}$, with the exception of the role conjunction constructor. The formal semantics is as usual (see, e.g., [17]), and for lack of space we do not repeat it here. Note that it follows easily from the arguments laid out in [17] that $\mathcal{SROIQ}(\sqcap_\exists)$ is decidable.

2.1 Description Logic Rules and Graph Notation

We define *U-Rules* (respectively, *B-Rules*) as rules of the form $\bigwedge B_i \rightarrow H$, where all B_i are unary or binary predicates and H is a unary (binary) predicate.[2] We refer to $\bigwedge B_i$ and H as the *body* and the *head* of the rule respectively. We assume without loss of generality that at least one of the variables in the head of the rule also appears in the body of the rule at least once.[3]

Let R be any given U-Rule or B-Rule. We can define an *undirected graph* G_R derived from R as a set of vertices and edges s.t. every variable in the body of R is a vertex of G_R and G_R contains an edge (t, u) if $S(t, u)$ is a binary predicate in the body of R, and t and u are different variables. Note, that rules containing a predicate of the form $R(x, x)$ where R is a complex role cannot be expressed in $\mathcal{SROIQ}$, as it will be shown in Section 3.1. Consequently, if this is the case the reduction process cannot be performed. Graphs will be used across the paper to represent and reduce rules in an easy and intuitive way.

Note that constants, and even binary predicates containing only one variable, are not included in the graph. Having a prefixed meaning, constants can be simplified independently and therefore, there is no need to relate them to other elements in the graph. On the other hand, FOL variables, having shared non-fixed meanings, need to keep the links that determine their relation to the predicates in the rule where they appear.

We have defined undirected graphs G_R as sets and consequently there cannot be repetitions amongst its elements. Even if two different binary predicates relate the same pair of terms, in the graph these predicates map to the same single edge. Note also that we work with undirected graphs and therefore, elements (u, t) and (t, u) stand for different representations of the same edge.

For an undirected graph G_R, derived from a U-Rule or a B-rule R, we define a vertex as a *root vertex* r_R if it has been derived from a variable appearing both in the head and the body of the rule. Note that by our standing assumption, there will always be at least one root vertex for any given graph G_R.

We define, for two given vertices t and u, to be *directly connected* if $(t, u) \in G_R$. We define, for two given vertices t and u, to be *connected* as the symmetric

[2] In our context unary predicates will refer to any kind of valid unary $\mathcal{SROIQ}(\sqcap_\exists)$ concept in negation normal form, either in the head or in the body of the rule.

[3] Note that, if none of the variables in the head of the rule appears in the body, we can always include one of them in the body using a unary top predicate.

transitive closure of being directly connected. We assume without loss of generality that any two vertices in the graph are connected.[4] We define the *degree of a vertex* $d(t)$ as the number of different edges (t, u) in the graph G_R where t appears.

3 Description Logic Rules in $\mathcal{SROIQ}(\sqcap_\exists)$

In this section, we formally describe rules that can be expressed in the DL fragment $\mathcal{SROIQ}(\sqcap_\exists)$. While close in general spirit to the brief exhibit in [12, Section 8.3], our treatment is quite different.

Definition 1. *We propose in this definition a formal method to verify if a U-Rule R is expressible in $\mathcal{SROIQ}(\sqcap_\exists)$. Given a graph G_R derived from a given rule R, repeat non-deterministically and exhaustively:*

- *Substitute a pair of edges (t, u) and (u, v) by a single edge (t, v) and eliminate vertex u if $d(u) = 2$ and u is a non-root vertex.*

$$G_R := \{G_R \setminus \{(t, u),\ (u, v),\ u\} |\ d(u) = 2,\ u \neq r_R\} \cup \{(t, v)\}$$

- *Eliminate an edge (t, u) and a vertex u where $d(u) = 1$ and u is a non-root vertex.*

$$G_R := \{G_R \setminus \{(t, u),\ u\}\ |(t, u) \in G_R,\ d(u) = 1,\ u \neq r_R\}$$

A rule R is expressible in $\mathcal{SROIQ}(\sqcap_\exists)$ if the graph G_R gets reduced to the root vertex after the reduction process.

The reduction process defined in Definition 1 only halts under two different situations. Either the graph G_R gets reduced to a single vertex or, for every non-root vertex u in the graph we have that $d(u) \geq 3$ (possibly after several reduction steps). Steps 1 and 2 presented in the previous reduction process can remove a vertex u if $d(u) = 2$ or $d(u) = 1$ respectively if u is a non-root vertex. It is obvious that a graph containing a non-root vertex u s.t. $d(u) < 3$ can be reduced at least one step further, and a graph where $d(u) \geq 3$ for every non-root vertex $u \in G_R$ cannot be simplified.

