Description Logic Rules*

Markus Krötzsch, Sebastian Rudolph, and Pascal Hitzler

Universität Karlsruhe (TH), Germany, email: [mak|sru|phi]@aifb.uni-karlsruhe.de

Abstract. We introduce *description logic (DL) rules* as a new rule-based formalism for knowledge representation in DLs. As a fragment of the Semantic Web Rule Language SWRL, DL rules allow for a tight integration with DL knowledge bases. In contrast to SWRL, however, the combination of DL rules with expressive description logics remains decidable, and we show that the DL $\mathcal{SROIQ}$ – the basis for the ongoing standardisation of OWL 2 – can completely internalise DL rules. On the other hand, DL rules capture many expressive features of $\mathcal{SROIQ}$ that are not available in simpler DLs yet. While reasoning in $\mathcal{SROIQ}$ is highly intractable, it turns out that DL rules can be introduced to various lightweight DLs without increasing their worst-case complexity. In particular, DL rules enable us to significantly extend the tractable DLs $\mathcal{EL}^{++}$ and DLP.

1 INTRODUCTION

The development of description logics (DLs) has been driven by the desire to push the expressivity bounds of these knowledge representation formalisms while still maintaining decidability and implementability. This has lead to very expressive DLs such as $\mathcal{SHOIN}$, the logic underlying the Web Ontology Language OWL DL, $\mathcal{SHOIQ}$, and more recently $\mathcal{SROIQ}$ [1] which is the basis for the ongoing standardisation of OWL 2[1] as the next version of the Web Ontology Language. On the other hand, more light-weight DLs for which most common reasoning problems can be implemented in (sub)polynomial time have also been sought, leading, e.g., to the tractable DL $\mathcal{EL}^{++}$ [2].

Another popular paradigm of knowledge representation are rule-based formalisms – ranging from logic programming to deductive databases. Similar to DLs, the expressivity and complexity of rule languages has been studied extensively [3], and many decidable and tractable formalisms are known. Yet, reconciling DLs and rule languages is far from easy, and many works have investigated this problem.

In this paper, we introduce *DL rules* as an expressive new rule language for combining DLs with first-order rules in a rather natural way that admits tight integration with existing DL systems. Since DLs can be considered as fragments of function-free first-order logic with equality, an obvious approach is to combine them with first-order

* Extended technical report with proofs for *Markus Krötzsch, Sebastian Rudolph, Pascal Hitzler: Description Logic Rules. Proceedings of the 18th European Conference on Artificial Intelligence (ECAI-08). IOS Press 2008.*

[1] OWL 2 is the forthcoming W3C recommendation for updating OWL, and is based on the OWL 1.1 member submission. See `http://www.w3.org/2007/OWL`.

Horn-logic rules. This is the basis of the *Semantic Web Rule Language SWRL* [4], proposed as a rule extension to OWL. However, reasoning becomes undecidable for the combination of OWL and SWRL, and thus more restricted rule languages have been investigated. A prominent example are *DL-safe rules* [5], which restrict the applicability of rules to a finite set of named individuals to retain decidability. Similar safety conditions have already been proposed for CARIN [6] in the context of the DL $\mathcal{ALCNR}$, where also acyclicity of rules and Tboxes was studied as an alternative for retaining decidability. Another basic approach is to identify the Horn-logic rules directly expressible in OWL DL (i.e. $\mathcal{SHOIN}$), and this fragment has been called *Description Logic Programs DLP* [7].

DL rules in turn can be characterised as a decidable fragment of SWRL, which corresponds to a large class of SWRL rules indirectly expressible in $\mathcal{SROIQ}$. They are based on the observation that DLs can express only tree-like interdependencies of variables. The concept expression $\exists\,\mathsf{worksAt}.\mathsf{University} \sqcap \exists\,\mathsf{supervises}.\mathsf{PhDStudent}$ that describes all people working at a university and supervising some PhD student, e.g., corresponds to the following first-order formula:

$$\exists y.\exists z.\mathsf{worksAt}(x, y) \wedge \mathsf{University}(y) \wedge \mathsf{supervises}(x, z) \wedge \mathsf{PhDStudent}(z)$$

Here variables form the nodes of a tree with root x, where edges are given by binary predicates. Intuitively, DL rules are exactly those SWRL rules, where premises (rule bodies) consist of one or more of such tree-shaped structures. One could, for example, formulate the following rule:

$$\mathsf{worksAt}(x, y) \wedge \mathsf{University}(y) \wedge \mathsf{supervises}(x, z) \wedge \mathsf{PhDStudent}(z) \to \mathsf{profOf}(x, z)$$

Since SWRL allows the use of DL concept expressions in rules, we obtain $\mathcal{SROIQ}$ rules, $\mathcal{EL}^{++}$ rules, or DLP rules as extensions of the respective DLs. For the case of $\mathcal{SROIQ}$, DL rules have independently been proposed in [8], where a tool for editing such rules was presented. As shown below, DL rules are indeed “syntactic sugar” in this case, even though rule-based presentations are often significantly simpler due to the fact that many rules require the introduction of auxiliary vocabulary for being encoded in $\mathcal{SROIQ}$. On the other hand, we also consider the light-weight DLs $\mathcal{EL}^{++}$ and DLP for which DL rules truly extend expressivity, and we show that the polynomial complexity of these DLs is preserved by this extension.

After providing some preliminary definitions in Section 2, we introduce DL rules in Section 3. Section 4 shows how DL rules can be internalised in $\mathcal{SROIQ}$, while Section 5 employs a novel reasoning algorithm to process $\mathcal{EL}^{++}$ rules directly. Finally, Section 6 introduces DLP 2 and establishes the tractability of reasoning in this DL-based rule language.

2 PRELIMINARIES

In this section, we recall the definition of the expressive description logic $\mathcal{SROIQ}$ [1]. We assume that the reader is familiar with description logics [9].

As usual, the DLs considered in this paper are based on three disjoint sets of *individual names* N_I, *concept names* N_C, and *role names* N_R containing the *universal role* $U \in \mathsf{N}_R$.

Definition 1. *A $\mathcal{SROIQ}$ Rbox for N_R is based on a set* $\mathbf{R}$ *of* roles *defined as* $\mathbf{R} := \mathsf{N}_R \cup \{R^- \mid R \in \mathsf{N}_R\}$, *where we set* $\mathrm{Inv}(R) := R^-$ *and* $\mathrm{Inv}(R^-) := R$ *to simplify notation. In the sequel, we will use the symbols R, S, possibly with subscripts, to denote roles.*

A generalised role inclusion axiom *(RIA) is a statement of the form $S_1 \circ \ldots \circ S_n \sqsubseteq R$, and a set of such RIAs is a generalised* role hierarchy. *A role hierarchy is* regular *if there is a strict partial order $\prec$ on* $\mathbf{R}$ *such that*

– *$S \prec R$ iff $\mathrm{Inv}(S) \prec R$, and*
– *every RIA is of one of the forms:*

$$R \circ R \sqsubseteq R, \quad R^- \sqsubseteq R, \quad S_1 \circ \ldots \circ S_n \sqsubseteq R, \quad R \circ S_1 \circ \ldots \circ S_n \sqsubseteq R, \quad S_1 \circ \ldots \circ S_n \circ R \sqsubseteq R$$

such that $R \in \mathsf{N}_R$ is a (non-inverse) role name, and $S_i \prec R$ for $i = 1, \ldots, n$.

The set of simple *roles for some role hierarchy is defined inductively as follows:*

– *If a role R occurs only on the right-hand-side of RIAs of the form $S \sqsubseteq R$ such that S is simple, then R is also simple.*
– *The inverse of a simple role is simple.*

A role assertion *is a statement of the form* $\mathsf{Ref}(R)$ *(*reflexivity*),* $\mathsf{Asy}(S)$ *(*asymmetry*), or* $\mathsf{Dis}(S, S')$ *(*role disjointness*), where S and S' are simple. A $\mathcal{SROIQ}$* Rbox *is the union of a set of role assertions together and a role hierarchy. A $\mathcal{SROIQ}$ Rbox is regular if its role hierarchy is regular.*

Definition 2. *Given a $\mathcal{SROIQ}$ Rbox $\mathcal{R}$, the set of* concept expressions $\mathbf{C}$ *is defined as follows:*

– $\mathsf{N}_C \subseteq \mathbf{C}$, $\top \in \mathbf{C}$, $\bot \in \mathbf{C}$,
– *if $C, D \in \mathbf{C}$, $R \in \mathbf{R}$, $S \in \mathbf{R}$ a simple role, $a \in \mathsf{N}_I$, and n a non-negative integer, then $\neg C$, $C \sqcap D$, $C \sqcup D$, $\{a\}$, $\forall R.C$, $\exists R.C$, $\exists S.\mathsf{Self}$, ${\leq}n\,S.C$, and ${\geq}n\,S.C$ are also concept expressions.*

Throughout this paper, the symbols C, D will be used to denote concept expressions. A $\mathcal{SROIQ}$ Tbox *is a set of* general concept inclusion axioms *(GCIs) of the form $C \sqsubseteq D$.*

An individual assertion *can have any of the following forms: $C(a)$, $R(a, b)$, $\neg R(a, b)$, $a \not\approx b$, with $a, b \in \mathsf{N}_I$ individual names, $C \in \mathbf{C}$ a concept expression, and $R, S \in \mathbf{R}$ roles with S simple. A $\mathcal{SROIQ}$* Abox *is a set of individual assertions.*

A $\mathcal{SROIQ}$ knowledge base KB *is the union of a regular Rbox $\mathcal{R}$, and an Abox $\mathcal{A}$ and Tbox $\mathcal{T}$ for $\mathcal{R}$.*

We further recall the semantics of $\mathcal{SROIQ}$ knowledge bases.

Definition 3. *An interpretation $\mathcal{I}$ consists of a set $\Delta^{\mathcal{I}}$ called* domain *(the elements of it being called* individuals*) together with a function $\cdot^{\mathcal{I}}$ mapping*

Name	Syntax	Semantics
inverse role	R^-	$\{\langle x,y\rangle \in \Delta^\mathcal{I} \times \Delta^\mathcal{I} \mid \langle y,x\rangle \in R^\mathcal{I}\}$
universal role	U	$\Delta^\mathcal{I} \times \Delta^\mathcal{I}$
top	$\top$	$\Delta^\mathcal{I}$
bottom	$\bot$	$\emptyset$
negation	$\neg C$	$\Delta^\mathcal{I} \setminus C^\mathcal{I}$
conjunction	$C \sqcap D$	$C^\mathcal{I} \cap D^\mathcal{I}$
disjunction	$C \sqcup D$	$C^\mathcal{I} \cup D^\mathcal{I}$
nominals	$\{a\}$	$\{a^\mathcal{I}\}$
univ. restriction	$\forall R.C$	$\{x \in \Delta^\mathcal{I} \mid \langle x,y\rangle \in R^\mathcal{I}$ implies $y \in C^\mathcal{I}\}$
exist. restriction	$\exists R.C$	$\{x \in \Delta^\mathcal{I} \mid$ for some $y \in \Delta^\mathcal{I}$, $\langle x,y\rangle \in R^\mathcal{I}$ and $y \in C^\mathcal{I}\}$
Self concept	$\exists S.\mathsf{Self}$	$\{x \in \Delta^\mathcal{I} \mid \langle x,x\rangle \in S^\mathcal{I}\}$
qualified number	$\leq n\, S.C$	$\{x \in \Delta^\mathcal{I} \mid \#\{y \in \Delta^\mathcal{I} \mid \langle x,y\rangle \in S^\mathcal{I}$ and $y \in C^\mathcal{I}\} \leq n\}$
restriction	$\geq n\, S.C$	$\{x \in \Delta^\mathcal{I} \mid \#\{y \in \Delta^\mathcal{I} \mid \langle x,y\rangle \in S^\mathcal{I}$ and $y \in C^\mathcal{I}\} \geq n\}$

Fig. 1. Semantics of concept constructors in $\mathcal{SROIQ}$ for an interpretation $\mathcal{I}$ with domain $\Delta^\mathcal{I}$.

