TECHNISCHE UNIVERSITÄT DRESDEN

Technische Universität Dresden
Institute for Theoretical Computer Science
Chair for Automata Theory

LTCS–Report

A Goal-Oriented Algorithm for Unification in $\mathcal{ELH}_{R^+}$ w.r.t. Cycle-Restricted Ontologies

Franz Baader Stefan Borgwardt Barbara Morawska

LTCS-Report 12-05

Postal Address:
Lehrstuhl für Automatentheorie
Institut für Theoretische Informatik
TU Dresden
01062 Dresden

http://lat.inf.tu-dresden.de

Visiting Address:
Nöthnitzer Str. 46
Dresden

Contents

Abstract

Unification in Description Logics (DLs) has been proposed as an inference service that can, for example, be used to detect redundancies in ontologies. For the DL $\mathcal{EL}$, which is used to define several large biomedical ontologies, unification is NP-complete. A goal-oriented NP unification algorithm for $\mathcal{EL}$ that uses nondeterministic rules to transform a given unification problem into solved form has recently been presented. In this report, we extend this goal-oriented algorithm in two directions: on the one hand, we add general concept inclusion axioms (GCIs), and on the other hand, we add role hierarchies ($\mathcal{H}$) and transitive roles (R^+). For the algorithm to be complete, however, the ontology consisting of the GCIs and role axioms needs to satisfy a certain cycle restriction.

1 Introduction

The DL $\mathcal{EL}$, which offers the constructors conjunction ($\sqcap$), existential restriction ($\exists r.C$), and the top concept ($\top$), has recently drawn considerable attention since, on the one hand, important inference problems such as the subsumption problem are polynomial in $\mathcal{EL}$, even in the presence of general concept inclusions (GCIs) [11]. On the other hand, though quite inexpressive, $\mathcal{EL}$ can be used to define biomedical ontologies, such as the large medical ontology SNOMED CT.[1] A tractable extension of $\mathcal{EL}$ [6], which includes role hierarchy and transitivity axioms, is the basis of the OWL 2 EL profile of the new Web Ontology Language OWL 2.[2]

Unification in DLs has been proposed in [10] as a novel inference service that can, for instance, be used to detect redundancies in ontologies. For example, assume that one developer of a medical ontology defines the concept of a *patient with severe injury of the frontal lobe* as

$$\exists \mathsf{finding}.(\mathsf{Frontal\_lobe\_injury} \sqcap \exists \mathsf{severity}.\mathsf{Severe}), \tag{1}$$

whereas another one represents it as

$$\exists \mathsf{finding}.(\mathsf{Severe\_injury} \sqcap \exists \mathsf{finding\_site}.\exists \mathsf{part\_of}.\mathsf{Frontal\_lobe}). \tag{2}$$

These two concept descriptions are not equivalent, but they are nevertheless meant to represent the same concept. They can obviously be made equivalent by treating the concept names Frontal_lobe_injury and Severe_injury as variables, and substituting the first one by $\mathsf{Injury} \sqcap \exists \mathsf{finding\_site}.\exists \mathsf{part\_of}.\mathsf{Frontal\_lobe}$ and the second one by $\mathsf{Injury} \sqcap \exists \mathsf{severity}.\mathsf{Severe}$. In this case, we say that the descriptions are unifiable, and call the substitution that makes them equivalent a *unifier*.

[1] see http://www.ihtsdo.org/snomed-ct/

[2] See http://www.w3.org/TR/owl2-profiles/

Our interest in unification w.r.t. GCIs, role hierarchies, and transitive roles stems from the fact that these features are important for expressing medical knowledge. For example, assume that the developers use the descriptions (3) and (4) instead of (1) and (2):

$$\exists \mathsf{finding}.\exists \mathsf{finding\_site}.\exists \mathsf{part\_of}.\mathsf{Brain} \;\sqcap\; \exists \mathsf{finding}.(\mathsf{Frontal\_lobe\_injury} \sqcap \exists \mathsf{severity}.\mathsf{Severe}) \tag{3}$$

$$\exists \mathsf{status}.\mathsf{Emergency} \;\sqcap\; \exists \mathsf{finding}.(\mathsf{Severe\_injury} \sqcap \exists \mathsf{finding\_site}.\exists \mathsf{part\_of}.\mathsf{Frontal\_lobe}) \tag{4}$$

The descriptions (3) and (4) are not unifiable without additional background knowledge, but they are unifiable, with the same unifier as above, if the GCIs

$$\begin{aligned} \exists \mathsf{finding}.\exists \mathsf{severity}.\mathsf{Severe} &\sqsubseteq \exists \mathsf{status}.\mathsf{Emergency}, \\ \mathsf{Frontal\_lobe} &\sqsubseteq \exists \mathsf{proper\_part\_of}.\mathsf{Brain} \end{aligned}$$

are present in a background ontology and this ontology additionally states that $\mathsf{part\_of}$ is transitive and $\mathsf{proper\_part\_of}$ is a subrole of $\mathsf{part\_of}$.

In [7], we were able to show that unification in the DL $\mathcal{EL}$ (without GCIs and role axioms) is NP-complete. In addition to a brute-force "guess and then test" NP-algorithm [7], we have developed a goal-oriented unification algorithm for $\mathcal{EL}$, in which nondeterministic decisions are only made if they are triggered by "unsolved parts" of the unification problem [9], and an algorithm that is based on a reduction to satisfiability in propositional logic (SAT) [8], which enables the use of highly-optimized SAT solvers [13]. Whereas both approaches are clearly better than the brute-force algorithm, none of them is uniformly better than the other. First experiments with our system UEL [1] show that the SAT translation is usually faster in deciding unifiability, but it needs more space than the goal-oriented algorithm and it produces more uninteresting and large unifiers. In fact, the SAT translation generates all so-called local unifiers, whereas the goal-oriented algorithm produces all so-called minimal unifiers, though it may also produce some non-minimal ones. The set of minimal unifiers is a subset of the set of local unifiers, and in our experiments the minimal unifiers usually made more sense in the application.

In [9] it was shown that the approaches for unification of $\mathcal{EL}$-concept descriptions (without any background ontology) mentioned above can easily be extended to the case of a so-called acyclic TBox (a simple form of GCIs, which basically introduce abbreviations for concept descriptions) as background ontology without really changing the algorithms or increasing their complexity. For more general GCIs, such a simple solution is no longer possible. In [3], we extended the brute-force "guess and then test" NP-algorithm from [7] to the case of GCIs, which required the development of a new characterization of subsumption w.r.t. GCIs in $\mathcal{EL}$. Unfortunately, the algorithm is complete only for general TBoxes (i.e.,

finite sets of GCIs) that satisfy a certain restriction on cycles, which, however, does not prevent all cycles. For example, the cyclic GCI $\exists \mathsf{child}.\mathsf{Human} \sqsubseteq \mathsf{Human}$ satisfies this restriction, whereas the cyclic GCI $\mathsf{Human} \sqsubseteq \exists \mathsf{parent}.\mathsf{Human}$ does not. In [4] we provide a more practical unification algorithm that is based on a translation into SAT, and can also deal with role hierarchies and transitive roles, but still needs the ontology (now consisting of GCIs and role axioms) to be cycle-restricted. In the presence of role hierarchies ($\mathcal{H}$) and transitive roles (R^+), we use the name $\mathcal{EL}\mathcal{H}_{R^+}$ rather than $\mathcal{EL}$ for the logic.