The process presented in Definition 1 presents a formal approach to determine if a given rule R is expressible in $\mathcal{SROIQ}(\sqcap_\exists)$. In the sequel we explain how, from this graph reduction approach, we can derive a set of $\mathcal{SROIQ}(\sqcap_\exists)$ axioms equivalent to the original rule R. Formally stated, the following two lemmas and theorem hold. Their correctness is shown in Section 3.1.

Note that, even if we do not provide an explicit definition for rules that are not expressible in $\mathcal{SROIQ}(\sqcap_\exists)$ this can be easily inferred. Rules with four nodes that are fully connected through paths not containing any of those nodes are not expressible. Even if all the other nodes in the paths get simplified these four nodes will form a clique where each one of them has degree three or higher. Therefore, they cannot be reduced using the approach presented in this paper.

[4] Again, we can connect any two variables in a rule using the universal role—although regularity issues need to be taken into consideration here.

Lemma 1. *Let R be a U-Rule with a derived graph G_R. Every transformation performed on G_R as explained in Definition 1 leads to a new reduced graph $G_{R'}$. Then a new rule R_1 can be constructed from R s.t. we can derive a graph G_{R_1} from R_1 and $G_{R_1} = G_{R'}$. Furthermore, there exist a set of $\mathcal{SROIQ}(\sqcap_\exists)$ statements α_1 s.t. $\{R_1\} \cup \alpha_1 \equiv \{R\}$.*

Definition 2. *The process defined in Lemma 1 can be repeated iteratively producing a new rule from every rule derived from R s.t. $R \equiv R_1 \cup \alpha_1$, $R_1 \equiv R_2 \cup \alpha_2, ..., R_{t-1} \equiv R_t \cup \alpha_t$. R_t is a rule with a derived graph G_{R_t} s.t. G_{R_t} cannot be reduced anymore. If G_{R_t} has been reduced to a single vertex we call R_t a* terminal rule*.*

Lemma 2. *A terminal rule R_t is directly expressible as a $\mathcal{SROIQ}(\sqcap_\exists)$ axiom.*

Intuitively, the simplifications done on the graph will be mirrored in the rule. Through this iterative process we can start reducing the rule, splitting it into several $\mathcal{SROIQ}$ axioms that preserve equisatisfiability. When the rule gets reduced to a terminal rule it can be directly translated into $\mathcal{SROIQ}(\sqcap_\exists)$. Adding this translation to the set of previous axioms derived in the reduction process we obtain an equivalent set of atoms in $\mathcal{SROIQ}(\sqcap_\exists)$. The following Theorem follows directly from Lemmas 1 and 2.

Theorem 1. *Assume that a rule R is reduced to terminal rule R_t. Then the original knowledge base KB containing R is equisatisfiable w.r.t. a new knowledge base KB'. KB' is obtained from KB by substituting R by $\alpha_1 \cup ... \cup \alpha_{t-1} \cup \{R_t\}$ where R_t is the direct translation of the terminal rule R_t and $\alpha_1, ..., \alpha_{t-1}$ are the sets of axioms produced in every step during the reduction process of rule R.*

3.1 Satisifiability Preserving Transformations

We show how to carry out the mentioned graph transformations for a given rule R, preserving equisatisfiability in every step. For every transformation we present a table with three columns. The first column shows the initial subset of R that will be replaced in the next iteration of the reduction process. The second column includes the new simplified subset. By substituting the initial subset by the simplified one in the original rule R we obtain the new simplified rule. Column three shows all the $\mathcal{SROIQ}(\sqcap_\exists)$ statements that we need to add to the knowledge base in order to preserve equivalence.

Roling Up Unary Predicates We start by proving that unary predicates can be automatically embedded into binary predicates using role constructors. Therefore, these predicates do not need to be considered when we build the graph to know if a description logic rule R is expressible in $\mathcal{SROIQ}(\sqcap_\exists)$. This technique, called rolification, is presented in [11].

Initial rule subset	Eq Subset	Set α
$V(x,y) \wedge D(y)$	$S(x,y)$	$D \sqsubseteq \exists W.\text{Self}$
		$V \circ W \sqsubseteq S$

Hereby, W and S are fresh names not already appearing in the knowledge base.