- *individual names to elements of* $\Delta^\mathcal{I}$,
- *concept names to subsets of* $\Delta^\mathcal{I}$, *and*
- *role names to subsets of* $\Delta^\mathcal{I} \times \Delta^\mathcal{I}$.

The function $\cdot^\mathcal{I}$ *is inductively extended to role and concept expressions as shown in Table 1. An interpretation* $\mathcal{I}$ satisfies *an axiom* φ *if we find that* $\mathcal{I} \models \varphi$*:*

- $\mathcal{I} \models S \sqsubseteq R$ *if* $S^\mathcal{I} \subseteq R^\mathcal{I}$,
- $\mathcal{I} \models S_1 \circ \ldots \circ S_n \sqsubseteq R$ *if* $S_1^\mathcal{I} \circ \ldots \circ S_n^\mathcal{I} \sqsubseteq R^\mathcal{I}$ *(*$\circ$ *being overloaded to denote the standard composition of binary relations here),*
- $\mathcal{I} \models \mathsf{Ref}(R)$ *if* $R^\mathcal{I}$ *is a reflexive relation,*
- $\mathcal{I} \models \mathsf{Asy}(R)$ *if* $R^\mathcal{I}$ *is antisymmetric and irreflexive,*
- $\mathcal{I} \models \mathsf{Dis}(R,S)$ *if* $R^\mathcal{I}$ *and* $S^\mathcal{I}$ *are disjoint,*
- $\mathcal{I} \models C \sqsubseteq D$ *if* $C^\mathcal{I} \subseteq D^\mathcal{I}$.

An interpretation $\mathcal{I}$ satisfies *a knowledge base* KB *(we then also say that* $\mathcal{I}$ *is a* model *of* KB *and write* $\mathcal{I} \models$ KB*) if it satisfies all axioms of* KB*. A knowledge base* KB *is* satisfiable *if it has a model. Two knowledge bases are* equivalent *if they have exactly the same models, and they are* equisatisfiable *if either both are unsatisfiable or both are satisfiable.*

Further details on $\mathcal{SROIQ}$ can be found in [1]. We have omitted here several syntactic constructs that can be expressed indirectly, especially Rbox assertions for transitivity, reflexivity of simple roles, and symmetry.

3 DESCRIPTION LOGIC RULES

In this section, we formally introduce *DL rules* as a syntactic fragment of first-order logic.

Definition 4. *Consider some description logic $\mathcal{L}$ with concept expressions* $\mathbf{C}$*, individual names* N_I*, roles* $\mathbf{R}$ *(possibly including inverse roles), and let* $\mathbf{V}$ *be a countable set of first-order variables. Given terms* $t, u \in \mathsf{N}_I \cup \mathbf{V}$*, a* concept atom (role atom) *is a formula of the form* $C(t)$ *(*$R(t, u)$*) with* $C \in \mathbf{C}$ *(*$R \in \mathbf{R}$*).*

To simplify notation, we will often use finite sets S *of (role and concept) atoms for representing the conjunction* $\bigwedge S$*. Given such a set* S *of atoms and terms* $t, u \in \mathsf{N}_I \cup \mathbf{V}$*, a* path *from* t *to* u *in* S *is a non-empty sequence* $R_1(x_1, x_2), \ldots, R_n(x_n, x_{n+1}) \in S$ *where* $x_1 = t$*,* $x_i \in \mathbf{V}$ *for* $2 \leq i \leq n$*,* $x_{n+1} = u$*, and* $x_i \neq x_{i+1}$ *for* $1 \leq i \leq n$*. A term* t *in* S *is* initial *(resp.* final*) if there is no path to* t *(resp. no path starting at* t*).*

Given sets B *and* H *of atoms, and a set* $\mathbf{x} \subseteq \mathbf{V}$ *of all variables in* $B \cup H$*, a* description logic rule (DL rule) *is a formula* $\forall \mathbf{x}. \bigwedge B \to \bigwedge H$ *such that*

R1 *for any* $u \in \mathsf{N}_I \cup \mathbf{V}$ *that is not initial in* B*, there is a path from exactly one initial* $t \in \mathsf{N}_I \cup \mathbf{V}$ *to* u *in* B*,*
R2 *for any* $t, u \in \mathsf{N}_I \cup \mathbf{V}$*, there is at most one path in* B *from* t *to* u*,*
R3 *if* H *contains an atom of the form* $C(t)$ *or* $R(t, u)$*, then* t *is initial in* B*.*

Here $\forall \mathbf{x}$ *for* $\mathbf{x} = \{x_1, \ldots, x_n\}$ *abbreviates an arbitrary sequence* $\forall x_1. \ldots \forall x_n$*. Since we consider only conjunctions with all variables quantified, we will often simply write* $B \to H$ *instead of* $\forall \mathbf{x}. \bigwedge B \to \bigwedge H$*.*

A rule base RB *for some DL* $\mathcal{L}$ *is a set of DL rules for* $\mathcal{L}$*.*

The semantics of DL rules in the context of a description logic knowledge base is given by interpreting both the rules and knowledge base as first-order theories in the usual way, and applying the standard semantics of predicate logic. This has been discussed in the context of SWRL in [4], and we will not repeat the details here.

Note that Definition 4 ensures that role atoms in rule bodies essentially form a (set of) directed trees, starting at initial elements. Since all but the first and last elements of a path must be variables, individuals effectively break paths apart. For example, the following might be the body of a DL rule if a and b are individual names: $\{R(x, a), S(a, z), S'(a, z'), T(z, b), T'(z', b)\}$. Using the well-known equivalence of formulae $\{p \to q_1 \wedge q_2\}$ and $\{p \to q_1, p \to q_2\}$, one can transform any rule into an equivalent set of rules without conjunctions in rule heads. Since this can be done in linear time, we will assume without loss of generality that all DL rules are of this form.

Moreover, since all DLs considered in this work support nominals, we will assume without loss of generality that all terms in rules are variables. Indeed, any atom $C(a)$ with $a \in \mathsf{N}_I$ can be replaced by $C(x) \wedge \{a\}(x)$ for some new variable $x \in \mathbf{V}$. In the presence of inverse roles, role atoms with individual names can be replaced by concept atoms as follows: $R(x, a)$ becomes $\exists R.\{a\}(x)$, $R(a, y)$ becomes $\exists\,\mathrm{Inv}(R).\{a\}(y)$, and $R(a, b)$ becomes $\exists R.\{b\}(x) \wedge \{a\}(x)$. A similar transformation is possible for rule heads, where generated concept atoms $\{a\}(x)$ are again added *to the rule body*.

Before proceeding with the formal treatment of DL rules in concrete description logics, let us consider some relevant special applications of DL rules.

Concept products Rules of the form $C(x) \wedge D(y) \to R(x, y)$ can encode *concept products* (sometimes written $C \times D \sqsubseteq R$) asserting that all elements of two classes must be related [10]. Examples include statements such as $\mathsf{Elephant}(x) \wedge \mathsf{Mouse}(y) \to \mathsf{biggerThan}(x, y)$ or $\mathsf{Alkaline}(x) \wedge \mathsf{Acid}(y) \to \mathsf{neutralises}(x, y)$.

Local reflexivity, universal role Rules of the forms $C(x) \to R(x, x)$ and $R(x, x) \to C(x)$ can replace the $\mathcal{SROIQ}$ Tbox expression $C \sqsubseteq \exists R.\mathsf{Self}$ and $\exists R.\mathsf{Self} \sqsubseteq C$. The universal role U of $\mathcal{SROIQ}$ can be defined as $\top(x) \wedge \top(y) \to U(x, y)$. Hence, a DL that permits such rules does not need to explicitly introduce those constructs.

Qualified RIAs DL rules of course can express arbitrary role inclusion axioms, but they also can state that a RIA applies only to instances of certain classes. Examples include $\mathsf{Woman}(x) \wedge \mathsf{hasChild}(x, y) \to \mathsf{motherOf}(x, y)$ and $\mathsf{trusts}(x, y) \wedge \mathsf{Doctor}(y) \wedge \mathsf{recommends}(y, z) \wedge \mathsf{Medicine}(z) \to \mathsf{buys}(x, z)$.

4 DL RULES IN $\mathcal{SROIQ}$

In this section, we show how knowledge bases of such rules can be completely internalised into the DL $\mathcal{SROIQ}$. First, however, we adopt the notions of *regularity* and *simplicity* to DL rule bases in $\mathcal{SROIQ}$.

Definition 5. *Consider a rule base* RB *and a knowledge base* KB *for* $\mathcal{SROIQ}$*. The set of* simple roles *of* KB ∪ RB *is the smallest set of roles containing every role* R *for which the following conditions hold:*

- *If* R *or* $\mathrm{Inv}(R)$ *occur on the right-hand-side of some RIA of* KB*, then this RIA is of the form* $S \sqsubseteq R$ *or* $S \sqsubseteq \mathrm{Inv}(R)$*, and* S *is simple.*
- *If* R *or* $\mathrm{Inv}(R)$ *occur in some rule head of the form* $R(x, y)$ *or* $\mathrm{Inv}(R)(x, y)$ *in* RB*, then the according rule body is of the form* $S(x, y)$ *with* S *simple, or of the form* $C(x)$ *where* $x = y$*.*

Note that this is indeed a proper inductive definition, where roles that do not occur on the right of either RIAs or rules form the base case. The extended knowledge base KB∪RB *is* admissible *for* $\mathcal{SROIQ}$ *if all roles* $S_{(i)}$ *occurring in concept (sub)expressions of the form* $\leq n\,S.C$*,* $\geq n\,S.C$*,* $\exists S.\mathsf{Self}$*, and* $\mathsf{Dis}(S_1, S_2)$*, and in role atoms of the form* $S(x, x)$ *(*$x \in \mathbf{V}$*) are simple.*

An extended knowledge base KB ∪ RB *is* regular *if there is a strict partial order* $\prec$ *on* $\mathbf{R}$ *such that*

- $S \prec R$ *iff* $\mathrm{Inv}(S) \prec R$*,*
- *the role box of* KB *is regular w.r.t.* $\prec$*, and*
- *for any rule* $B \to R(x, y)$*, each* $S(z, v) \in B$ *satisfies one of the following:*
  - $S \prec R$*, or*
  - *there is no path from* v *to* y*, or*
  - $S = R$*, there is no other* $R(z', v') \in B$ *with a path from* v' *to* y*, and we find that: either* $x = z$ *and there is no* $C(x) \in B$*, or* $y = v$ *and there is no* $C(y) \in B$*.*

Note that RIAs in regular $\mathcal{SROIQ}$ knowledge bases are allowed to have two special forms for transitivity and symmetry, which we do omit for the definition of regularity in DL rules to simplify notation. Since S in $S(x, x)$ is simple, we can replace such role atoms by concept atoms $C(x)$ where C is a new concept name for which a new axiom

$C \equiv \exists S.\mathsf{Self}$ is added. We will thus assume that no role atoms of this form occur in admissible knowledge bases.