Motivated by our experience that, for the case of $\mathcal{EL}$ without background ontology, the goal-oriented algorithm sometimes behaves better than the one based on a translation into SAT, we introduce a goal-oriented algorithm for unification in $\mathcal{EL}\mathcal{H}_{R^+}$ w.r.t. cycle-restricted ontologies. In a previous report [2], we have described a specialized version of the algorithm that only deals with cycle-restricted $\mathcal{EL}$-ontologies.

2 The Description Logics $\mathcal{EL}$ and $\mathcal{EL}\mathcal{H}_{R^+}$

The expressiveness of a DL is determined both by the formalism for describing concepts (the concept description language) and the terminological formalism, which allows to state additional constraints on the interpretation of concepts and roles in a so-called ontology.

The concept description language considered in this report is called $\mathcal{EL}$. Starting with a finite set N_C of *concept names* and a finite set N_R of *role names*, $\mathcal{EL}$-*concept descriptions* are built from concept names by the constructors *conjunction* ($C \sqcap D$), *existential restriction* ($\exists r.C$ for every $r \in N_R$), and *top* ($\top$). Since we only consider $\mathcal{EL}$-concept descriptions, we will sometimes dispense with the prefix $\mathcal{EL}$.

On the semantic side, concept descriptions are interpreted as sets. To be more precise, an *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a non-empty domain $\Delta^{\mathcal{I}}$ and an interpretation function that maps concept names to subsets of $\Delta^{\mathcal{I}}$ and role names to binary relations over $\Delta^{\mathcal{I}}$. This function is inductively extended to concept descriptions as shown in the semantics column of Table 1.

A *general concept inclusion axiom (GCI)* is of the form $C \sqsubseteq D$ for concept descriptions C, D, a *role hierarchy axiom* is of the form $r \sqsubseteq s$ for role names r, s, and a *transitivity axiom* is of the form $r \circ r \sqsubseteq r$ for a role name r. An interpretation $\mathcal{I}$ *satisfies* such an axiom if the corresponding condition in the semantics column of Table 1 holds, where $\circ$ in this column stands for composition of binary relations. An $\mathcal{EL}\mathcal{H}_{R^+}$*-ontology* is a finite set of such axioms. It is an $\mathcal{EL}$*-ontology* if it contains only GCIs. An interpretation is a *model* of an ontology if it satisfies all its axioms.

Name	Syntax	Semantics
concept name	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
role name	r	$r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
top	$\top$	$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
conjunction	$C \sqcap D$	$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
existential restr.	$\exists r.C$	$(\exists r.C)^{\mathcal{I}} = \{x \mid \exists y : (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
concept def.	$A \equiv C$	$A^{\mathcal{I}} = C^{\mathcal{I}}$
GCI	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
role hierarchy	$r \sqsubseteq s$	$r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$
transitivity	$r \circ r \sqsubseteq r$	$r^{\mathcal{I}} \circ r^{\mathcal{I}} \subseteq r^{\mathcal{I}}$

Table 1: Syntax and semantics of $\mathcal{EL}$.

A concept description C is *subsumed* by a concept description D w.r.t. an ontology $\mathcal{O}$ (written $C \sqsubseteq_{\mathcal{O}} D$) if every model of $\mathcal{O}$ satisfies the GCI $C \sqsubseteq D$. We say that C is *equivalent* to D w.r.t. $\mathcal{O}$ ($C \equiv_{\mathcal{O}} D$) if $C \sqsubseteq_{\mathcal{O}} D$ and $D \sqsubseteq_{\mathcal{O}} C$. If $\mathcal{O}$ is empty, we also write $C \sqsubseteq D$ and $C \equiv D$ instead of $C \sqsubseteq_{\mathcal{O}} D$ and $C \equiv_{\mathcal{O}} D$, respectively. As shown in [11, 6], subsumption w.r.t. $\mathcal{ELH}_{R^+}$-ontologies (and thus also w.r.t. $\mathcal{EL}$-ontologies) is decidable in polynomial time.

Since conjunction is interpreted as intersection, the concept descriptions $(C \sqcap D) \sqcap E$ and $C \sqcap (D \sqcap E)$ are always equivalent. Thus, we dispense with parentheses and write nested conjunctions in flat form $C_1 \sqcap \cdots \sqcap C_n$. Nested existential restrictions $\exists r_1.\exists r_2. \ldots \exists r_n.C$ will sometimes also be written as $\exists r_1 r_2 \ldots r_n.C$, where $r_1 r_2 \ldots r_n$ is viewed as a word over the alphabet of role names, i.e., an element of N_R^*.

The *role hierarchy* induced by an ontology $\mathcal{O}$ is a binary relation $\trianglelefteq_{\mathcal{O}}$ on N_R, which is defined as the reflexive-transitive closure of the relation $\{(r,s) \mid r \sqsubseteq s \in \mathcal{O}\}$. Using elementary reachability algorithms, the role hierarchy can be computed in polynomial time in the size of $\mathcal{O}$. It is easy to see that $r \trianglelefteq_{\mathcal{O}} s$ implies that $r^{\mathcal{I}} \subseteq s^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{O}$.

An $\mathcal{EL}$-concept description is an *atom* if it is an existential restriction or a concept name. The atoms of an $\mathcal{EL}$-concept description C are the subdescriptions of C that are atoms, and the top-level atoms of C are the atoms occurring in the top-level conjunction of C. Obviously, any $\mathcal{EL}$-concept description is the conjunction of its top-level atoms, where the empty conjunction corresponds to $\top$. The atoms of an $\mathcal{ELH}_{R^+}$-ontology $\mathcal{O}$ are the atoms of all the concept descriptions occurring in GCIs of $\mathcal{O}$.

An atom is called *flat* if it is a concept name or an existential restriction of the form $\exists r.A$ for a concept name A. A GCI is called *flat* if it is of the form $A \sqcap B \sqsubseteq C$, where A, B are flat atoms or $\top$ and C is a flat atom. An $\mathcal{ELH}_{R^+}$-ontology is called *flat* if all its GCIs are flat. Every $\mathcal{ELH}_{R^+}$-ontology can be transformed in

polynomial time into an equivalent flat ontology (see [5] for details).

3 Subsumption in $\mathcal{ELH}_{R^+}$

All previous unification algorithms for $\mathcal{EL}$ depend on a structural characterization of subsumption. While [7] provides one for the case of an empty background ontology, in [3] a more general characterization is used that accounts for GCIs. This is proved using a Gentzen-style proof calculus for subsumption. We now extend this calculus to prove a similar characterization for arbitrary $\mathcal{ELH}_{R^+}$-ontologies.