Proposition 1. *Let KB be a knowledge base containing a U-Rule R s.t. $V(x,y) \wedge D(y)$ appears in R. From KB we can construct an equivalent knowledge base KB' s.t. $KB' := \{KB \setminus R\} \cup \{R'\} \cup \alpha$, where R' is a new U-Rule obtained from R by substituting $V(x,y) \wedge D(y)$ with $S(x,y)$ and the set α contains the axioms $D \equiv \exists W.Self$ and $V \circ W \sqsubseteq S$, with W and S fresh predicate names. We prove that KB and KB' are equisatisfiable knowledge bases.*

Proof. Assume M is a model for the KB. Obviously M models R. From M we can build a model M' for KB' s.t. M' is identical to M except for the mappings $W^{M'} = \{\ (b,b) \mid b \in D^M\ \}$ and $S^{M'} = \{\ (a,b) \mid (a,b) \in V^M$ and $(b,b) \in W^{M'}\ \}$. These mappings are enforced by the new α axioms added to the knowledge base. Obviously, if M models R, M' models axioms $\{R'\} \cup \alpha$ and hence, M' is a model for KB'.

Now assume that M' is a model of $\{R'\} \cup \alpha$. Then we have that $M = M'$ is a model for R. Consider ground instances of the rule and assume that $M \models \bigwedge B_i$. Now we need to prove that $M \models H$. Since we assume that $M \models \bigwedge B_i$ we have that M models every B_i in the body of the rule, in particular $M \models V(a,b) \wedge D(b)$. Hence, $(b,b) \in W^{M'}$ and $(a,b) \in S^{M'}$. Then we have that $M \models \bigwedge B'_i$ where B'_i is the R' body. Therefore $M \models H'$, where H' is the head of the new rule R'. We have that $H' = H$, hence, we have shown that that $M \models H$ being H. Consequently M models R and is a model for KB. □

All the other transformations presented below can be proved in a similar way. We thus refrain from including any more detailed formal arguments for them.

Note that $V(x,y) \wedge D(x)$ can be simplified in a similar way. In the following, W and S are fresh role names in the ontology.

Initial rule subset	Eq Subset	Set α
$V(x,y) \wedge D(x)$	$S(x,y)$	$D \sqsubseteq \exists W.\text{Self}$
		$W \circ V \sqsubseteq S$

Undirected Graph The direction of the edges in the graph can be changed using the inverse role constructor. Therefore, there is no need for our algorithm in Definition 1 to check the direction of binary predicates when we apply different simplifications. Below, S is fresh role name in the knowledge base.

Rule Subset	Eq Subset	Set α to the KB
$R(x,y)$	$S(y,x)$	$R^- \sqsubseteq S$

Reducing Vertices of Degree Two

Rule Subset	Eq Subset	Set α to the KB
$R(x,y) \wedge W(y,z)$	$S(x,z)$	$R \circ W \sqsubseteq S$

Hereby, $d(y) = 2$ and S is fresh role name in the knowledge base.

Note that when we simplify a vertex by using a role chain we loose the reference to the term u in the middle of the role chain for a given rule R. Therefore this can only be executed for terms u with a degree of two, without further references in other predicates of the rule R.

Reducing Vertices of Degree One

Rule Subset	Eq Subset	Set α to the KB
$R(x,y) \wedge V(y,z)$	$R(x,y) \wedge D(y)$	$\exists V.\top \sqsubseteq D$

Hereby, $d(z) = 1$ and D is a fresh concept name in the knowledge base.

Note that even if we part from a very similar subset rule as in the previous subsection we follow a different method here. The fresh unary predicate produced can be simplified as shown in earlier sections.

Simplifying Binary Predicates with One Constant

Initial rule subset	Eq Subset	Set α
$V(x,a)$	$S(x)$	$\exists V.\{a\} \sqsubseteq S$

Hereby, S is a fresh unary predicate and a is a is a constant.

The fresh unary predicate can be simplified as shown in earlier sections. Note that we can always swap the order of the terms in a predicate of the form $V(a,x)$.

Simplifying Binary Predicates of the Form (x,x)

Initial rule subset	Eq Subset	Set α
$V(x,x)$	$S(x)$	$\exists V.\text{Self} \sqsubseteq S$

Hereby, S is a fresh unary predicate.

Using this simplification whenever possible, there is no need to consider these kind of predicates in the graph. Note that this can only be done if V is a simple role w.r.t. role box in the knowledge base. To retain decidability $\mathcal{SROIQ}$ does not allow to use complex in a concept of the form $\exists C.\text{Self}$.

Unifying Binary Predicates

Rule Subset	Eq Subset	Set α to the KB
$R(x,y) \wedge V(x,y)$	$S(x,y)$	$R \sqcap V \sqsubseteq S$

Hereby, S is a fresh role name in the knowledge base.

Any pair of binary predicates can be unified if both contain the same pair of variables. Note that, even if the variables do not appear in the same order, we can use the inverse role construct to align them. We assume without loss of generality that every pair of binary predicates containing the same pair of variables in a rule is automatically unified and therefore, we can define graphs as sets without repetitions of the same edge. Note that the unification of binary predicates needs to be done with a higher priority than other transformations, such as the reduction of vertices of degree two. If not cycles may be reduced to predicates of the form $R(x,x)$ where R is a complex role that cannot be simplified using the $\exists R.\text{Self}$.