In the remainder of this section, we show that checking the satisfiability of extended $\mathcal{SROIQ}$ knowledge bases that are admissible and regular is decidable, and has the same worst-case complexity as reasoning in $\mathcal{SROIQ}$. This is achieved by a polynomial transformation of rule bases into $\mathcal{SROIQ}$ axioms. The first step of doing this is to replace "dead branches" of the tree-shaped query body by DL concepts. The proof is a variation of the "rolling-up" technique used for conjunctive query answering [11].

Lemma 6. *Any DL rule $B \to H$ for $\mathcal{SROIQ}$ can be transformed into a semantically equivalent rule $B' \to H$ such that all paths in B' are contained in a single maximal path. If $H = R(x, y)$, then y is the final element of that maximal path, and if $H = C(x)$ then there are no paths in B. A rule with these properties is called* linearised.

Proof. We provide an iterative reduction algorithm for transforming B into B'. Initially, we set $B' := B$. Every iteration of the algorithm proceeds in two steps:

S1 For each variable $x \in \mathbf{V}$ in B', let $S = \{C_1(x), \ldots, C_n(x)\}$ be the set of all concept atoms in B' that refer to x, and set $B' := (B' \setminus S) \cup \{(C_1 \sqcap \ldots \sqcap C_n)(x)\}$.

S2 Let $R(x, y) \in B'$ be any atom where y is a final term in B such that H is not of the form $S(z, y)$ for a variable z. If no such atom exists, the algorithm terminates and returns B'. Otherwise, let D denote the (unique by S1) concept such that $D(y) \in B$, and let D denote $\top$ if no concept with variable y exists. Now B' is changed by setting $B' := (B' \setminus \{R(x, y), D(y)\}) \cup \{(\exists R.D)(x)\}$

Clearly, this algorithm terminates after a linear number of iterations, since it reduces the number of role atoms in B' in every non-final iteration. Moreover, B' after termination cannot contain final terms that are part of some path, unless they occur as the second argument of the rule head. Thus all paths, if any, end in this final element, and if $H = C(x)$ then all paths have been reduced.

This shows that the result has the required form. It remains to verify that $B' \to H$ is semantically equivalent to $B \to H$. By construction, the final variable y chosen for elimination is a variable that occurs in at most one concept atom but not in H (since it is neither the second term in H nor initial in B). Now it is easy to see that the computed rules before and after one iteration are semantically equivalent. By induction, the algorithm thus returns a rule that is semantically equivalent to its input. □

As an example, the DL rule that was given in the introduction can be simplified to yield:

$$\exists\,\mathsf{worksAt.University}(x) \wedge \mathsf{supervises}(x,z) \wedge \mathsf{PhDStudent}(z) \to \mathsf{profOf}(x,z)$$

The proof of Lemma 6 also shows that, in the presence of inverse roles, condition (R1) of Definition 4 can be relaxed as follows:

R1' for any $u \in \mathsf{N}_I \cup \mathbf{V}$ that is not initial in B, there is a path from one *or more* initial elements $t \in \mathsf{N}_I \cup \mathbf{V}$ to u in B.

Indeed, using inverse roles, one can eliminate those initial elements that are not required by (R3) just like the final elements in the above proof.

The above transformation allows us to reduce tree-shaped rules to rules of only linear structure that are much more similar to RIAs in $\mathcal{SROIQ}$. But while all role atoms now belong to a single maximal path, rules might still contain disconnected concept atoms. The rule $R(x,y) \wedge S(u,v) \wedge C(z) \rightarrow T(x,v)$, e.g., is rewritten to $\exists R.\top(x) \wedge S(u,v) \wedge C(z) \rightarrow T(x,v)$.

We now show that DL rules in $\mathcal{SROIQ}$ can indeed be internalised.

Theorem 7. *Consider a rule base* RB *and a knowledge base* KB *for $\mathcal{SROIQ}$, such that* RB $\cup$ KB *is admissible. There is a $\mathcal{SROIQ}$ knowledge base* $\mathrm{KB}_{\mathrm{RB}}$ *that can be computed in time polynomial in the size of* RB*, such that* KB $\cup$ RB *and* KB $\cup$ $\mathrm{KB}_{\mathrm{RB}}$ *are equisatisfiable.*

Moreover, if KB $\cup$ RB *is regular, then* KB $\cup$ $\mathrm{KB}_{\mathrm{RB}}$ *is also regular.*

Proof. We can assume that all rules in RB are in the form defined in Lemma 6. Indeed, the transformation used in this lemma preserves simplicity of roles in KB $\cup$ RB, since it only affects rules entailing non-simple roles. Moreover, since the transformation may only remove role atoms from rule bodies, it also preserves regularity of KB $\cup$ RB.

We can assume without loss of generality that RB contains no rule with the universal role U in its head – clearly such rules are tautological (yet, they would formally violate the requirement of regularity given the below transformations).

Rules can easily be transformed into an equivalent rule such that all variables occurring in rule heads do also occur in the according rule bodies, by simply adding atoms $\top(x)$ to the body if required. Moreover, for any rule $B \rightarrow R(x,y)$, Lemma 6 asserts that B contains at most one maximal path with final element y, and all role atoms of B (if any) are part of that path. Let z be the initial element of the maximal path if it exists, and let z be y otherwise. Now if $x \neq z$, then x occurs in B only in concept atoms $C(x)$, and we can add a role atom $U(x,z)$ to B without violating (R1)–(R3). Moreover, this change preserves the semantics of the rule since $U(x,z)$ is true for any variable assignment (mapping free variables to domain elements of $\mathcal{I}$; sometimes also called *variable binding* [4]) in any interpretation. Regularity of the role base is preserved since we can assume without loss of generality that U is the least element of $\prec$ (which is feasible since U does not occur in rule heads). Simplicity is not a concern since R by assumption is not a simple role in KB $\cup$ RB. In summary, we can assume that the body of any rule with head $R(x,y)$ has been transformed to contain exactly one maximal path starting at x and leading to y.

We now describe the step-wise computation of $\mathrm{KB}_{\mathrm{RB}}$. Initially, we set $\mathrm{KB}_{\mathrm{RB}} := \emptyset$, and define the set of remaining rules as $\mathrm{RB}' := \mathrm{RB}$. The reduction proceeds iteratively until RB' is empty. In every step, we select some rule $B \rightarrow H \in \mathrm{RB}$. Note that by Lemma 6, there is only a single maximal path of roles in B, all role atoms in B are part of that path, and all but adjacent variables in the path are distinct (there are no cycles). We distinguish various cases:

(1) If B contains two concept atoms $D(z)$ and $D'(z)$ referring to the same variable z, then both atoms are deleted from B and a new atom $(D \sqcap D')(z)$ is added.

(2) Otherwise, if $H = C(x)$ and $B = D(x)$, then $B \to H$ is removed from RB′, and a Tbox axiom $D \sqsubseteq C$ is inserted into $\mathrm{KB}_{\mathrm{RB}}$.
(3) Otherwise, if $H = R(x, y)$ and B is of the form $\{R_1(x, x_2), \ldots, R_n(x_n, y)\}$, then $B \to H$ is removed from RB′, and an Rbox axiom $R_1 \circ \ldots \circ R_n \sqsubseteq R$ is inserted into $\mathrm{KB}_{\mathrm{RB}}$.
(4) Otherwise, if $H = R(x, y)$, and there is some $D(z) \in B$ such that z occurs in some role atom of B or H (in first or second argument position), then the following is done. First, a new role name S is introduced, and the Tbox axiom $D \equiv \exists S.\mathsf{Self}$ is added to $\mathrm{KB}_{\mathrm{RB}}$. Second, a new variable $z' \in \mathbf{V}$ is introduced, the role atom $S(z, z')$ is added to B, every role atom $T(x', z) \in B$ is replaced by $T(x', z')$, and every role atom $T(z, y') \in B$ is replaced by $T(z', y')$. Finally, the atom $D(z)$ is removed from B, and if $z = y$ then the rule head is replaced by $R(x, z')$.
(5) Otherwise, if $H = C(x)$ or $H = R(x, y)$, and there is some $D(z) \in B$ such that z occurs neither in H nor in any role atom of B, then the following is done. If B contains some atom of the form $R(x, t)$ so there is no atom of the form $D'(x) \in B$, then define $u := y$; otherwise define $u := x$. Now $D(z)$ in B is replaced by the concept atom $\exists U.D(u)$.

We verify the correctness of the algorithm in multiple steps.

Claim 1 The cases distinguished by the algorithm are exhaustive.

We need to show that all cases that do not satisfy the precondition of case (1)–(4) must satisfy the conditions of (5). If $H = C(x)$, then the non-applicability of (1) and (2) ensure that a required $D(z) \in B$ exists, and by Lemma 6 there are no role atoms in B at all. Otherwise, if $H = R(x, y)$, then non-applicability of (3) ensures that there is some concept atom $D(z) \in B$: initially and in each construction step, role atoms are always required to form a chain as in (3). But then either (4) or (5) must be applicable.

Claim 2 The algorithm terminates after a polynomial number of steps.

(1) and (4) strictly reduce the number of concept atoms for a rule. Since no step increases the number of such atoms in a rule, (1) and (4) can only applied once for any concept atom occurring in any role. (2) and (3) reduce the number of rules, and again this can happen only a linear number of times. Finally, (5) reduces the number of concept atoms that do not contain variables that occur in the head. Again, no other step introduces such atoms and hence (5) is applicable only a linear number of times.

Claim 3 The computed knowledge base $\mathrm{KB} \cup \mathrm{KB}_{\mathrm{RB}}$ is a $\mathcal{SROIQ}$ knowledge base.