3.1 Proving Subsumptions by Inference Rules

In [2] we defined a system of inference rules that characterize subsumption in $\mathcal{EL}$ modulo a flat $\mathcal{EL}$-ontology $\mathcal{O}$. More precisely, they define a binary relation $\vdash_{\mathcal{O}}$ on all concept descriptions such that $C \vdash_{\mathcal{O}} D$ iff $C \sqsubseteq_{\mathcal{O}} D$. For details about this relation, we refer to [2]. We will describe here only the changes that are necessary to generalize this approach to $\mathcal{ELH}_{R^+}$-ontologies.

We only need to modify the inference rule that deals with existential restrictions:

($\mathbf{R}_8$) *Closure under existential restriction:* For all $\mathcal{EL}$-concept descriptions C, D and each $r \in N_R$,

$$\frac{C \vdash_{\mathcal{O}} D}{\exists r.C \vdash_{\mathcal{O}} \exists r.D}$$

We replace it by the following two rules:

($\mathbf{R}_{8'}$) *Role hierarchy:* For all $\mathcal{EL}$-concept descriptions C, D and all role names r, s with $r \sqsubseteq s \in \mathcal{O}$,

$$\frac{C \vdash_{\mathcal{O}} D}{\exists r.C \vdash_{\mathcal{O}} \exists s.D}$$

($\mathbf{R}_{8''}$) *Role transitivity:* For all $\mathcal{EL}$-concept descriptions C, D and each transitive role r,

$$\frac{C \vdash_{\mathcal{O}} \exists r.D}{\exists r.C \vdash_{\mathcal{O}} \exists r.D}$$

Notice that if $r = s$, then $(\mathbf{R}_{8'})$ is exactly $(\mathbf{R}_8)$. The following lemma is an extension of Lemma 9 from [2].

Lemma 1. *Let $\mathcal{O}$ be a flat $\mathcal{EL}\mathcal{H}_{R^+}$-ontology and C, D be two $\mathcal{EL}$-concept descriptions. Then $C \vdash_\mathcal{O} D$ iff $C \sqsubseteq_\mathcal{O} D$.*

Proof. It is easy to verify that the two new inference rules are sound, i.e., we have $C \sqsubseteq_\mathcal{O} D$ whenever we can derive $C \vdash_\mathcal{O} D$ using these rules.

If $C \vdash_\mathcal{O} D$ does not hold, we can show that $C^\mathcal{I} \not\subseteq D^\mathcal{I}$ holds in the following *canonical model* $\mathcal{I}$ of $\mathcal{T}$. The domain of $\mathcal{I}$ is the set $\mathfrak{C}$ of all $\mathcal{EL}$-concept descriptions built over N_C and N_R. For every concept name A, we define $A^\mathcal{I} := \{E \in \mathfrak{C} \mid E \vdash_\mathcal{O} A\}$ and for every role name r, we set $r^\mathcal{I} := \{(E, F) \in \mathfrak{C}^2 \mid E \vdash_\mathcal{O} \exists r.F\}$. With exactly the same arguments as in [2], we can show that $C'^\mathcal{I} = \{E \in \mathfrak{C} \mid E \vdash_\mathcal{O} C'\}$ holds for each concept description C'. There it was also shown that $\mathcal{I}$ is a model of all GCIs in $\mathcal{O}$. It remains to verify that it also satisfies the role axioms.

Let $r \sqsubseteq s \in \mathcal{O}$ and $(E, F) \in r^\mathcal{I}$, i.e., there is a proof tree $\mathbf{T}$ for $E \vdash_\mathcal{O} \exists r.F$. Then the following is a proof tree for $E \vdash_\mathcal{O} \exists s.F$:

$$\dfrac{\dfrac{}{E \vdash_\mathcal{O} \exists r.F}\,(\mathbf{T}) \qquad \dfrac{\dfrac{}{F \vdash_\mathcal{O} F}\,(\mathbf{R}_3)}{\exists r.F \vdash_\mathcal{O} \exists s.F}\,(\mathbf{R}_{8'})}{E \vdash_\mathcal{O} \exists s.F}\,(\mathbf{R}_9)$$

This shows that $r^\mathcal{I} \subseteq s^\mathcal{I}$, i.e., $\mathcal{I}$ satisfies this role hierarchy axiom.

For each transitive role r, we have to show that $r^\mathcal{I} \circ r^\mathcal{I} \subseteq r^\mathcal{I}$ holds. Let $(E_1, E_2) \in r^\mathcal{I}$ and $(E_2, E_3) \in r^\mathcal{I}$, i.e., there are proof trees $\mathbf{T}_1$ for $E_1 \vdash_\mathcal{O} \exists r.E_2$ and $\mathbf{T}_2$ for $E_2 \vdash_\mathcal{O} \exists r.E_3$. Then the following is a proof tree for $E_1 \vdash_\mathcal{O} \exists r.E_3$:

$$\dfrac{\dfrac{}{E_1 \vdash_\mathcal{O} \exists r.E_2}\,(\mathbf{T}_1) \qquad \dfrac{\dfrac{}{E_2 \vdash_\mathcal{O} \exists r.E_3}\,(\mathbf{T}_2)}{\exists r.E_2 \vdash_\mathcal{O} \exists r.E_3}\,(\mathbf{R}_{8''})}{E_1 \vdash_\mathcal{O} \exists r.E_3}\,(\mathbf{R}_9)$$

To conclude the proof, we notice that $C \in C^\mathcal{I}$, since $C \vdash_\mathcal{O} C$ holds by rule $(\mathbf{R}_3)$. On the other hand, we assumed that $C \vdash_\mathcal{O} D$ does not hold, which implies $C \notin D^\mathcal{I}$, and thus $C^\mathcal{I} \not\subseteq D^\mathcal{I}$. □

3.2 A Structural Characterization of Subsumption

Based on this proof-theoretic characterization of subsumption, we will now generalize a structural characterization of subsumption from [3] to $\mathcal{EL}\mathcal{H}_{R^+}$-ontologies.

We say that a subsumption between two atoms is *structural* if their top-level structure is compatible. To be more precise, following [4] we define structural

subsumption between atoms as follows: the atom C is *structurally subsumed* by the atom D w.r.t. $\mathcal{O}$ ($C \sqsubseteq^{\mathsf{s}}_{\mathcal{O}} D$) iff one of the following holds:

1. $C = D$ is a concept name,
2. $C = \exists r.C'$, $D = \exists s.D'$, $r \trianglelefteq_{\mathcal{O}} s$, and $C' \sqsubseteq_{\mathcal{O}} D'$.
3. $C = \exists r.C'$, $D = \exists s.D'$, and $C' \sqsubseteq_{\mathcal{O}} \exists t.D'$ for a transitive role t such that $r \trianglelefteq_{\mathcal{O}} t \trianglelefteq_{\mathcal{O}} s$.

It is easy to see that subsumption w.r.t. $\emptyset$ between two atoms implies structural subsumption w.r.t. $\mathcal{O}$, which in turn implies subsumption w.r.t. $\mathcal{O}$. Other important properties of $\sqsubseteq^{\mathsf{s}}_{\mathcal{O}}$ are reflexivity and transitivity (see [5]). The following lemma extends Lemma 6 from [2] and is an important foundation for the algorithm we will present later.