The transformations just shown correspond to graph transformations as mentioned in Definition 1. The arguments just given thus constitute a proof of Lemma 1. Note that the order of the transformations is non-deterministic. This is a kind of "don't care" determinism, where the order in which we apply the rules does not matter. We further elaborate about this in Section 5.

Translating Terminal Rules A terminal rule R is a rule of the form $\bigwedge B_i \rightarrow H$. We have that the body of the rule $\bigwedge B_i$ contains one, and at most one, free FOL variable x appearing only once in a unary predicate of the form $B(x)$ (the graph has been reduced to the root vertex, and therefore, there is only one variable left appearing only once). The body $\bigwedge B_i$ might also contain other predicates of the form $C(a)$ or $R(b, c)$ s.t. a, b, and c are constants. The head H is composed of a single unary predicate $H(x)$ s.t. x is the same free variable that appears in the body.

A terminal rule R is translated into a DL inclusion axiom of the form $\bigsqcap B_i \sqsubseteq H$. This axiom contains a fresh concept H on the right hand side of the role inclusion axiom and a concept intersection on the left hand side featuring the next elements:

- A fresh concept B standing for the unary predicate $B(x)$ s.t. x is the only free variable appearing in the rule.
- A concept $\exists U.(C \sqcap \{a\})$ for every unary predicate of the form $C(a)$ appearing in the body where a is a constant.
- A concept $\exists U.(\{b\} \sqcap \exists R.\{c\})$ for every binary predicate of the form $R(b, c)$ appearing in the body of the rule where b and c are constants.

The argument just given also constitutes a proof of Lemma 2.

4 Rules with Binary Predicates in the Head

We can extend our approach to cover rules with binary predicates in the head. As already stated, at least one variable in the head of the rule needs to appear in the body. Attending to this fact we need to consider two different situations.

If only one of the terms in the head of the rule appears in the body the simplification is straightforward. We just need to substitute the binary predicate $R(x, y)$[5] in the head by a fresh unary predicate $C(x)$ and add the axiom $\exists R.\top \sqsubseteq C$ to the $\mathcal{SROIQ}(\sqcap_{\exists})$ knowledge base. After this modification the rule can be reduced using the same approach presented in the previous section.

If both variables appearing in the head of the rule appear in the body we need to slightly modify our method presented in Section 3. In this case, we consider both variables in the head as root vertices. Now, if the rule is expressible in $\mathcal{SROIQ}(\sqcap_{\exists})$ the graph gets reduced to a single edge containing both variables in the head. This new kind of terminal rule can be expressed in $\mathcal{SROIQ}(\sqcap_{\exists})$ using a role inclusion axiom $S \sqsubseteq R$.

Theorem 2. *Let R be a B-Rule s.t. $S(x, y)$ is the predicate in the head H of R and both x and y appear in the body $\bigwedge B_i$ of the rule.*

Given a graph G_R derived from rule R, where we consider both x and y to be root vertices. Then exhaustively apply steps 1 and 2 from Definition 1.

If after the reduction process, the graph G_R gets reduced to a single edge, then rule R is expressible as a $\mathcal{SROIQ}(\sqcap_{\exists})$axiom.

[5] Assume x is the root vertex. Otherwise use the inverse role constructor to change the order of the terms in the predicate.

Again, we see that any G_R where every vertex u has $d(u) \geq 3$ (possibly obtained after several reduction steps) cannot be simplified. Otherwise the graph can be reduced to a single edge (u, t) s.t. $u \in H$ and $t \in H$ with H the head of the rule. Note that the procedure is almost the same as in Section 3, except for the accepting condition.

The process to translate the rule into a set of equivalent $\mathcal{SROIQ}(\sqcap_\exists)$ statements and proofs remain the same as the one presented in Section 3 except for the trivial translation of the terminal rule.

It is important to remark that in some cases a B-Rule may not be expressible while a U-Rule with the same body is. The second vertex might block a possible role reduction forbidding further simplifications. As an example we have that $R_1(x, y) \wedge R_2(x, w) \wedge R_3(w, y) \wedge R_4(y, z) \wedge R_5(w, z) \rightarrow C(x)$ is expressible in $\mathcal{SROIQ}(\sqcap_\exists)$ using our approach, while $R_1(x, y) \wedge R_2(x, w) \wedge R_3(w, y) \wedge R_4(y, z) \wedge R_5(w, z) \rightarrow C(x, z)$ is not.