We need to verify the correct use of simple and non-simple roles in all axioms. First note that (4) is the only case where new concept expressions are introduced that might violate simplicity restrictions. However, since the involved roles S are new, they are trivially simple. It remains to verify that all transformations preserve simplicity of roles, i.e. that all roles that are simple in $\mathrm{KB} \cup \mathrm{RB}$ are also simple in $\mathrm{KB} \cup \mathrm{KB}_{\mathrm{RB}}$. This is obvious since simple roles can occur only in rules that are transformed by (2) without prior modifications.

Claim 4 After termination of the algorithm, $\mathrm{KB}\cup\mathrm{RB}$ and $\mathrm{KB}\cup\mathrm{KB}_{\mathrm{RB}}$ are equisatisfiable.

The claim follows by induction if every single step preserves satisfiability. Hence let $\mathrm{KB}_0/\mathrm{RB}_0$ and $\mathrm{KB}_1/\mathrm{RB}_1$ be the sets $\mathrm{KB}_{\mathrm{RB}}/\mathrm{RB}'$ before and after the application of one transformation step. We need to show that $\mathrm{KB} \cup \mathrm{KB}_0 \cup \mathrm{RB}_0$ and $\mathrm{KB} \cup \mathrm{KB}_1 \cup \mathrm{RB}_1$ are equisatisfiable.

The cases (1), (2), and (3) clearly yield semantically equivalent results.

For case (4), we find that $\mathrm{KB}_1 = \mathrm{KB}_0 \cup \{D \equiv \exists S.\mathsf{Self}\}$. Clearly, $\mathrm{KB} \cup \mathrm{KB}_0 \cup \mathrm{RB}_0$ and $\mathrm{KB} \cup \mathrm{KB}_1 \cup \mathrm{RB}_0$ are equisatisfiable since S is new. We show that $\mathrm{KB} \cup \mathrm{KB}_1 \cup \mathrm{RB}_0$ and $\mathrm{KB} \cup \mathrm{KB}_1 \cup \mathrm{RB}_1$ are equivalent. To this end, first observe that for any model $\mathcal{I}$ of KB_1, we find that $S^{\mathcal{I}} = \{\langle \delta, \delta \rangle \mid \delta \in D^{\mathcal{I}}\}$. Now let $B_0 \rightarrow H_0$ and $B_1 \rightarrow H_1$ denote the transformed rule before and after the translation step.

For the one direction, consider some interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathrm{KB} \cup \mathrm{KB}_1 \cup \mathrm{RB}_0$. Thus, for all variable assignments Z, we find $(B_0 \rightarrow H_0)^{\mathcal{I},Z} = \mathit{true}$ (where we silently equate each set of atoms with the conjunction of its elements). If Z is such that $Z(z) \neq Z(z')$ or $Z(z) \notin D^{\mathcal{I}}$, then $B_1^{\mathcal{I},Z} = \mathit{false}$ and we find $(B_1 \rightarrow H_1)^{\mathcal{I},Z} = \mathit{true}$. Otherwise, we have that $B_0^{\mathcal{I},Z} = \mathit{true}$ by the construction of B_1, and hence $H_0^{\mathcal{I},Z} = \mathit{true}$ by assumption. But then again $H_1^{\mathcal{I},Z} = \mathit{true}$ and $(B_1 \rightarrow H_1)^{\mathcal{I},Z} = \mathit{true}$ as required. This shows that $\mathcal{I}$ is a model of $B_1 \rightarrow H_1$. Since all other formulae in $\mathrm{KB} \cup \mathrm{KB}_1 \cup \mathrm{RB}_0$ and $\mathrm{KB} \cup \mathrm{KB}_1 \cup \mathrm{RB}_1$ agree, we find that $\mathcal{I} \models \mathrm{KB} \cup \mathrm{KB}_1 \cup \mathrm{RB}_1$ as required.

For the other direction, assume that $\mathcal{I} \models \mathrm{KB} \cup \mathrm{KB}_1 \cup \mathrm{RB}_1$, and again consider any variable assignment Z such that $B_0^{\mathcal{I},Z} = \mathit{true}$. A variable assignment Z' is defined by setting $Z'(z') := Z(z)$, and $Z'(x) := Z(x)$ for all $x \neq z'$. It is easy to see that $B_1^{\mathcal{I},Z'} = \mathit{true}$ and hence $H_1^{\mathcal{I},Z'} = \mathit{true}$ by assumption. As before, we conclude that $H_0^{\mathcal{I},Z'} = \mathit{true}$. But since Z' agrees with Z on all variables occurring in H_0, this implies $H_0^{\mathcal{I},Z} = \mathit{true}$ and hence we find $\mathcal{I} \models B_0 \rightarrow H_0$ as required. This finishes case (4).

For case (5), we use again $B_0 \rightarrow H$ and $B_1 \rightarrow H$ to denote the modified rule before and after the translation step, and let $\mathcal{I}$ be any interpretation. For the first direction, assume that $\mathcal{I} \models B_0 \rightarrow H$. Now consider any variable assignment Z such that $B_1^{\mathcal{I},Z} = \mathit{true}$. Then, using the notation of (5), $\exists U.D(u)^{\mathcal{I},Z} = \mathit{true}$. Especially, there is some domain element $\delta \in \Delta^{\mathcal{I}}$ such that $\delta \in D^{\mathcal{I}}$. A variable assignment Z' is obtained by setting $Z'(z) := \delta$, and $Z'(x) := Z(x)$ for all $x \neq z$. Then $D(z)^{\mathcal{I},Z'} = \mathit{true}$ and, since z does not occur in any other atom (by non-applicability of (1) and the precondition of (5)) we also find $B_0^{\mathcal{I},Z'} = \mathit{true}$. But then $H^{\mathcal{I},Z'} = H^{\mathcal{I},Z} = \mathit{true}$ by assumption, which shows the required $\mathcal{I} \models B_1 \rightarrow H$.

For the other direction, assume that $\mathcal{I} \models B_1 \rightarrow H$, and consider a variable assignment Z such that $B_0^{\mathcal{I},Z} = \mathit{true}$. Then $D(z)^{\mathcal{I},Z} = \mathit{true}$, and we find that $Z(z) \in D^{\mathcal{I}}$. But then $\exists U.D(u)^{\mathcal{I},Z}$ for any variable u, and hence $B_1^{\mathcal{I},Z} = \mathit{true}$. Again this implies $H^{\mathcal{I},Z} = \mathit{true}$ and we conclude $\mathcal{I} \models B_0 \rightarrow H$.

Claim 5 If $\mathrm{KB} \cup \mathrm{RB}$ is regular, then so is $\mathrm{KB} \cup \mathrm{KB}_{\mathrm{RB}}$.

By Definition 5, the RIA created in case (3) satisfies all conditions of regularity as long as the transformed rule $B \rightarrow H$ did (where one might use the same ordering $\prec$). Since regularity clearly is not affected by cases (1) and (2), it remains to show that (4) and (5) preserve regularity of the extended knowledge base.

For case (4) this is indeed the case, since the new role S can by chosen to be $\prec$-smaller than the role R in the rule head. Then regularity can only be affected if S introduces a new initial or final element to the maximal path in B, where a role atom $R(s,t)$ had been in an initial or final position before. However, in this case the reduced concept atom $D(z)$ would be of the form $D(x)$ or $D(y)$, and in both cases adding $S(z_1, z_2)$ does not affect regularity by Definition 5.

For case (5) the claim again follows since adding a concept $\exists U.D(u)$ can affect regularity only if B contains an atom $R(s,t)$ that forms the first or last segment of the maximal path. If $R(s,t)$ is the first segment, then u is chosen to be y and hence preserves regularity. Otherwise $R(s,t)$ must be the final segment, and by setting $u = x$ regularity again is preserved.

The above claims together yield the required proof. □

Considering again our introductory example, we arrive at the following $\mathcal{SROIQ}$ axioms (where S_1, S_2 are new auxiliary roles):

$$S_1 \circ \mathsf{supervises} \circ S_2 \sqsubseteq \mathsf{profOf}$$

$$\exists\,\mathsf{worksAt.University} \equiv \exists S_1.\mathsf{Self} \qquad \mathsf{PhDStudent} \equiv \exists S_2.\mathsf{Self}$$

Based on Theorem 7, we conclude that the problem of checking the satisfiability of $\mathcal{SROIQ}$ knowledge bases extended with DL rules is decidable, as long as the extended knowledge base is admissible and regular. Since the internalisation is possible in polynomial time, the worst-case complexity for this problem is the same as for checking satisfiability of $\mathcal{SROIQ}$ knowledge bases.

5 DL RULES IN $\mathcal{EL}^{++}$

In this section, we investigate DL rules for the DL $\mathcal{EL}^{++}$ [2], for which many typical inference problems can be solved in polynomial time. As $\mathcal{EL}^{++}$ cannot internalise DL rules, they constitute a true extension of expressivity. We therefore take a different approach than in $\mathcal{SROIQ}$: instead of considering rule bases as an auxiliary set of axioms that is successively reduced and internalised, we introduce DL rules as core expressive mechanism to which all other $\mathcal{EL}^{++}$ axioms can be reduced. While $\mathcal{EL}^{++}$ rule bases offer many expressive features formerly unavailable in $\mathcal{EL}^{++}$, we show that the complexity of core inference problems remains tractable. We simplify our presentation by omitting concrete domains from $\mathcal{EL}^{++}$ – they are not affected by our extension and can be treated as shown in [2].

Definition 8. *A role of $\mathcal{EL}^{++}$ is a (non-inverse) role name. An $\mathcal{EL}^{++}$ Rbox is a set of generalised role inclusion axioms, and an $\mathcal{EL}^{++}$ Tbox (Abox) is a $\mathcal{SROIQ}$ Tbox (Abox) that contains only the following concept constructors: $\sqcap$, $\exists$, $\top$, $\bot$, as well as nominal classes $\{a\}$. An $\mathcal{EL}^{++}$* knowledge base *is the union of an $\mathcal{EL}^{++}$ Rbox, Tbox and Abox. An $\mathcal{EL}^{++}$* rule base *is a set of DL rules for $\mathcal{EL}^{++}$ that do not contain atoms of the form $R(x,x)$ in the body.*

Note that we do not have any requirement for regularity or simplicity of roles in the context of $\mathcal{EL}^{++}$. It turns out that neither is relevant for obtaining decidability or tractability. The case of $R(x,x)$ in bodies is not addressed by the below algorithm – [12] significantly extends the below approach to cover this and other features. Since it is obvious that both concept and role inclusion axioms can directly be expressed by DL rules, we will consider only $\mathcal{EL}^{++}$ rule bases without any additional $\mathcal{EL}^{++}$ knowledge base axioms. We can restrict our attention to $\mathcal{EL}^{++}$ rules in a certain normal form:

Definition 9. *An $\mathcal{EL}^{++}$ rule base* RB *is in* normal form *if all concept atoms in rule bodies are either concept names or nominals, all variables in a rule's head also occur in its body, and all rule heads are of one of the following forms:*

$$A(x) \qquad \exists R.A(x) \qquad R(x,y)$$

where $A \in \mathsf{N}_C \cup \{\{a\} \mid a \in \mathsf{N}_I\} \cup \{\top, \bot\}$ *and* $R \in \mathsf{N}_R$*. A set* $\mathcal{B}$ *of* basic concept expressions *for* RB *is defined as* $\mathcal{B} := \{C \mid C \in \mathsf{N}_C, C$ *occurs in* RB$\} \cup \{\{a\} \mid a \in \mathsf{N}_I, a$ *occurs in* RB$\} \cup \{\top, \bot\}$.