Lemma 2. *Let $\mathcal{O}$ be an $\mathcal{ELH}_{R^+}$-ontology and $C_1, \dots, C_n$, $D_1, \dots, D_m$ be atoms. Then $C_1 \sqcap \dots \sqcap C_n \sqsubseteq_{\mathcal{O}} D_1 \sqcap \dots \sqcap D_m$ iff for every $j \in \{1, \dots, m\}$*

1. *there is an index $i \in \{1, \dots, n\}$ such that $C_i \sqsubseteq^{\mathsf{s}}_{\mathcal{O}} D_j$ or*
2. *there are atoms $A_1, \dots, A_k, B$ of $\mathcal{T}$ ($k \geq 0$) such that*
   (a) *$A_1 \sqcap \dots \sqcap A_k \sqsubseteq_{\mathcal{O}} B$,*
   (b) *for every $\eta \in \{1, \dots, k\}$ there is $i \in \{1, \dots, n\}$ with $C_i \sqsubseteq^{\mathsf{s}}_{\mathcal{O}} A_\eta$, and*
   (c) *$B \sqsubseteq^{\mathsf{s}}_{\mathcal{O}} D_j$.*

Proof. If one of the alternatives 1. or 2. holds for every atom D_j, then clearly the claimed subsumption relationship holds. In [2], the other direction was first reduced to the case of a *flat* $\mathcal{EL}$-ontology. This reduction also works in the same way for $\mathcal{ELH}_{R^+}$-ontologies, which is why we can reuse most of the proof of Lemma 6 from [2].

Assume that $C_1 \sqcap \dots \sqcap C_n \sqsubseteq_{\mathcal{O}} D_1 \sqcap \dots \sqcap D_m$. By Lemma 1, there must be a proof tree $\mathbf{T}$ for $C_1 \sqcap \dots \sqcap C_n \vdash_{\mathcal{O}} D_1 \sqcap \dots \sqcap D_m$. We prove by induction on the height of $\mathbf{T}$ that for every atom D_j one of the alternatives 1. or 2. holds. We consider the rule applied at the root of $\mathbf{T}$. If this rule is one of the original rules, we can show the claim using the same arguments as in [2] since they only depend on reflexivity and transitivity of $\sqsubseteq^{\mathsf{s}}_{\mathcal{O}}$ and the GCIs in $\mathcal{O}$, but not on any other properties derived from $\mathcal{O}$. Hence we consider only proof trees that end with the application of one of our new inference rules ($\mathbf{R}_{8'}$) or ($\mathbf{R}_{8''}$).

If ($\mathbf{R}_{8'}$) has been applied, then $n = m = 1$, $C_1 = \exists r.C'$, $D_1 = \exists s.D'$, $r \sqsubseteq s \in \mathcal{O}$, and there is a proof tree for $C' \vdash_{\mathcal{O}} D'$. By Lemma 1, we have $C' \sqsubseteq_{\mathcal{O}} D'$. Since $r \trianglelefteq_{\mathcal{O}} s$, we have $C_1 \sqsubseteq^{\mathsf{s}}_{\mathcal{O}} D_1$ by definition of $\sqsubseteq^{\mathsf{s}}_{\mathcal{O}}$, i.e., the first alternative holds for D_1.

If ($\mathbf{R}_{8''}$) has been applied, then $n = m = 1$, $C_1 = \exists r.C'$, $D_1 = \exists r.D'$, $r \circ r \sqsubseteq r \in \mathcal{O}$, and there is a proof tree for $C' \vdash_{\mathcal{O}} \exists r.D'$. Again, Lemma 1 shows that $C' \sqsubseteq_{\mathcal{O}} \exists r.D'$, and thus we have $C_1 \sqsubseteq^{\mathsf{s}}_{\mathcal{O}} D_1$ since $r \trianglelefteq_{\mathcal{O}} r \trianglelefteq_{\mathcal{O}} r$, i.e., the first alternative holds for D_1. □

4 Unification in $\mathcal{ELH}_{R^+}$

We partition the set N_C into a set N_v of concept variables (which may be replaced by substitutions) and a set N_c of concept constants (which must not be replaced by substitutions). A *substitution* σ maps every concept variable to an $\mathcal{EL}$-concept description. It is extended to concept descriptions in the usual way:

- $\sigma(A) := A$ for all $A \in N_c \cup \{\top\}$,
- $\sigma(C \sqcap D) := \sigma(C) \sqcap \sigma(D)$ and $\sigma(\exists r.C) := \exists r.\sigma(C)$.

An $\mathcal{EL}$-concept description C is *ground* if it does not contain variables. Obviously, a ground concept description is not modified by applying a substitution. An $\mathcal{ELH}_{R^+}$-ontology is *ground* if it does not contain variables.

Definition 3. Let $\mathcal{O}$ be a ground $\mathcal{ELH}_{R^+}$-ontology. An *$\mathcal{ELH}_{R^+}$-unification problem* w.r.t. $\mathcal{O}$ is a finite set $\Gamma = \{C_1 \sqsubseteq^? D_1, \ldots, C_n \sqsubseteq^? D_n\}$ of subsumptions between $\mathcal{EL}$-concept descriptions. A substitution σ is a *unifier* of Γ w.r.t. $\mathcal{O}$ if σ *solves* all the subsumptions in Γ, i.e., if $\sigma(C_1) \sqsubseteq_{\mathcal{O}} \sigma(D_1), \ldots, \sigma(C_n) \sqsubseteq_{\mathcal{O}} \sigma(D_n)$. We say that Γ is *unifiable* w.r.t. $\mathcal{O}$ if it has a unifier.

Note that some of the previous papers on unification in DLs use equivalences $C \equiv^? D$ instead of subsumptions $C \sqsubseteq^? D$. This difference is, however, irrelevant since $C \equiv^? D$ can be seen as a shorthand for the two subsumptions $C \sqsubseteq^? D$ and $D \sqsubseteq^? C$, and $C \sqsubseteq^? D$ has the same unifiers as $C \sqcap D \equiv^? C$. Also note that we have restricted the background ontology $\mathcal{O}$ to be ground. This is not without loss of generality. If $\mathcal{O}$ contained variables, then we would need to apply the substitution also to its GCIs, and instead of requiring $\sigma(C_i) \sqsubseteq_{\mathcal{O}} \sigma(D_i)$ we would thus need to require $\sigma(C_i) \sqsubseteq_{\sigma(\mathcal{O})} \sigma(D_i)$, which would change the nature of the problem considerably (see [5] for a more detailed discussion).

As mentioned in the introduction, the unification algorithm we will present in Section 5 is complete only for $\mathcal{ELH}_{R^+}$-ontologies that satisfy a certain restriction on cycles.

Definition 4. The $\mathcal{ELH}_{R^+}$-ontology $\mathcal{O}$ is called *cycle-restricted* iff there is no nonempty word $w \in N_R^+$ and $\mathcal{EL}$-concept description C such that $C \sqsubseteq_{\mathcal{O}} \exists w.C$.