Definition 3. *Formally, by* C-Rules *we mean the collection of all rules which are U-Rules or B-Rules and which are expressible in $\mathcal{SROIQ}$ using the approach presented in this paper.*

5 Examples

We start with a worked example for our transformation. As initial rule, we use

$$A(x, y) \wedge B(y) \wedge C(z, y) \wedge D(y, z) \wedge E(x, a) \wedge F(x, z) \wedge Y(a, b) \rightarrow Z(x)$$

where a and b are constant and x, y and z are free variables. Transformations following the discussion from Section 3 are detailed in Table 1.

Note that the rule listed in step 6 of Table 1 can already be directly translated to $\mathcal{SROIQ}(\sqcap_\exists)$ as $\exists M.\top \sqcap \exists E.\{a\} \sqcap \exists U.(\{a\} \sqcap \exists Y.\{b\}) \sqsubseteq Z$. But to improve readability of the paper, our rule reduction approach has been presented in a simpler form, avoiding such shortcuts. So, although the method shown is sound and correct, there are U-Rules and B-Rules, as the one presented in the example, where at some step of the reduction process no further simplifications are strictly required. An earlier translation of the rule reduces the number of statements that need to be added to the knowledge base. Recall, in particular, that rules with tree shaped graphs are directly expressible in DL [11,13].

Also, we have that reductions according to our transformations are applied non-deterministically. Although any rule reduction leading to a terminal rule is essentially correct, there might be differences in the set of axioms added to the knowledge base. For example, let R be a U-Rule containing the binary predicates $A(x, y)$ and $B(y, z)$ s.t. both y and z are variables not appearing anywhere else in the rule (hence, we have that $d(y) = 2$ and $d(z) = 1$). In the next reduction step, we can decide which variable, y or z, we want to erase.

Assuming we want to reduce $A(x, y)$ and $B(y, z)$ to $Z(x)$, there are two different ways of doing so, namely (1) first reducing y, and (2) first reducing z. In the first case, we end up with two axioms $A \circ B \sqsubseteq C$ and $\exists C.\top \sqsubseteq Z$, while in

Table 1. Reduction example. For every step substitute the rule in the previous row by the one in the current one and add the axioms on the second column to the knowledge base.

Step	Add to KB	Rule
1. Original Rule		$A(x,y) \wedge B(y) \wedge C(z,y) \wedge D(y,z) \wedge$ $E(x,a) \wedge F(x,z) \wedge Y(a,b) \rightarrow Z(x)$
2. Invert C	$C^- \sqsubseteq H$	$A(x,y) \wedge B(y) \wedge H(y,z) \wedge D(y,z) \wedge$ $E(x,a) \wedge F(x,z) \wedge Y(a,b) \rightarrow Z(x)$
3. Intersect D and H	$D \sqcap H \sqsubseteq I$	$A(x,y) \wedge B(y) \wedge I(y,z) \wedge E(x,a) \wedge$ $F(x,z) \wedge Y(a,b) \rightarrow Z(x)$
4. Role Up B	$B \sqsubseteq \exists J.\text{Self}$ $A \circ J \sqsubseteq K$	$K(x,y) \wedge I(y,z) \wedge E(x,a) \wedge$ $F(x,z) \wedge Y(a,b) \rightarrow Z(x)$
5. Simplify E	$\exists E.\{a\} \sqsubseteq L$	$K(x,y) \wedge I(y,z) \wedge L(x) \wedge$ $F(x,z) \wedge Y(a,b) \rightarrow Z(x)$
6. Role Up L	$L \sqsubseteq \exists M.\text{Self}$ $M \circ F \sqsubseteq N$	$K(x,y) \wedge I(y,z) \wedge$ $N(x,z) \wedge Y(a,b) \rightarrow Z(x)$
7. Reduce y	$K \circ I \sqsubseteq O$	$O(x,z) \wedge N(x,z) \wedge Y(a,b) \rightarrow Z(x)$
8. Intersect N and O	$N \sqcap O \sqsubseteq P$	$P(x,z) \wedge Y(a,b) \rightarrow Z(x)$
9. Reduce z	$\exists P.\top \sqsubseteq Q$	$Q(x) \wedge Y(a,b) \rightarrow Z(x)$
10. Translate Terminal Rule		$Q \sqcap \exists U.(\{a\} \sqcap \exists Y.\{b\}) \sqsubseteq Z$

the second case four axioms are required $\exists B.\top \sqsubseteq D$, $D \sqsubseteq \exists E.\text{Self}$, $A \circ E \sqsubseteq F$, and $\exists F.\top \sqsubseteq Z$. Note that giving priority to the reduction of variables with degree 2 reduces the number of required axioms to preserve equisatisfiability.