Proposition 10. *Any $\mathcal{EL}^{++}$ rule base can be transformed into an equisatisfiable $\mathcal{EL}^{++}$ rule base in normal form. The transformation can be done in polynomial time.*

Proof. First note that, since $\mathcal{EL}^{++}$ supports no inverse roles, individual names in rule heads cannot always be reduced as described in Section 3. We will therefore assume that, initially, rules in RB may contain role atoms of the form $R(a, x)$ with $a \in \mathsf{N}_I$ (while all other individual occurrences have been removed as describe before).

The transformation algorithm iteratively transforms RB. In each iteration, a rule $B \to H$ that is not in normal form yet is selected from RB, and one of the following is done:

- if H is of the form $R(a,y)$ with $a \in \mathsf{N}_I$, then $B \to H$ is replaced by the rule $B \cup \{\{a\}(x)\} \to R(x,y)$ where $x \in \mathbf{V}$ is new,
- if H is of the form $\exists R.C(x)$ with non-basic $C \notin \mathcal{B}$, then the rule $B \to H$ is replaced by two new rules $B \to \exists R.A(x)$ and $A(x) \to C(x)$ where $A \in \mathsf{N}_C$ is new,
- if H is of the form $(C \sqcap D)(x)$, then the rule $B \to H$ is replaced by two new rules $B \to C(x)$ and $B \to D(x)$,
- if B contains an atom of the form $\exists R.C(x)$, it is replaced by two new atoms $R(x,y)$ and $C(y)$ where $y \in \mathbf{V}$ is new,
- if B contains an atom of the form $(C \sqcap D)(x)$, it is replaced by two new atoms $C(x)$ and $D(x)$,
- if B contains an atom of the form $R(a,y)$ with $a \in \mathsf{N}_I$, it is replaced by two new atoms $R(x,y)$ and $\{a\}(x)$ where $x \in \mathbf{V}$ is new.

It is easy to see that the transformation yields an equisatisfiable $\mathcal{EL}^{++}$ rule base in normal form, the size of which is polynomial in the size of the original rule base. □

When checking satisfiability of $\mathcal{EL}^{++}$ rule bases, we can thus restrict to rule bases in the above normal form. A polynomial algorithm for checking class subsumptions in $\mathcal{EL}^{++}$ knowledge bases has been given in [2], and it was shown that other standard inference problems can easily be reduced to that problem. We now present a new algorithm for checking satisfiability of $\mathcal{EL}^{++}$ rule bases, and show its correctness and tractability. Clearly, subsumption checking can be reduced to this problem: given a new individual $a \in \mathsf{N}_I$, the rule base $\mathrm{RB} \cup \{C(a), \{a\}(x) \sqcap D(x) \to \bot(x)\}$ is unsatisfiable iff RB entails $C \sqsubseteq D$. Instance checking in turn is directly reducible to subsumption checking in the presence of nominals.

Algortihm 1. *The algorithm proceeds by computing two sets: a set $\mathcal{E}$ of inferred "domain elements", and a set $\mathcal{S}$ of relevant subclass inclusion axioms that are entailed by*

RB. *The elements of $\mathcal{E}$ are represented by basic concept expressions of* RB, *i.e.* $\mathcal{E} \subseteq \mathcal{B}$, *and the inclusion axioms in $\mathcal{S}$ are of the form $C \sqsubseteq D$ or $C \sqsubseteq \exists R.D$, where $C, D \in \mathcal{E}$. Hence, both $\mathcal{E}$ and $\mathcal{S}$ are polynomially bounded by the size of* RB.

Initially, we set $\mathcal{E} := \{\{a\} \mid \{a\} \in \mathcal{B}\} \cup \{\top\}$ and $\mathcal{S} := \emptyset$. Now a DL rule is applied whenever we find that there is a match *with the rule body. Given a rule $B \to H$, a match θ is a mapping from all variables in B to elements of $\mathcal{E}$, such that the following hold:*

- *for every $C(y) \in B$, $\theta(y) \sqsubseteq C \in \mathcal{S}$, and*
- *for every $R(y, z) \in B$, $\theta(y) \sqsubseteq \exists R.\theta(z) \in \mathcal{S}$.*

An algorithm for partially computing matches is given below. The algorithm now proceeds by applying the following rules until no possible rule application further modifies the set $\mathcal{E}$ or $\mathcal{S}$:

(EL1) *If $C \in \mathcal{E}$, then $\mathcal{S} := \mathcal{S} \cup \{C \sqsubseteq C, C \sqsubseteq \top\}$.*
(EL2) *If there is a rule $B \to E(x) \in$* RB, *and if there is a match θ for B with $\theta(x) = \theta_x$, then $\mathcal{S} := \mathcal{S} \cup \{\theta_x \sqsubseteq E\}$. In this case, if $E = C$ or $E = \exists R.C$, then $\mathcal{E} := \mathcal{E} \cup \{C\}$.*
(EL3) *If there is a rule $B \to R(x, y) \in$* RB, *and if there is a match θ for B with $\theta(x) = \theta_x$ and $\theta(y) = \theta_y$, then $\mathcal{S} := \mathcal{S} \cup \{\theta_x \sqsubseteq \exists R.\theta_y\}$.*
(EL4) *If $\{C \sqsubseteq \{a\}, D \sqsubseteq \{a\}, D \sqsubseteq E\} \subseteq \mathcal{S}$ then $\mathcal{S} := \mathcal{S} \cup \{C \sqsubseteq E\}$.*

Here we assume that $C, D, D' \in \mathcal{B}$, $E \in \mathcal{B} \cup \{\exists R.C \mid C \in \mathcal{B}\}$, and $R \in \mathsf{N}_R$. After termination, the algorithm returns "unsatisfiable" if $\bot \in \mathcal{E}$, and "satisfiable" otherwise.

Assuming that all steps of Algorithm 1 are computable in polynomial time, it is easy to see that the algorithm also terminates in polynomial time, since there are only polynomially many possible elements for $\mathcal{E}$ and $\mathcal{S}$, and each case adds new elements to either set. However, we still have not verified that individual steps can be computed efficiently, and in particular this is not obvious for the match-checks in (EL2) and (EL3). Indeed, finding matches in query graphs is known to be NP-complete in general, and the tree-like structure of queries is crucial to retain tractability. Moreover, even tree-like rule bodies admit exponentially many matches. But note that Algorithm 1 does not consider *all matches* but only the (polynomially many) possible values of θ_x (and θ_y). We will now specify an algorithm that checks in polynomial time whether a match θ as in (EL2) and (EL3) exists. Naturally, this is closely related to the general task of testing the existence of homomorphisms between trees and graphs.

Proposition 11. *Consider a rule of the form $B \to C(x)$ ($B \to R(x, y)$), sets $\mathcal{E}$ and $\mathcal{S}$ as in Algorithm 1, and an element $\theta_x \in \mathcal{E}$ (elements $\theta_x, \theta_y \in \mathcal{E}$). There is an algorithm that decides whether there is a match θ such that $\theta(x) = \theta_x$ ($\theta(x) = \theta_x$ and $\theta(y) = \theta_y$), running in polynomial time w.r.t. the size of the inputs.*

Proof. We first specify a suitable algorithm, which works by propagating restrictions along the paths of the body B. For every variable x in B, a set $\Theta(x)$ of possible values is computed. Initially, we set $B' := B$, and $\Theta(x) := \mathcal{E}$ for all x. While B' is non-empty, the algorithm does the following:

- Select a variable z that is final in B'.

- If there is some atom $D(z) \in B'$, select some such $D(z)$. Then set $\Theta(z) := \Theta(z) \cap \{D' \mid D' \sqsubseteq D \in \mathcal{S}\}$ and $B' := B' \setminus \{D(z)\}$.
- If there is some atom $S(z', z) \in B'$, select some such $S(z', z)$. Then set $\Theta(z') := \Theta(z') \cap \{D \mid D \sqsubseteq \exists S.D' \in \mathcal{S}$ for some $D' \in \Theta(z)\}$ and $B' := B' \setminus \{S(z', z)\}$.

Finally, if $\theta_x \notin \Theta(x)$ or $\Theta(z) = \emptyset$ for some variable z in B, then the algorithm returns *false* (i.e. no according match exists). Otherwise, if H is of the form $C(x)$, the algorithm returns *true*.

Otherwise, H is of the form $R(x, y)$. The algorithm sets $\Theta(x) := \{\theta_x\}$. If B contains some path $R_0(x_0, x_1) \ldots R_n(x_n, x_{n+1})$ with $x_{n+1} = y$ and x_0 initial in B, then, for $i = 1$ to n, do the following:

- Set $\Theta(x_i) := \Theta(x_i) \cap \{D \mid D' \sqsubseteq \exists R_{i-1}.D \in \mathcal{S}$ for some $D' \in \Theta(x_{i-1})\}$.

Finally, the algorithm returns *true* if $\theta_y \in \Theta(y)$, and it returns *false* otherwise.

Claim 1 The algorithm terminates after polynomially many steps.

In the first processing stage, every iteration removes some atom from B', and hence there are only a linear number of steps. Note that the algorithm is guaranteed to terminate, i.e. that every atom must be processed at some point, since $B \to H$ is a DL rule. Selecting some final variable z is naively possible by checking, for all variables z, whether some atom $S(z, z')$ exists in B' or not (note that B' contains only variables as terms as it is normalised). One can obviously find an atom $D(z)$ or $S(z, z')$ that is to be reduced next in linear time. It remains to check that the computations for $\Theta(z)$ and $\Theta(z')$ can be done in polynomial time. This follows since the intersections of polynomially large sets can be computed in polynomial time, where we note that $\Theta(z) \subseteq \mathcal{E}$ is bounded by the size of $\mathcal{E}$, and that the intersected sets can be computed by a linear number of comparisons with elements of $\mathcal{S}$.

For the case $H = R(x, y)$, one first needs to find a (unique) path from some initial x_0 to y. The length $n + 1$ of this path is bounded by the size of B, and one can construct the path backwards starting from y, where each next section can be found by a linear number of comparisons with role atoms of B. The $n + 1$ iterations of i can again be performed in polynomial time each, where three polynomially large sets are intersected in each computation step.

Claim 2 If there is a match θ with the required properties, then the algorithm returns *true*.