In [5] we show that a given $\mathcal{ELH}_{R^+}$-ontology can be tested for cycle-restrictedness in polynomial time. The main idea is that it is sufficient to consider the cases where C is a concept name or $\top$.

To simplify the description of the algorithm, it is convenient to first flatten the ontology and the unification problem. The unification problem Γ is called *flat* if it contains only flat subsumptions of the form $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$, where $n \geq 0$ and $C_1, \ldots, C_n, D$ are flat atoms.[3]

Let Γ be a unification problem and $\mathcal{O}$ an $\mathcal{ELH}_{R^+}$-ontology. By introducing auxiliary variables and concept names, respectively, Γ and $\mathcal{O}$ can be transformed in polynomial time into a flat unification problem Γ' and a flat $\mathcal{ELH}_{R^+}$-ontology $\mathcal{O}'$ such that the unifiability status remains unchanged, i.e., Γ has a unifier w.r.t. $\mathcal{O}$ iff Γ' has a unifier w.r.t. $\mathcal{O}'$. In addition, if $\mathcal{O}$ was cycle-restricted, then so is $\mathcal{O}'$ (see [5] for details). Thus, we can assume without loss of generality that the input unification problem and ontology are flat.

4.1 Local Unifiers

The main idea underlying the "in NP" results in [7, 3] is to show that any unification problem that is unifiable has a so-called local unifier.

We denote by At the set of atoms occurring as subdescriptions in subsumptions in Γ or axioms in $\mathcal{O}$ and define

$$\mathsf{At}_{\mathsf{tr}} := \mathsf{At} \cup \{\exists t.D' \mid \exists s.D' \in \mathsf{At},\ t \trianglelefteq_{\mathcal{O}} s,\ t \text{ transitive}\}.$$

Furthermore, we define the set of *non-variable atoms* by $\mathsf{At}_{\mathsf{nv}} := \mathsf{At}_{\mathsf{tr}} \setminus N_v$. Though the elements of $\mathsf{At}_{\mathsf{nv}}$ cannot be variables, they may contain variables if they are of the form $\exists r.X$ for some role r and a variable X.

We call a function S that associates every variable $X \in N_v$ with a set $S_X \subseteq \mathsf{At}_{\mathsf{nv}}$ an *assignment*. Such an assignment induces the following relation $>_S$ on N_v: $>_S$ is the transitive closure of

$$\{(X, Y) \in N_v \times N_v \mid Y \text{ occurs in an element of } S_X\}.$$

We call the assignment S *acyclic* if $>_S$ is irreflexive (and thus a strict partial order). Any acyclic assignment S induces a unique substitution σ_S, which can be defined by induction along $>_S$:

- If $X \in N_v$ is minimal w.r.t. $>_S$, then we define $\sigma_S(X) := \bigsqcap_{D \in S_X} D$.
- Assume that $\sigma(Y)$ is already defined for all Y such that $X >_S Y$. Then we define $\sigma_S(X) := \bigsqcap_{D \in S_X} \sigma_S(D)$.

[3] If $n = 0$, then we have an empty conjunction on the left-hand side, which as usual stands for $\top$.

We call a substitution σ *local* if it is of this form, i.e., if there is an acyclic assignment S such that $\sigma = \sigma_S$. If the unifier σ of Γ w.r.t. $\mathcal{O}$ is a local substitution, then we call it a *local unifier* of Γ w.r.t. $\mathcal{O}$.

The main technical result shown in [3] is that any unifiable $\mathcal{EL}$-unification problem w.r.t. a cycle-restricted ontology has a local unifier. This yields the following brute-force unification algorithm for $\mathcal{EL}$ w.r.t. cycle-restricted ontologies: first guess an acyclic assignment S, and then check whether the induced local substitution σ_S solves Γ. As shown in [3], this algorithm runs in nondeterministic polynomial time. NP-hardness follows from the fact that already unification in $\mathcal{EL}$ w.r.t. the empty ontology is NP-hard [7]. In [3] it is also shown why cycle-restrictedness is needed: there is a non-cycle-restricted $\mathcal{EL}$-ontology $\mathcal{O}$ and an $\mathcal{EL}$-unification problem Γ such that Γ has a unifier w.r.t. $\mathcal{O}$, but it does not have a local unifier.

5 A Goal-Oriented Unification Algorithm

The brute-force algorithm is not practical since it blindly guesses an acyclic assignment and only afterwards checks whether the guessed assignment induces a unifier. In this section, we introduce a more goal-oriented unification algorithm, in which nondeterministic decisions are only made if they are triggered by "unsolved parts" of the unification problem. In addition, failure due to wrong guesses can be detected early. This goal-oriented algorithm generalizes the goal-oriented algorithm for unification in $\mathcal{EL}$ (without background ontology) introduced in [9], though the rules look quite different because here we consider unification problems that consist of subsumptions whereas in [9] we considered equivalences. It is more closely related to the algorithm presented in [2] for unification w.r.t. cycle-restricted $\mathcal{EL}$-ontologies.

We assume that the cycle-restricted $\mathcal{ELH}_{R^+}$-ontology $\mathcal{O}$ and the unification problem Γ_0 are flat. Given $\mathcal{O}$ and Γ_0, the sets At, $\mathsf{At_{tr}}$, and $\mathsf{At_{nv}}$ are defined as above. Starting with Γ_0, the algorithm maintains a current unification problem Γ and a current acyclic assignment S, which initially assigns the empty set to all variables. For each subsumption in Γ it maintains the information on whether it is *solved* or not. Initially, all subsumptions of Γ_0 are unsolved, except those with a variable on the right-hand side. Rules are applied only to unsolved subsumptions. A (non-failing) rule application does the following:

- it solves exactly one unsolved subsumption,
- it may extend the current assignment S, and
- it may introduce new flat subsumptions built from elements of $\mathsf{At_{tr}}$.

Each rule application that extends S additionally *expands* Γ *w.r.t.* X as follows: every subsumption $\mathfrak{s} \in \Gamma$ of the form $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? X$ is *expanded* by adding the subsumption $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? A$ to Γ for every $A \in S_X$.

Subsumptions are only added if they are not already present in Γ. If a new subsumption is added to Γ, either by a rule application or by expansion of Γ, then it is initially designated unsolved, except if it has a variable on the right-hand side. Once a subsumption is in Γ, it will not be removed. Likewise, if a subsumption in Γ is marked as solved, then it will not become unsolved later.

If a subsumption is marked as solved, this does not necessarily mean that it is indeed already solved by the substitution induced by the current assignment. Instead, it may be the case that the task of satisfying the subsumption was deferred to solving other subsumptions, which are "smaller" than the given subsumption in a certain well-defined sense. A subsumption whose right-hand side is a variable is always marked as solved since the task of solving it is deferred to solving the subsumptions introduced by expansion.

The rules of the algorithm consist of the three *eager* rules Eager Ground Solving, Eager Solving, and Eager Extension (see Figure 1), and several *nondeterministic* rules (see Figures 2 and 3). Eager rules are applied with higher priority than nondeterministic rules, and among the eager rules, Eager Ground Solving has the highest priority, then comes Eager Solving, and then Eager Extension.