The regularity issue. In order to preserve decidability, $\mathcal{SROIQ}(\sqcap_\exists)$ enforces a strict partial order on complex roles (known as the *regularity* condition). When a C-Rule R is translated into $\mathcal{SROIQ}$, we add many new complex role inclusions axioms to the Rbox. These new roles may violate the partial order established by previous roles or even contradict role regularity by themselves. After the reduction of a C-Rule and the inclusion of new produced $\mathcal{SROIQ}(\sqcap_\exists)$ axioms, role regularity needs to be carefully checked in order to preserve decidability.

It is important to note that only the translation of expressible B-Rules might cause these role regularity violations. Although the role regularity hierarchy is modified in many steps of our rule reduction approach note that for every statement $R \prec S$ added we have that S is a fresh role. Fresh roles do not appear in any other part of the role hierarchy and therefore they cannot produce violations of the irreflexive order.

The only step of the process where the RBox may loose its regularity is in the translation of a terminal B-Rule. Adding this last axiom of the form $R \sqsubseteq S$ also adds the statement $R \prec S$ to the role hierarchy where S is a role which may have appear in any other part of the knowledge base.

Let us look at an example of a knowledge base where the reduction of some of the rules leads to role regularity violations. Let KB be a knowledge base

containing the following rule.

$$\text{TeacherOf}(y, x) \wedge \text{ReviewerOf}(y, z) \wedge \text{AuthorOf}(x, z) \rightarrow \text{IllegalReviewerOf}(y, z)$$

This rule places a pair of individuals under the binary predicate IllegalReviewerOf if the first is a teacher of the student who is the author of the reviewed paper. It can be transformed into the following set of $\mathcal{SROIQ}(\sqcap_\exists)$ axioms.

$$\begin{aligned} \text{TeacherOf} \circ \text{AuthorOf} &\sqsubseteq R_1 \\ \text{ReviewerOf} \sqcap R_1 &\sqsubseteq R_2 \\ R_2 &\sqsubseteq \text{IllegalReviewer} \end{aligned}$$

From these axioms, we obtain the relations TeacherOf $\prec R_1$, AuthorOf $\prec R_1$, ReviewerOf $\prec R_2$, $R_1 \prec R_2$, and $R_2 \prec$ IllegalReviewer, which entail the statement ReviewerOf $\prec$ IllegalReviewerOf. If would be natural to also add the axiom IllegalReviewerOf $\sqsubseteq$ ReviewerOf to the same ontology. However, the inclusion of this axiom adds the statement IllegalReviewerOf $\prec$ ReviewerOf which then violates role regularity.

A workaround to this issue, however, is possible, namely by employing nominal schemas, and we will return to this issue at the end of the next section.

6 Using Nominal Schemas and $\mathcal{SROIQV}(\sqcap_\exists)$

In earlier sections of this paper we have shown how to translate some FOL rules into DL notation. Although some rules can be translated to $\mathcal{SROIQ}(\sqcap_\exists)$ using the presented approach there are still more complex rules that cannot be simplified in the same way. To express these rules we employ the DL $\mathcal{SROIQV}(\sqcap_\exists)$.

$\mathcal{SROIQV}(\sqcap_\exists)$ adds nominal schemas, a DL constructor that can be used as "variable nominal classes," to the previously described $\mathcal{SROIQ}(\sqcap_\exists)$. We will refrain from introducing all formal details and refer the reader to [9,10,11,15] for this. While the semantic intuition behind nominal schemas is the same as that behind DL-safe variables, nominal schemas integrate seamlessly with DL syntax. As a consequence, the DL fragment $\mathcal{SROIQV}(\sqcap_\exists)$ encompasses DL-safe variables while staying within the DL/OWL language paradigm avoiding the use of hybrid approaches.

Using these nominal schemas we are able to express FOL rules that are not part of the treatment in Sections 3 and 4. Consider, for example, the rule

$$R_1(x, y) \wedge R_2(x, z) \wedge R_3(x, w) \wedge R_4(y, z) \wedge R_5(y, w) \wedge R_6(w, z) \rightarrow C(x). \quad (1)$$

Using $\{z\}$ and $\{w\}$ as nominal schemas, we can express it as

$$\exists R_1.(\exists R_4.\{z\} \sqcap \exists R_5.\{w\}) \sqcap \exists R_2.\{z\} \sqcap \exists R_3.(\{w\} \sqcap \exists.R_6.\{z\}) \sqsubseteq C$$

Note that, as already stated, nominal schemas do not share the same semantics defined for FOL variables. Nominal schemas, as DL-safe variables, are

restricted to stand only for nominals which are explicitly present in the knowledge base, while FOL variables can represent both named and unknown individuals. Therefore, the statements presented in the example just given are not strictly equivalent. Despite this fact, nominal schemas allow us to retain most of the entailments from the original FOL axiom without increasing the worst-case complexity of the DL fragment.