Let θ be the required match with $\theta(x) = \theta_x$ (and $\theta(y) = \theta_y$). We first show that, throughout the first processing stage, $\theta(z) \subseteq \Theta(z)$ for any variable z in B. Initially this is clearly the case, as $\theta(z) \in \mathcal{E}$ by definition. For the induction step, it suffices to note that $\theta(z) \in \{D' \mid D' \sqsubseteq D \in \mathcal{S}\}$ whenever $D(z) \in B$ to obtain the result for reduction of concept atoms. The case of role atoms is similar, and we thus conclude that $\theta_x = \theta(x) \in \Theta(x)$ and $\Theta(z) \neq \emptyset$ for all z after the first processing stage.

In the case $H = R(x, y)$, we can continue the above inductive argument. Clearly, setting $\Theta(x) := \theta_x = \theta(x)$ preserves the claimed property. For the iteration, we can again observe that, for any variable z, the value of $\theta(z)$ is contained in the sets intersected when computing $\Theta(z)$. Hence we obtain $\theta_y = \theta(y) \in \Theta(y)$ as required.

Claim 3 If the algorithm returns *true*, then there is a match θ with the required properties.

After the completion of the first processing stage, we construct a match θ as follows. For each variable z that is initial in B, select $\theta(z)$ to be any element of $\Theta(z)$, which must exists since $\Theta(z) \neq \emptyset$ for all z. All other values of θ are defined iteratively:

(a) Select some variable z such that $\theta(z)$ is yet undefined, but there is some atom $S(z', z) \in B$ such that $\theta(z')$ is defined.
(b) Select $\theta(z)$ to be any element of the set $\Theta(z) \cap \{D \mid \theta(z') \sqsubseteq \exists S.D \in \mathcal{S}\}$.

We claim that this defines a match θ for B. First note that each variable z in B will indeed be considered in the iteration, based on property (R1) of DL rules, and that the selected atom $S(z', z)$ is unique by (R2). Second, we claim that the intersection in (b) is necessarily non-empty. Indeed, since $S(z', z)$ must have been considered in the iteration on B', we know that for any $D \in \Theta(z')$ there is some $D \sqsubseteq \exists S.D' \in \mathcal{S}$ with $D' \in \Theta(z)$. Note that the set $\Theta(z)$ is not changed at any point after the processing of $S(z', z)$, and hence we still find some element $\theta(z) \in \Theta(z)$ with the required property.

Finally, we show that θ is a match. The according condition is clearly satisfied for all concept atoms $D(z)$, since they were explicitly checked for all elements in $\Theta(z)$ when processing this atom. For the case of role atoms, the matching condition follows directly from (b).

This settles the case for $H = C(x)$. For $H = R(x, y)$, note that the final computation of $\Theta(y)$ is similar to the iterative construction of θ above, where we consider only one initial element x_0 (which exists due to (R1)), and where all possible choices for each $\theta(z)$ are considered. So, if $\theta_y \in \Theta(y)$, then there is a way of constructing θ as above so that $\theta(y) = \theta_y$. This finishes the claim and the proof. □

We can now proceed to show correctness and tractability of Algorithm 1.

Lemma 12. *Algorithm 1 terminates after polynomial time w.r.t. the size of the considered rule base.*

Proof. As argued above, the algorithm can perform only a polynomial number of iterations due to the restricted size of $\mathcal{E}$ and $\mathcal{S}$. Steps (EL1) and (EL4) clearly can be performed in polynomial time. For (EL2) and (EL3), Proposition 11 asserts that it can be decided in polynomial time whether there is some match θ such that $\theta(x) = \theta_x$ (and $\theta(y) = \theta_y$). Since there are only polynomially many possible choices of θ_x (and θ_y), the preconditions of (EL2) and (EL3) can thus be checked in polynomial time as required. □

Lemma 13. *For any normalised $\mathcal{EL}^{++}$ rule base* RB, *Algorithm 1 returns "unsatisfiable" only if* RB *is unsatisfiable.*

Proof. We claim that, for any interpretation $\mathcal{I}$ with $\mathcal{I} \models \text{RB}$, we have that $\mathcal{I} \models \mathcal{S}$ and $C^{\mathcal{I}} \neq \emptyset$ for each $C \in \mathcal{E}$. We proceed by induction. The base case is obvious, since $\top$ and all nominal classes must not be empty. For the induction step, we consider each derivation rule separately.

For (EL1) the claim is immediate, since all generated statements are tautologies.

For (EL2) and (EL3), we first show the following auxiliary claim ($*$). Given some match θ for a rule $B \to H$, let Z be any variable assignment for $\mathcal{I}$ such that $Z(x) \in \theta(x)^{\mathcal{I}}$ for all x. Then we find that $B^{\mathcal{I},Z} = \mathit{true}$. Indeed, for any concept atom $C(x) \in B$, we have $\theta(x) \sqsubseteq C \in \mathcal{S}$ (since θ is a match) and thus $\mathcal{I} \models \theta(x) \sqsubseteq C$ by induction hypothesis. But then also $Z(x) \in \theta(x)^{\mathcal{I}} \subseteq C^{\mathcal{I}}$. The case of role atoms $R(x, y)$ is similar.

Now consider a rule $B \to E(x)$, a match θ, and concept expression θ_x as in (EL2). For any $\delta \in \theta_x^{\mathcal{I}}$, there is some variable assignment Z such that $Z(x) = \delta$ and $Z(z) \in \theta(z)^{\mathcal{I}}$ for all variables z. This follows from the induction hypothesis that $C^{\mathcal{I}} \neq \emptyset$ for each $C \in \mathcal{E}$, since $\theta(z) \in \mathcal{E}$. Using ($*$) we conclude that for any such Z, we have $B^{\mathcal{I},Z} = \mathit{true}$, and therefore also $E(x)^{\mathcal{I},Z} = \mathit{true}$ since $\mathcal{I} \models B \to E(x)$. Thus, for any $\delta \in \theta_x^{\mathcal{I}}$, we conclude that $\delta \in E^{\mathcal{I}}$, and thus $\mathcal{I} \models \theta_x \sqsubseteq E$ as claimed. Moreover, this ensures that $E^{\mathcal{I}} \neq \emptyset$ and, if $E = \exists R.C$, also $C^{\mathcal{I}} \neq \emptyset$. This shows the claim of the induction for $\mathcal{E}$ and $\mathcal{S}$.

The case for (EL3) is similar to (EL2).

Finally consider case (EL4). It is easy to see that all basic concept expressions occurring in axioms of $\mathcal{S}$ are also contained in $\mathcal{E}$. Hence, C and D in (EL4) are non-empty in $\mathcal{I}$, and thus $C^{\mathcal{I}} = D^{\mathcal{I}} = \{a^{\mathcal{I}}\}$. From this the induction claim on $\mathcal{S}$ is immediate.

In summary we have shown that, whenever $\bot \in \mathcal{E}$, we find that $\bot^{\mathcal{I}} \neq \emptyset$ for each model $\mathcal{I}$ of RB. Since this cannot be, this shows the claimed unsatisfiability of RB. □

Lemma 14. *For any normalised $\mathcal{EL}^{++}$ rule base* RB*, Algorithm 1 returns "unsatisfiable" whenever* RB *is unsatisfiable.*

Proof. We show the contrapositive: if the algorithm does not return "unsatisfiable" then there is some interpretation $\mathcal{I}$ that is a model of RB. The proof proceeds by constructing this model.

The domain $\Delta^{\mathcal{I}}$ of $\mathcal{I}$ is chosen to consist of the set of computed elements $\mathcal{E}$, factorised to take inferred equalities into account. To this end, a binary relation $\sim$ on $\mathcal{E}$ that will serve us to represent inferred equalities is defined as follows:

$$C \sim D \quad \text{iff} \quad C = D \text{ or } \{C \sqsubseteq \{a\}, D \sqsubseteq \{a\}\} \subseteq \mathcal{S} \text{ for some } a \in \mathsf{N}_I.$$

We show that $\sim$ is an equivalence relation on $\mathcal{E}$. Reflexivity and symmetry are obvious. For transitivity, we first note that elements related by $\sim$ are subject to the same assertions in $\mathcal{S}$. Indeed, rule (EL4) allows us to conclude that, for any $C, C' \in \mathcal{E}$ with $C \sim C'$, $C \sqsubseteq E \in \mathcal{S}$ implies $C' \sqsubseteq E \in \mathcal{S}$ ($*$).

This also yields transitivity of $\sim$, since $\{C_1 \sqsubseteq \{a\}, C_2 \sqsubseteq \{a\}\} \subseteq \mathcal{S}$ and $C_2 \sim C_3$ implies $C_3 \sqsubseteq \{a\} \in \mathcal{S}$ and thus $C_1 \sim C_3$. We use $[C]$ to denote the equivalence class of $C \in \mathcal{E}$ w.r.t. $\sim$.

These observations allow us to make the following definition of $\mathcal{I}$:

- $\Delta^{\mathcal{I}} := \{[C] \mid C \in \mathcal{E}\}$,
- $C^{\mathcal{I}} := \{[D] \in \Delta^{\mathcal{I}} \mid D \sqsubseteq C \in \mathcal{S}\}$ for all $C \in \mathsf{N}_C$,
- $a^{\mathcal{I}} := [\{a\}]$ for all $\{a\} \in \mathcal{B}$, and $a^{\mathcal{I}} := [\top]$ for all $\{a\} \notin \mathcal{B}$,
- $R^{\mathcal{I}} := \{\langle [C], [D] \rangle \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid C \sqsubseteq \exists R.D \in \mathcal{S}\}$ for all $R \in \mathsf{N}_R$.

Roles and concepts not involved in $\mathcal{E}$ or $\mathcal{S}$ are automatically interpreted as the empty set by the above definition. The definitions of $C^{\mathcal{I}}$ and $R^{\mathcal{I}}$ are well-defined due to ($*$) above.

We can now observe the following desired correspondence between $\mathcal{I}$ and $\mathcal{S}$: For any $C, D \in \mathcal{E}$, we find that $[C] \in D^{\mathcal{I}}$ iff $C \sqsubseteq D \in \mathcal{S}$ (†). We distinguish the following cases based on the structure of D:

- $D = \bot$. Clearly $[C] \notin \bot^{\mathcal{I}}$. To show $C \sqsubseteq \bot \notin \mathcal{S}$, note that $C' \sqsubseteq \bot \notin \mathcal{S}$ for all $C' \in \mathcal{E}$. Otherwise, the first axiom of the form $C' \sqsubseteq \bot$ could only have been introduced in (EL2), which contradicts our assumption that $\bot \notin \mathcal{E}$.
- $D = \top$. By (EL1) $C \sqsubseteq \top \in \mathcal{S}$, and of course also $[C] \in \top^{\mathcal{I}}$.
- $D \in \mathsf{N}_C$. This case follows directly from the definition of $\mathcal{I}$.
- $D = \{a\}$ for some $a \in \mathsf{N}_I$. If $[C] \in \{a\}^{\mathcal{I}}$ then $[C] = [\{a\}]$, and hence $C \sim \{a\}$. Since $\{a\} \sqsubseteq \{a\} \in \mathcal{S}$ (EL1), we obtain $C \sqsubseteq \{a\} \in \mathcal{S}$ from (∗).
  Conversely, if $C \sqsubseteq \{a\} \in \mathcal{S}$, then $C \sim \{a\}$ and hence $\{[C]\} = \{[\{a\}]\} = \{a\}^{\mathcal{I}}$ as required.