Algorithm 5. Let Γ_0 be a flat $\mathcal{EL}$-unification problem. We initialize $\Gamma := \Gamma_0$ and $S_X := \emptyset$ for all variables $X \in N_v$. While Γ contains an unsolved subsumption, do the following:

(1) **Eager rule application:** If some eager rules apply to an unsolved subsumption $\mathfrak{s}$ in Γ, apply the one with the highest priority. If the rule application fails, return "not unifiable".

(2) **Nondeterministic rule application:** If no eager rule is applicable, let $\mathfrak{s}$ be an unsolved subsumption in Γ. If one of the nondeterministic rules applies to $\mathfrak{s}$, choose one of these rules and apply it. If none of these rules apply to $\mathfrak{s}$ or the rule application fails, then return "not unifiable".

(3) **Eager application of Decomposition:** If in the previous step one of the rules Mutation 2 or 3 was applied, do the following for all subsumptions $\mathfrak{s}'$ added to Γ by this rule application: If one of the rules Decomposition 1 or 2 applies to $\mathfrak{s}'$, nondeterministically choose one of the applicable decomposition rules and apply it to $\mathfrak{s}'$.[4]

Once all subsumptions in Γ are solved, return the substitution σ induced by the current assignment.

[4]Note that *Decomposition 1* always applies to the new subsumptions. Whether *Decomposition 2* is also applicable depends on the existence of an appropriate transitive role t.

Eager Ground Solving:

Condition: This rule applies to $\mathfrak{s} = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$ if it is ground.
Action: If $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq_{\mathcal{O}} D$ does not hold, the rule application fails. Otherwise, $\mathfrak{s}$ is marked as *solved*.

Eager Solving:

Condition: This rule applies to $\mathfrak{s} = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$ if either

- there is an index $i \in \{1, \ldots, n\}$ such that $C_i = D$ or $C_i = X \in N_v$ and $D \in S_X$, or
- D is ground and $\prod \mathcal{G} \sqsubseteq_{\mathcal{O}} D$ holds, where $\mathcal{G}$ is the set of all ground atoms in $\{C_1, \ldots, C_n\} \cup \bigcup_{X \in \{C_1, \ldots, C_n\} \cap N_v} S_X$.

Action: Its application marks $\mathfrak{s}$ as *solved*.

Eager Extension:

Condition: This rule applies to $\mathfrak{s} = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$ if there is an index $i \in \{1, \ldots, n\}$ with $C_i = X \in N_v$ and $\{C_1, \ldots, C_n\} \setminus \{X\} \subseteq S_X$.
Action: Its application adds D to S_X. If this makes S cyclic, the rule application fails.
Otherwise, Γ is expanded w.r.t. X and $\mathfrak{s}$ is marked as *solved*.

Figure 1: The Eager rules of Algorithm 5.

In step (2), the choice which unsolved subsumption to consider next is don't care nondeterministic. However, choosing which rule to apply to the chosen subsumption is don't know nondeterministic. Additionally, the application of nondeterministic rules requires don't know nondeterministic guessing.

This is an extension of the algorithm from [2] by more general nondeterministic rules and the addition of step (3). This additional step immediately applies all possible Decomposition rules after each application of Mutation 2 or 3. This ensures that in each rule application the generated subsumptions are "smaller" than the triggering subsumption in some well-defined sense, which will be used to prove soundness (see Section 5.1).

The *eager rules* are mainly there for optimization purposes, i.e., to avoid nondeterministic choices if a deterministic decision can be made. For example, a ground subsumption, as considered by *Eager Ground Solving*, either holds, in which case any substitution solves it, or it does not, in which case it does not have a solution. This condition can be checked in polynomial time using the subsumption algorithm for $\mathcal{EL}\mathcal{H}_{R^+}$ [6]. In the case considered by *Eager Solving*, the substitution induced by the current assignment already solves the subsumption. The *Eager Extension* rule solves a subsumption that contains only a variable X

Decomposition 1:

Condition: This rule applies to $\mathfrak{s} = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? \exists s.D'$ if there is an index $i \in \{1, \ldots, n\}$ with $C_i = \exists r.C'$ and $r \trianglelefteq_{\mathcal{O}} s$.
Action: Its application chooses such an index i, adds the subsumption $C' \sqsubseteq^? D'$ to Γ, expands it w.r.t. D' if D' is a variable, and marks $\mathfrak{s}$ as *solved*.

Decomposition 2:

Condition: This rule applies to $\mathfrak{s} = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? \exists s.D'$ if there is an index $i \in \{1, \ldots, n\}$ and a transitive role t with $C_i = \exists r.C'$ and $r \trianglelefteq_{\mathcal{O}} t \trianglelefteq_{\mathcal{O}} s$.
Action: Its application chooses such an index i, adds the subsumption $C' \sqsubseteq^? \exists t.D'$ to Γ and marks $\mathfrak{s}$ as *solved*.

Extension:

Condition: This rule applies to $\mathfrak{s} = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$ if there is an index $i \in \{1, \ldots, n\}$ with $C_i \in N_v$.
Action: Its application chooses such an index i and adds D to S_{C_i}. If this makes S cyclic, the rule application fails. Otherwise, Γ is expanded w.r.t. C_i and $\mathfrak{s}$ is marked as *solved*.

Figure 2: The nondeterministic rules *Decomposition 1 and 2* and *Extension*.

and some elements of S_X on the left-hand side. The rule is motivated by the following observation: for any assignment S' extending the current assignment, the induced substitution σ' satisfies $\sigma'(X) \equiv \sigma'(C_1) \sqcap \cdots \sqcap \sigma'(C_n)$. Thus, if S'_X contains D, then $\sigma'(X) \sqsubseteq \sigma'(D)$, and σ' solves the subsumption. Conversely, if σ' solves the subsumption, then $\sigma'(X) \sqsubseteq \sigma'(D)$, and thus adding D to S'_X yields an equivalent induced substitution.

The *nondeterministic rules* only come into play if no eager rules can be applied. In order to solve an unsolved subsumption $\mathfrak{s} = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$, we consider the two conditions of Lemma 2. Regarding the first condition, which is addressed by the rules *Decomposition 1 and 2* and *Extension*, assume that γ is induced by an acyclic assignment S. To satisfy the first condition of the lemma with γ, the atom $\gamma(D)$ must structurally subsume a top-level atom in $\gamma(C_1) \sqcap \cdots \sqcap \gamma(C_n)$. This atom can either be of the form $\gamma(C_i)$ for a non-variable atom C_i, or of the form $\gamma(C)$ for $C \in S_{C_i}$ and a variable C_i. In the second case, the atom C can either already be in S_{C_i} or it can be put into S_{C_i} by an application of the *Extension* rule. The two versions of *Decomposition* correspond to the cases (2) and (3) in the definition of structural subsumption.[5]

The *Mutation rules* cover the second condition in Lemma 2. Each rule covers a different case: For example, *Mutation 1* is applicable only to subsumptions with more that one conjunct on the left-hand side. It solves the subsumption by making

[5]Case (1) of this definition is already handled by the *Solving* rules.