Although nominal schemas do not increase the worst-case complexity of the language [15], the number of different nominal schemas per axiom can affect the performance of the reasoning process [3,10]. It is therefore desirable to use as few nominal schemas as possible.

We now discuss two different ways of translating complex rules into DLs. First we prove the following.

Theorem 3. *Any U-Rule or B-Rule R containing m different free variables, where $m > 3$, can be directly expressed in DL using n nominal schemas s.t. $n \leq m - 2$.*

Proof. Given a rule R, firstly role up to simplify all binary predicates containing one constant as shown in Section 3. All binary predicates in the rule containing the same pair of variables are also replaced by a single binary predicate as described under *Unifying Binary Predicates* in Section 3.

Due to these transformations, we can now assume without loss of generality that the rule R contains only unary predicates with a constant, binary predicates with two constants, and binary predicates with two variables as arguments.

Now choose two variables x and y s.t. x is a root vertex and y is not. Using the inverse role construct we can now swap arguments in binary predicates s.t. x is always appearing in the first argument and y is in the first argument of every predicate where the other variable is not x. The variables selected will be the only ones not substituted by a nominal schema in the translated rule.

The rule body is now translated as shown in Table 2. The resulting DL expressions are joined by conjunction. $\bigwedge B_i(y)$, $R(x, y)$, and $\bigwedge R_i(y, v_i)$ are all the predicates where y appears.

Finally, the head $H(x)$ can be rewritten into the concept H (or if it is a binary predicate $H(x, z)$, a concept of the form $\exists H.\{z\}$), and the implication arrow replaced by class inclusion $\sqsubseteq$.

It is straightforward to formally verify the correctness of this transformation, and parts of the proof are simliar to the correctness proof from [15] for the embedding of binary Datalog into $\mathcal{SROIQV}$.

Clearly, the number of nominal schemas used to represent rule R is $n - 2$, the total number of free variables minus 2. □

With the transformation just given, rule (1) can be rewritten as

$$\exists R_1.(\exists R_4.\{z\} \sqcap \exists R_5.\{w\}) \sqcap \exists R_2.\{z\} \sqcap \exists R_3.\{w\} \sqcap \exists.U(\{w\} \sqcap \exists.R_6.\{z\}) \sqsubseteq C$$

Table 2. Translating Rules with Nominal Schemas

Predicate type	FOL	DL
Unary Predicates with 1 constant	$B(a)$	$\exists U.(\{a\} \sqcap B)$
Binary Predicates with 2 constants	$R(a,b)$	$\exists U.(\{a\} \sqcap \exists R.\{b\})$
Binary Predicates containing x and not y	$R(x,v)$	$\exists R.\{v\}$
Unary Predicates containing x	$B(x)$	B
Binary and Unary Predicates containing y	$R(x,y) \wedge \bigwedge B_i(y) \wedge \bigwedge R_i(y,v_i)$	$\exists R.(\bigsqcap B_i \sqcap \bigsqcap R_i.\{v_i\})$
Unary Predicates not containing x, y, or constants	$B(v)$	$\exists U.(\{v\} \sqcap B)$
Binary Predicates not containing x, y, or constants	$R(v,u)$	$\exists U.(\{v\} \sqcap \exists R.\{u\})$

As another example, the following rule transforms into the subsequent axiom.

$$R_1(x,y) \wedge R_2(y,z) \wedge R_3(w,z) \wedge R_4(x,z) \wedge R_5(y,w) \wedge R_6(w,u) \wedge R_7(y,u) \rightarrow H(x,u)$$

$$\exists R_1.(\exists R_2.\{z\} \sqcap \exists R_5.\{w\} \sqcap \exists R_7.\{u\}) \sqcap \exists U.(\exists\{w\} \sqcap \exists R_4.\{z\}) \sqcap \exists R_4.\{z\} \sqcap \exists U.(\{w\} \sqcap \exists R_6.\{u\}) \sqsubseteq \exists H.\{u\}$$

Theorem 4. *Any U-Rule or B-Rule R containing m different free variables can be expressed in DL by fully grounding $m-3$ free variables in R.*

Proof. By grounding every variable but three in the rule to named individuals we end up with a larger number of rules s.t. each one of them contains only three different free variables.[6] All these new grounded rules are expressible in DL using the approach presented in Section 3 of this paper. □

While the first of the approaches just mentioned allows us to represent all knowledge in $\mathcal{SROIQV}(\sqcap_\exists)$, the second one, although initially looking more efficient, requires preprocessing steps. Further research and algorithms are required to smartly deal with nominal schemas other than through such grounding, a cumbersome technique that requires too much space and time for current reasoners [3,10].