Finally, it only remains to show that $\mathcal{I}$ is indeed a model of RB. We argue that each rule $B \to H$ of RB is satisfied by $\mathcal{I}$. Thus consider some variable assignment Z such that $B^{\mathcal{I},Z} = \mathit{true}$. This means that for all $C(x) \in B$ ($R(x,y) \in B$), we find that $Z(x) \in C^{\mathcal{I}}$ ($\langle Z(x), Z(y)\rangle \in R^{\mathcal{I}}$). Now assume that $Z(x) = [D]$ ($Z(y) = [D']$). Now for concept atoms $C(x)$, we conclude $D \sqsubseteq C \in \mathcal{S}$ by (†). For role atoms $R(x,y)$, we obtain $D \sqsubseteq \exists R.D' \in \mathcal{S}$ as a direct consequence of the definition of $\mathcal{I}$. Since this reasoning applies to all atoms in B, there must be a match θ such that $Z(x) = [\theta(x)]$ for all variables x of B.

Now consider the rule head H. If H is of the form $C(x)$, then by (EL2) we find that $\theta(x) \sqsubseteq C \in \mathcal{S}$. If $C \in \mathcal{E}$ we can conclude $[\theta(x)] \in C^{\mathcal{I}}$ by (†), and since $Z(x) = [\theta(x)]$, we find that $\mathcal{I} \models B \to H$. Otherwise, if $C = \exists R.D$ (thus $\theta(x) \sqsubseteq \exists R.D \in \mathcal{S}$), we find that $D \in \mathcal{E}$, again by (EL2). Hence, according to the definition of $\mathcal{I}$, we have $\langle [\theta(x)], [D]\rangle \in R^{\mathcal{I}}$, and also $[D] \in D^{\mathcal{I}}$ where we use (†) again. This shows $[\theta(x)] \in C^{\mathcal{I}}$ as above, and hence $\mathcal{I} \models B \to H$ as required.

The case of rule heads of the form $R(x,y)$ is treated similarly, using (EL3). □

Combining the above results, we obtain the main result of this section:

Theorem 15. *Satisfiability checking, instance retrieval, and computing class subsumptions for $\mathcal{EL}^{++}$ rule bases is possible in polynomial time in the size of the rule base.*

6 DLP 2

Description Logic Programs (DLP) have been proposed as a tractable knowledge representation formalism for bridging the gap between DL and (Horn) logic programming [7]. This clearly suggests further extension with DL rules, and we will see below that reasoning with this extension is still possible in polynomial time. Moreover, various further features of $\mathcal{SROIQ}$ can easily be included as well, and thus we arrive at a DL rule language that might be dubbed DLP 2 in analogy to the ongoing standardisation of the extended OWL 2 based on $\mathcal{SROIQ}$.

DLP has been defined in various ways, and a detailed syntactic characterisation is found in [13]. Essentially, however, DLP can be characterised as the fragment of $\mathcal{SHOIQ}$ that can entail neither disjunctive information nor the existence of anonymous individuals. The former condition has been extensively studied in the context of Horn description logics [14], and rather complex syntactic definitions can be given to characterise all admissible axioms of such logics. Here, we adopt a much simpler definition that focusses on the essential expressive features without encompassing all alternative syntactic forms of DLP axioms:

Definition 16. *Roles of DLP are defined as in $\mathcal{SROIQ}$, including inverse roles. A* DLP body concept *is any $\mathcal{SROIQ}$ concept expression that includes only concept names, nominals, $\sqcap$, $\exists$, $\top$, and $\bot$. A* DLP head concept *is any $\mathcal{SROIQ}$ concept expression that includes only concept names, nominals, $\sqcap$, $\forall$, $\top$, $\bot$, and expressions of the form $\leq 1\, R.C$ where C is a DLP body concept.*

A DLP knowledge base *is a set of Rbox axioms of the form $R \sqsubseteq S$ and $R \circ R \sqsubseteq R$, Tbox axioms of the form $C \sqsubseteq D$, and Abox axioms of the form $D(a)$ and $R(a,b)$, where $C \in \mathbf{C}$ is a body concept, $D \in \mathbf{C}$ is a head concept, and $a, b \in \mathsf{N}_I$ are individual names. A* DLP rule base *is a set of DL rules such that all concepts in rule bodies are body concepts, and all concepts in rule heads are head concepts.*

A DLP 2 *knowledge base consists of a DLP knowledge base that additionally might contain Rbox axioms of the form $\mathsf{Dis}(R, S)$ and $\mathsf{Asy}(R)$, together with some DLP rule base.*

Note that neither regularity nor simplicity restrictions apply in DLP. It is immediate that DLP Rbox and Tbox axioms can directly be expressed by DLP rules. The same holds for Abox axioms: though we cannot use the common translation of $R(a,b)$ into $\{a\}(x) \to \exists R.\{b\}(x)$, the DLP rule $\{a\}(x) \wedge \{b\}(y) \to R(x,y)$ serves the same purpose. Hence we can restrict our further considerations to DLP 2 knowledge bases into which all knowledge base axioms other than $\mathsf{Dis}(R, S)$ and $\mathsf{Asy}(R)$ have been internalised. The core observation of this section is as follows:

Proposition 17. *Any DLP 2 knowledge base* KB *can be transformed into an equisatisfiable set of function-free first-order Horn rules with at most five variables per formula, and this transformation is possible in polynomial time w.r.t. the size of* KB.

Proof. We use RB to denote the DLP rule base of KB. The transformation proceeds in multiple stages, that we will present and verify independently.

First of all, we expand DL concept atoms as done in Proposition 10. Individual names in argument positions are not a problem now – they can just be kept throughout the translation. The transformation algorithm iteratively transforms RB until further iterations do no longer modify RB. In each iteration, the following steps are applied to each rule $B \to H$ in RB:

– if H is of the form $\forall R.C(x)$ such that C is no concept name, then the rule $B \to H$ is replaced by two new rules $B \to \forall R.A(x)$ and $A(x) \to C(x)$ where $A \in \mathsf{N}_C$ is new,
– if H is of the form $\leq 1\, R.C(x)$ such that C is no concept name, then the rule $B \to H$ is replaced by two new rules $B \to\ \leq 1\, R.A(x)$ and $C(x) \to A(x)$ where $A \in \mathsf{N}_C$ is new,
– if H is of the form $(C \sqcap D)(x)$, then the rule $B \to H$ is replaced by two new rules $B \to C(x)$ and $B \to D(x)$,

– if B contains an atom of the form $\exists R.C(x)$, it is replaced by two new atoms $R(x, y)$ and $C(y)$ where $y \in \mathbf{V}$ is new,
– if B contains an atom of the form $(C \sqcap D)(x)$, it is replaced by two new atoms $C(x)$ and $D(x)$.

Again it is easy to see that this transformation preserves satisfiability of RB in each transformation step. The number of applicable steps is bounded by the size of RB: expansion of rule heads may generate new rules for each conjunction operator occurring in rule heads, but their number is linearly bounded, and expansion of body atoms may only incur a linear increase in size for each rule body.

We thus arrive at an equisatisfiable rule base RB all of whose concept atoms are concept names and nominals, with the only exception of rule heads of the form $\forall R.A$ and $\leq 1\, R.A$ with $A \in \mathsf{N}_C$.

We proceed by reducing the structure of rule bodies. Given some rule body B and term t, we define $B_t := \{C(t) \mid C(t) \in B\}$ for some term t. In each iteration step of the reduction, select some rule $B \to H$ in RB that contains more than three variables, and do one of the following:

(1) If there is some $R(t, u) \in B$ such that u is final and u does not occur in H, then the rule $B \to H$ is replaced by two new rules $(B \setminus (B_u \cup \{R(t, u)\})) \cup \{C(t)\} \to H$ and $B_u \cup \{R(t, u)\} \to C(t)$, where $C \in \mathsf{N}_C$ is a new concept name.
(2) If there is some $C(t) \in B$ such that t occurs neither in H nor in any role atom of B, then the rule $B \to H$ is replaced by two new rules $(B \setminus B_t) \cup \{D(u)\} \to H$ and $B_t \to D(u)$, where $u \neq t$ is some arbitrary term in H, and $D \in \mathsf{N}_C$ is a new concept name.
(3) If $H = R(t, u)$ and there are role atoms $S(v, v'), S'(v', u) \in B$ but no further role atom of the form $S''(v', v'') \in B$, then $B \to H$ is replaced by two new rules $(B \setminus (B_{v'} \cup \{S(v, v'), S'(v', u)\})) \cup T(v, u) \to H$ and $B_{v'} \cup \{S(v, v'), S'(v', u)\} \to T(v, u)$, where $T \in \mathsf{N}_R$ is a new role name.

This iteration is repeated until no further changes occur. It is easy to see that the process terminates after polynomially many steps: every step removes atoms from an existing rule body, and none of the generated rules has more than three variables.

Claim 1 After the above translation, all rules in RB have at most three variables in the body.

For a contradiction, suppose that there is some rule $B \to H$ with at least four variables in B. By assumption, none of the three cases of the translation is applicable. Due to case (1), for any role atom $R(t, u) \in B$ where u is final, u must occur in H (since H contains at most one non-initial element by (R3)) and is unique $(*)$. Thus, by case (2), all variables in B must also occur in role atoms or in H.

Now assume H is a concept atom. Then H cannot contain any final u that occurs in a role atom $R(t, u) \in B$ $(*)$, and hence B contains no role atoms. But then B contains at most one variable, which would contradict our assumption. Thus assume that H is a role atom. Based on our conclusion $(*)$ that B contains at most one final term that is part of some role atom, we conclude that the role atoms of B must form a chain. But then, assuming that B contains at least four variables, there must be atoms $S(v, v'), S'(v', u) \in$

B as required by (3). Since (3) was assumed to not be applicable, there must be some atom $S''(v', v'')$ as in the condition of (3). Since $v'' \neq u$ cannot be final, and since u is the only final element in role atoms of B, there must be some path from v'' to u. But this contradicts (R2) and hence refutes the initial assumption on the number of variables in B.

Claim 2 The above translation preserves satisfiability of RB.

This can be shown by a simple induction, given that all possible transformation steps preserve satisfiability. Thus consider step (1), where $B \to H$ is the processed rule, and $B_1 \to H$ and $B_2 \to C(t)$ denote the generated rules. Clearly, adding $B_2 \to C(t)$ to RB preserves satisfiability since C is new. Thus it remains to show equisatisfiability of $\mathrm{RB}_1 := \mathrm{RB} \cup \{B_2 \to C(t)\}$ and $\mathrm{RB}_2 := \mathrm{RB} \cup \{B_2 \to C(t), B_1 \to H\} \setminus \{B \to H\}$.