Mutation 1:

Condition: This rule applies to $\mathfrak{s} = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$ if $n > 1$ and there are atoms $A_1, \ldots, A_k, B$ of $\mathcal{O}$ such that $A_1 \sqcap \cdots \sqcap A_k \sqsubseteq_\mathcal{O} B$ holds.
Action: Its application chooses such atoms, marks $\mathfrak{s}$ as *solved*, and generates the following subsumptions:

- it chooses for each $\eta \in \{1, \ldots, k\}$ an $i \in \{1, \ldots, n\}$ and adds the subsumption $C_i \sqsubseteq^? A_\eta$ to Γ,
- it adds the subsumption $B \sqsubseteq^? D$ to Γ.

Mutation 2:

Condition: This rule applies to $\mathfrak{s} = \exists r.X \sqsubseteq^? D$ if X is a variable, D is ground, and there are atoms $\exists r_1.A_1, \ldots, \exists r_k.A_k$ of $\mathcal{O}$ such that $r \trianglelefteq_\mathcal{O} r_1$, $\ldots$, $r \trianglelefteq_\mathcal{O} r_k$, and $\exists r_1.A_1 \sqcap \cdots \sqcap \exists r_k.A_k \sqsubseteq_\mathcal{O} D$ hold.
Action: Its application chooses such atoms, adds the subsumptions $\exists r.X \sqsubseteq^? \exists r_1.A_1$, $\ldots$, $\exists r.X \sqsubseteq^? \exists r_k.A_k$ to Γ, and marks $\mathfrak{s}$ as *solved*.

Mutation 3:

Condition: This rule applies to $\mathfrak{s} = \exists r.X \sqsubseteq^? \exists s.Y$ if X and Y are variables, and there are atoms $\exists r_1.A_1, \ldots, \exists r_k.A_k, \exists u.B$ of $\mathcal{O}$ such that $r \trianglelefteq_\mathcal{O} r_1$, $\ldots$, $r \trianglelefteq_\mathcal{O} r_k$, $u \trianglelefteq_\mathcal{O} s$, and $\exists r_1.A_1 \sqcap \cdots \sqcap \exists r_k.A_k \sqsubseteq_\mathcal{O} \exists u.B$ hold.
Action: Its application chooses such atoms, adds the subsumptions $\exists r.X \sqsubseteq^? \exists r_1.A_1$, $\ldots$, $\exists r.X \sqsubseteq^? \exists r_k.A_k$, $\exists u.B \sqsubseteq^? \exists s.Y$ to Γ, and marks $\mathfrak{s}$ as *solved*.

Mutation 4:

Condition: This rule applies to $\mathfrak{s} = C \sqsubseteq^? \exists s.Y$ if C is a ground atom or $\top$, Y is a variable, and there is an atom $\exists u.B$ of $\mathcal{O}$ such that either

- $C \sqsubseteq_\mathcal{O} \exists u.B$ and $u \trianglelefteq_\mathcal{O} s$, or
- $C \sqsubseteq_\mathcal{O} \exists t.B$ for a transitive role t with $u \trianglelefteq_\mathcal{O} t \trianglelefteq_\mathcal{O} s$.

Action: Its application chooses such an atom, adds the subsumption $B \sqsubseteq^? Y$ to Γ, and marks $\mathfrak{s}$ as *solved*.

Figure 3: The nondeterministic *Mutation rules* of Algorithm 5.

sure that all conditions of the second condition of Lemma 2 are satisfied. The rule guesses atoms $A_1, \ldots, A_k, B$ of $\mathcal{O}$ such that $A_1 \sqcap \cdots \sqcap A_k \sqsubseteq_{\mathcal{O}} B$ holds. This can be checked using the polynomial-time subsumption algorithm for $\mathcal{ELH}_{R^+}$. Whenever the second condition of Lemma 2 requires a structural subsumption $\gamma(E) \sqsubseteq^{\mathsf{s}}_{\mathcal{O}} \gamma(F)$ to hold for a (hypothetical) unifier γ of Γ, the rule creates the new subsumption $E \sqsubseteq^? F$, which has to be solved later on. This way, the rule ensures that the substitution built by the algorithm actually satisfies the conditions of the lemma. The *other Mutation rules* follow the same idea, but they consider cases where only a single atom occurs on the left-hand side of the subsumption to be solved. The reason for considering these cases separately is that in the proof of soundness we need the newly introduced subsumptions to be "smaller" than the subsumption that triggered their introduction. For *Mutation 1* this is the case due to the smaller left-hand side (only one atom), whereas for the other mutation rules this is not so clear. Actually, for *Mutation 2 and 3*, the new subsumptions turn out to be smaller only after *Decomposition* is applied to them. *Mutation 4* implicitly applies a form of decomposition.

In contrast to the algorithm presented in [2] for unification w.r.t. cycle-restricted $\mathcal{EL}$-ontologies, this algorithm uses two modified Decomposition rules instead of only one to account for the generalized definition of structural subsumption. Additionally, the new algorithm separates the implicit application of the Decomposition rules in the rules *Mutation 2 and 3* into a separate step in the algorithm. For *Mutation 1* this is not necessary, while for *Mutation 4* it is not possible since applying condition (3) of the definition of structural subsumption to the resulting subsumption could lead to non-termination (see the proof of Lemma 8 for details).

5.1 Soundness

The soundness of the unification algorithm in [2] is shown with the help of a measure assigned to all subsumptions generated during the process of computation. The measure is equipped with a well-founded order $\succ$, which is used for an induction argument. In order to show that the modified algorithm is also sound, we have to slightly modify the measure and show that it works for the new rules, i.e., the measure of the subsumptions obtained by applications of these rules is smaller than the measure of the unsolved subsumptions to which they were applied.

In the following, let S be the final assignment computed by Algorithm 5 on input Γ_0 and σ be the substitution induced by S. With $\widehat{\Gamma}$ we denote the final set of subsumptions computed by this run, i.e., the original subsumptions of Γ_0 together with the new ones generated by rule applications.

Definition 6. Let $\mathfrak{s} = C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? C_{n+1} \in \widehat{\Gamma}$.

- $\mathfrak{s}$ is *small* if $n = 1$ and C_1 is ground or C_{n+1} is ground.

- We define $m(\mathfrak{s}) := (m_1(\mathfrak{s}), m_2(\mathfrak{s}), m_3(\mathfrak{s}), m_4(\mathfrak{s}))$, where
  - $m_1(\mathfrak{s}) := 0$ if $\mathfrak{s}$ is small, and $m_1(\mathfrak{s}) := 1$ otherwise.
  - $m_2(\mathfrak{s}) := X$ if $C_{n+1} = X$ or $C_{n+1} = \exists r.X$ for a variable X and a role name $r \in N_R$, and $m_2(\mathfrak{s}) := \bot$ otherwise.
  - $m_3(\mathfrak{s}) := \max\{\mathsf{rd}(\sigma(C_i)) \mid i \in \{1, \ldots, n\}\}$.
  - $m_4(\mathfrak{s}) := \mathsf{rd}(\sigma(C_{n+1}))$
- The strict partial order $\succ$ on such tuples is the lexicographic order, where the first, third, and fourth components are compared w.r.t. the normal order $>$ on natural numbers. The variables in the second component are compared w.r.t. the relation $>_S$ induced by the acyclic assignment and $\bot$ is smaller than any variable.
- We extend $\succ$ to $\widehat{\Gamma}$ by setting $\mathfrak{s}_1 \succ \mathfrak{s}_2$ iff $m(\mathfrak{s}_1) \succ m(\mathfrak{s}_2)$.