Let us finally return to the regularity issue discussed at the very end of Section 5. In the example discussed there, if we desire to also add the statement IllegalReviewerOf $\sqsubseteq$ ReviewerOf to the knowledge base, we cannot do so directly without violating regularity. Using nominal schemas, however, we can weaken this axiom to the form

$$\exists \text{IllegalReviewerOf}.\{x\} \sqsubseteq \exists \text{ReviewerOf}.\{x\}$$

[6] Note that any rule with three variables can be reduced used our approach. Having only three nodes in the graph for all of them we have that $d(u) \leq 2$ and therefore all of them can be reduced.

(or, e.g., to

$$\exists \text{IllegalReviewerOf}^{-}.\{x\} \sqsubseteq \exists \text{ReviewerOf}^{-}.\{x\}$$

or to both), where $\{x\}$ is a nominal schema. Essentially, this means that the role inclusion will apply in case the first argument or the second argument (the filler) is a known individual. I.e., the individuals connected by the IllegalReviewerOf property are not both are unnamed. While this is weaker than the standard semantics, it should provide a viable workaround in many cases. Also note that, alternatively, the regularity violation could be avoided by using a similarly weakened form of any of the other statements involved in the violation.

7 Conclusions And Future Work

This paper presents an extension of previous work on Description Logic Rules. We specify a translation of rules into OWL extended by role conjunction (more precisely, the description logic $\mathcal{SROIQ}(\sqcap_{\exists})$), which strengthens previously obtained such translations. In essense, our work shows that the fragment of Datalog which can be expressed in description logics is larger than previously assumed.

We furthermore included a discussion proposing two approaches to express more complex rules within the DL notation. Two different options are considered, the use of nominal schemas and fully grounding of some variables to named individuals. While the former might present a higher complexity, it allows to express these rules in native DL/OWL notation and avoid some cumbersome preprocessing steps.

Future work includes the development of a goal directed algorithm that can solve inference problems in $\mathcal{SROIQV}(\sqcap_{\exists})$ possibly including some other role constructs (probably the extension of a current tableau algorithm). This algorithm could serve as basis for practical implementations of reasoners that include role constructs amongst their features.

Also, the development of APIs that can automatically validate FOL rules as DL expressible and translate them into sets of equisatisfiable OWL axioms might be a very useful tool. Although some aspects of modeling ontologies, such as building concept hierarchies, are very intuitive when we work with DL/OWL languages, the translation of DL rules, as shown in this paper, may not be so straightforward for users that do not have a strong background in more formal logics. Additional tool support will be required to provide convenient modeling interfaces.

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. de Bruijn, J.: RIF RDF and OWL Compatibility. W3C Recommendation (22 June 2010), available at `http://www.w3.org/TR/rif-rdf-owl/`

3. 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/
4. Hitzler, P., Parsia, B.: Ontologies and rules. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, pp. 111–132. Springer, 2 edn. (2009)
5. 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/
6. Hitzler, P., Krötzsch, M., Rudolph, S.: Foundations of Semantic Web Technologies. Chapman & Hall/CRC (2009)
7. Horrocks, I., Patel-Schneider, P., Bechhofer, S., Tsarkov, D.: OWL Rules: A proposal and prototype implementation. Journal of Web Semantics 3(1), 23–40 (2005)
8. 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)
9. Knorr, M., Hitzler, P., Maier, F.: Reconciling OWL and non-monotonic rules for the Semantic Web. Tech. rep., Kno.e.sis Center, Wright State University, Dayton, OH, U.S.A. (2011), available from http://www.pascal-hitzler.de/
10. Krisnadhi, A., Hitzler, P.: A tableau algorithm for description logics with nominal schemas. Tech. rep., Kno.e.sis Center, Wright State University, Dayton, OH, U.S.A. (2011), available from http://www.pascal-hitzler.de/
11. 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)
12. Krötzsch, M.: Description Logic Rules, Studies on the Semantic Web, vol. 008. IOS Press/AKA (2010)
13. Krötzsch, M., Rudolph, S., Hitzler, P.: Description logic rules. In: Ghallab, M., Spyropoulos, C.D., Fakotakis, N., Avouris, N.M. (eds.) Proceeding of the 18th European Conference on Artificial Intelligence, Patras, Greece, July 21-25, 2008. pp. 80–84. IOS Press, Amsterdam (2008)
14. 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)
15. 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)
16. Motik, B., Sattler, U., Studer, R.: Query answering for OWL DL with rules. J. of Web Semantics 3(1), 41–60 (2005)
17. Rudolph, S., Krötzsch, M., Hitzler, P.: Cheap Boolean role constructors for description logics. In: Hölldobler, S., Lutz, C., Wansing, H. (eds.) Proceedings of the 11th European Conference on Logics in Artificial Intelligence (JELIA'08). LNAI, vol. 5293, pp. 362–374. Springer (2008)