Thus consider some interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathrm{RB}_1$. Then there is some interpretation $\mathcal{I}'$ with $\mathcal{I}' \models \mathrm{RB}_1$ and $C^{\mathcal{I}'} = \{\delta \in \Delta^{\mathcal{I}'} \mid B_2^{\mathcal{I}',Z} = \mathit{true}$ for some variable assignment Z with $t^{\mathcal{I}',Z} = \delta\}$. A suitable $\mathcal{I}'$ can be obtained from $\mathcal{I}$ by minimising the extent of C while preserving all other aspects of the interpretation, which can be done since C is new. Note that $\mathcal{I}' \models B_2 \to C(t)$ by definition. We claim that $\mathcal{I}' \models \mathrm{RB}_2$. Thus assume that $B_1^{\mathcal{I}',Z} = \mathit{true}$ for some variable assignment Z. Then $C(t)^{\mathcal{I}',Z} = \mathit{true}$ and thus $t^{\mathcal{I}',Z} \in C^{\mathcal{I}'}$. By the assumptions on $C^{\mathcal{I}'}$, we find that there is some variable assignment Z' such that $B_2^{\mathcal{I}',Z'} = \mathit{true}$ where $t^{\mathcal{I}',Z} = t^{\mathcal{I}',Z'}$. Now observe that, by construction, B_2 and B_1 contain no common variables, other than possibly t (if t is a variable). Thus there is some variable assignment Z'' such that $Z''(x) = Z(x)$ for any variable x in B_1 and $Z''(x) = Z'(x)$ for any variable x in B_2. But then $(B_1 \cup B_2)^{\mathcal{I}',Z''} = \mathit{true}$. As defined in (1), $(B_1 \cup B_2) = B$ and thus $B^{\mathcal{I}',Z''} = \mathit{true}$, and we can conclude $H^{\mathcal{I}',Z''} = \mathit{true}$ since $\mathcal{I}' \models B \to H$. By definition, Z and Z'' agree on all terms in H and thus we obtain $H^{\mathcal{I}',Z} = \mathit{true}$ as required. Since Z was arbitrary, this shows that $\mathcal{I} \models B_1 \to H$, and hence $\mathcal{I}' \models \mathrm{RB}_2$.

For the other direction, consider some interpretation $\mathcal{I}$ such that $\mathcal{I} \models \mathrm{RB}_2$. We claim that $\mathcal{I} \models \mathrm{RB}_1$. Thus assume that $B^{\mathcal{I},Z} = \mathit{true}$ for some variable assignment Z. Then also $B_2^{\mathcal{I},Z} = \mathit{true}$ as $B_2 \subseteq B$, and hence $C(t)^{\mathcal{I},Z} = \mathit{true}$. But then $B_1^{\mathcal{I},Z} = \mathit{true}$ and thus $H^{\mathcal{I},Z} = \mathit{true}$ as required.

The cases (2) and (3) can be treated in a similar fashion, where again it is essential that each case completely eliminates some term from the transformed rule, so that the required merging of variable assignments Z' and Z'' is indeed possible.

Finally, we can rewrite the transformed rules and axioms into equivalent first-order formulae. After the above transformations, RB contains only rules with at most three variables in the body, and with heads of one of the following forms: $R(t, u)$, $A(t)$, $\{a\}(t)$, $\forall R.A(t)$ and $\leq 1\, R.A(t)$ (with $A \in \mathsf{N}_C$). Concept atoms in rule bodies contain only concept names and nominals. Now consider a new binary predicate $\approx$ and let P be the logic program consisting of the following rules (as before, we omit universal quantifiers from first-order rules):

$$\begin{array}{lcrlcr}
 & \to & x \approx x & C(x) \wedge x \approx y & \to & C(y) \\
x \approx y & \to & y \approx x & R(x,z) \wedge x \approx y & \to & R(y,z) \\
x \approx y \wedge y \approx z & \to & x \approx z & R(z,x) \wedge x \approx y & \to & R(z,y) \\
R(x,y) & \to & R^-(y,x) & R^-(x,y) & \to & R(y,x)
\end{array}$$

instantiated for every concept name C and role name R occuring in KB. Clearly P is still polynomial in size. We now extend P with translated rules from RB. Thus, for any rule $B \to H \in$ RB, do the following:

- replace every concept atom of the form $\{a\}(t)$ in $B \to H$ with $a \approx t$,
- if $H = \forall R.A(t)$, replace H by $A(x)$ and add $R(t, x)$ to B, where x is a new variable,
- if $H = \leq 1\, R.A(t)$, replace H by $x \approx y$ and add $\{R(t, x), R(t, y), A(x), A(y)\}$ to B, where x and y are new variables,
- add $B \to H$ to P.

It is easy to see that the above translations preserve the semantics of each rule, and that each resulting rule contains at most five variables. Finally, we translate all Rbox axioms of KB to rules as follows:

- $\mathsf{Dis}(R, S)$ is translated to $R(x, y) \wedge S(x, y) \to$.
- $\mathsf{Asy}(R)$ is translated to $R(x, y) \wedge R(y, x) \to$.

This finishes the proof. □

This establishes the tractability of DLP 2:

Theorem 18. *Satisfiability checking, instance retrieval, and computing class subsumptions for DLP 2 knowledge bases is possible in polynomial time in the size of the knowledge base.*

Proof. First note that instance retrieval and class subsumption can be reduced to satisfiability checking just as in the case of $\mathcal{EL}^{++}$. Now to check satisfiability of a DLP 2 knowledge base, it is first transformed into an equisatisfiable set of function-free first-order Horn rules as in Proposition 17. The satisfiability of such a set of formulae can be checked in polynomial time, since any Horn logic program is semantically equivalent to its *grounding* (the set of all possible ground instances of the given rules based on the occuring individual names). For a program with a bounded number n of variables per rule, this grounding is bounded by $r \times i^n$, where i is the number of individual names and r is the number of rules in the program. Finally, the evaluation of ground Horn logic programs is known to be P-complete. □

7 CONCLUSION

We have introduced *DL rules* as a rule-based formalism for augmenting description logic knowledge bases. For all DLs considered in this paper – $\mathcal{SROIQ}$, $\mathcal{EL}^{++}$, and DLP – the extension with DL rules does not increase the worst-case complexity. In particular, $\mathcal{EL}^{++}$ rules and the extended DLP 2 allow for polynomial time reasoning for common inference tasks, even though DL rules do indeed provide added expressive features in those cases.

The main contributions of this paper therefore are twofold. Firstly, we have extended the expressivity of two tractable DLs while preserving their favourable computational properties. The resulting formalisms of $\mathcal{EL}^{++}$ rules and DLP 2 are arguably close to being maximal tractable fragments of $\mathcal{SROIQ}$. In particular, note that the union of $\mathcal{EL}^{++}$ and DLP is no longer tractable, even when disallowing number restrictions and inverse roles: this follows from the fact that this DL contains the DL Horn-$\mathcal{FLE}$ which was shown to be ExpTime-complete in [14].

Secondly, while DL rules do not truly add expressive power to $\mathcal{SROIQ}$, our characterisation and reduction methods for DL rules provides a basis for developing ontology modelling tools. Indeed, even without any further extension, the upcoming OWL 2 standard would support all DL rules. Hence OWL-conformant tools can choose to provide rule-based user interfaces (as done for Protégé in [8]), and rule-based tools may offer some amount of OWL support. We remark that in the case of DLP and $\mathcal{EL}^{++}$, the conditions imposed on DL rules can be checked individually for each rule without considering the knowledge base as a whole. Moreover, in order to simplify rule editing, the general syntax of DL rules can be further restricted without sacrificing expressivity, e.g. by considering only chains rather than trees for rule bodies. We thus argue that DL rules can be a useful interface paradigm for many application fields.

Our treatment of rules in $\mathcal{EL}^{++}$ and DLP 2 – used only for establishing complexity bounds in this paper – can be the basis for novel rule-based reasoning algorithms for those DLs, and we leave it for future research to explore this approach.

Acknowledgements Research reported in this paper was supported by the EU in the IST projects ACTIVE (IST-2007-215040) and NeOn (IST-2006-027595) as well as by the German Research Foundation under the ReaSem project.

References

1. Horrocks, I., Kutz, O., Sattler, U.: The even more irresistible $\mathcal{SROIQ}$. In: Proc. 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR2006), AAAI Press (2006) 57–67
2. Baader, F., Brandt, S., Lutz, C.: Pushing the EL envelope. In: Proc. 19th Int. Joint Conf. on Artificial Intelligence (IJCAI 2005), Edinburgh, UK, Morgan-Kaufmann Publishers (2005)
3. Dantsin, E., Eiter, T., Gottlob, G., Voronkov, A.: Complexity and expressive power of logic programming. ACM Computing Surveys **33** (2001) 374–425
4. Horrocks, I., Patel-Schneider, P.F.: A proposal for an OWL rules language. In Feldman, S.I., Uretsky, M., Najork, M., Wills, C.E., eds.: Proc. 13th Int. Conf. on World Wide Web (WWW 2004), ACM (2004) 723–731
5. Motik, B., Sattler, U., Studer, R.: Query answering for OWL-DL with rules. J. Web Sem. **3**(1) (2005) 41–60
6. Levy, A.Y., Rousset, M.C.: Combining Horn rules and description logics in CARIN. Artificial Intelligence **104** (1998) 165–209
7. Grosof, B.N., Horrocks, I., Volz, R., Decker, S.: Description logic programs: combining logic programs with description logic. In: Proc. 12th Int. Conf. on World Wide Web (WWW 2003), ACM (2003) 48–57
8. Gasse, F., Sattler, U., Haarslev, V.: Rewriting rules into $\mathcal{SROIQ}$ axioms. Poster at 21st Int. Workshop on DLs (DL-08) (2008)
9. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P., eds.: The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press (2007)

10. Rudolph, S., Krötzsch, M., Hitzler, P.: All elephants are bigger than all mice. In: Proc. 21st Int. Workshop on Description Logics (DL-08). (2008)
11. Calvanese, D., Giacomo, G.D., Lenzerini, M.: On the decidability of query containment under constraints. In: Proc. 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'98), ACM Press (1998) 149–158
12. Krötzsch, M., Rudolph, S., Hitzler, P.: ELP: Tractable rules for OWL 2. Technical report, Universität Karlsruhe, Germany (2008) `http://korrekt.org/page/ELP`.
13. Volz, R.: Web Ontology Reasoning with Logic Databases. PhD thesis, Universität Karlsruhe (TH), Germany (2004)
14. Krötzsch, M., Rudolph, S., Hitzler, P.: Complexity boundaries for Horn description logics. In: Proc. 22nd AAAI Conf. (AAAI'07). (2007)