As in the proof of Lemma 30 in [2], we use induction on this well-founded strict partial order to show that σ solves $\widehat{\Gamma}$.

Lemma 7. *σ is a unifier of $\widehat{\Gamma}$ w.r.t. $\mathcal{O}$, and thus also of its subset Γ_0.*

Proof. Let $\mathfrak{s} \in \widehat{\Gamma}$ and assume that σ solves all subsumptions $\mathfrak{s}' \in \widehat{\Gamma}$ with $\mathfrak{s}' \prec \mathfrak{s}$. There are several cases to consider depending on the way $\mathfrak{s}$ was solved.

- If $\mathfrak{s}$ has a *non-variable atom on the right-hand side*, then it was initially marked as unsolved and must have been solved by a successful rule application. The cases of the Eager rules and the Extension rule can be handled exactly as in [2] since the measure is not needed. We now consider the new nondeterministic rules.
  - Decomposition 1: Then $\mathfrak{s}$ is of the form $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? \exists s.D'$ with $C_i = \exists r.C'$ and $r \trianglelefteq_{\mathcal{O}} s$ and we have $\mathfrak{s}' = C' \sqsubseteq^? D' \in \widehat{\Gamma}$. We will show that $\mathfrak{s} \succ \mathfrak{s}'$ holds. By induction, this implies that σ solves $\mathfrak{s}'$, and by Lemma 2 thus also $\mathfrak{s}$. To compare $m(\mathfrak{s})$ and $m(\mathfrak{s}')$, we observe first that $m_2(\mathfrak{s}) = m_2(\mathfrak{s}')$. We now make a case distinction on $m_1(\mathfrak{s}')$.

    If $\mathfrak{s}'$ is small, then $\mathfrak{s}$ is either not small, i.e., $m_1(\mathfrak{s}) > m_1(\mathfrak{s}')$, or small and of the form $\exists r.C' \sqsubseteq^? \exists s.D'$. In the second case, we have $m_1(\mathfrak{s}) = m_1(\mathfrak{s}')$ and $m_3(\mathfrak{s}) > m_3(\mathfrak{s}')$.

    If $\mathfrak{s}'$ is not small, then both C' and D' are variables, and thus $\mathfrak{s}$ is also not small, which yields $m_1(\mathfrak{s}) = m_1(\mathfrak{s}')$. Furthermore,

    $$m_3(\mathfrak{s}) \geq \mathsf{rd}(\sigma(\exists r.C')) = \mathsf{rd}(\sigma(C')) + 1 > \mathsf{rd}(\sigma(C')) = m_3(\mathfrak{s}').$$

    Thus, in all cases we have $\mathfrak{s} \succ \mathfrak{s}'$.

– Decomposition 2: Then $\mathfrak{s}$ is of the form $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? \exists s.D'$ with $C_i = \exists r.C'$ and we have $\mathfrak{s}' = C' \sqsubseteq^? \exists t.D' \in \widehat{\Gamma}$ for some transitive role t with $r \trianglelefteq_{\mathcal{O}} t \trianglelefteq_{\mathcal{O}} s$. Again, $\mathfrak{s} \succ \mathfrak{s}'$ can be shown in exactly the same way as in the case of Decomposition 1.

– Mutation 1: Then $\mathfrak{s}$ is of the form $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$ with $n > 1$ and there are atoms $A_1, \ldots, A_k, B$ of $\mathcal{T}$ with $A_1 \sqcap \cdots \sqcap A_k \sqsubseteq_{\mathcal{O}} B$. Furthermore, for every $\eta \in \{1, \ldots, k\}$ there is a subsumption $\mathfrak{s}_\eta = C_i \sqsubseteq^? A_\eta \in \widehat{\Gamma}$ for some $i \in \{1, \ldots, n\}$ and the subsumption $\mathfrak{s}' = B \sqsubseteq^? D$ is also in $\widehat{\Gamma}$.

  Since $\mathfrak{s}$ is not small and all the subsumptions $\mathfrak{s}_1, \ldots, \mathfrak{s}_k$, $\mathfrak{s}'$ are small, they are smaller than $\mathfrak{s}$ w.r.t. $\succ$. By induction, σ solves those small subsumptions, and thus we have $\sigma(C_1) \sqcap \cdots \sqcap \sigma(C_n) \sqsubseteq_{\mathcal{O}} A_1 \sqcap \cdots \sqcap A_k \sqsubseteq_{\mathcal{O}} B \sqsubseteq_{\mathcal{O}} \sigma(D)$, i.e., σ solves $\mathfrak{s}$.

– Mutation 2: Then $\mathfrak{s}$ is of the form $\exists r.X \sqsubseteq^? D$, D is ground, and $\exists r_1.A_1 \sqcap \cdots \sqcap \exists r_k.A_k \sqsubseteq_{\mathcal{O}} D$ holds for atoms $\exists r_1.A_1, \ldots, \exists r_k.A_k$ of $\mathcal{O}$ with $r \trianglelefteq_{\mathcal{O}} s1$, …, $r \trianglelefteq_{\mathcal{O}} r_k$. We take a look at one of the produced subsumptions $\exists r.X \sqsubseteq^? \exists r_\eta.A_\eta$ for $\eta \in \{1, \ldots, k\}$.

  Itself, this subsumption is not smaller than $\mathfrak{s}$ w.r.t. $\succ$. However, one or more of the Decomposition rules applies to it, and thus there must also be a subsumption $\mathfrak{s}' = X \sqsubseteq^? A_\eta$ or $\mathfrak{s}'' = X \sqsubseteq^? \exists t.A_\eta$ with $r \trianglelefteq_{\mathcal{O}} t \trianglelefteq_{\mathcal{O}} r_\eta$ in $\widehat{\Gamma}$. Both of these are smaller than $\mathfrak{s}$ w.r.t. $\succ$ since they are both small, their right-hand sides are ground, and $m_3(\mathfrak{s}') = m_3(\mathfrak{s}'') = \mathsf{rd}(\sigma(X)) < \mathsf{rd}(\sigma(\exists r.X)) = m_3(\mathfrak{s})$. By induction, the produced subsumption is solved by σ, which implies by Lemma 2 that $\sigma(\exists r.X) \sqsubseteq_{\mathcal{O}} \exists r_\eta.A_\eta$.

  Since this holds for all $\eta \in \{1, \ldots, k\}$, we conclude that $\sigma(\exists r.X) \sqsubseteq_{\mathcal{O}} \exists r.A_1 \sqcap \cdots \sqcap \exists r.A_k \sqsubseteq_{\mathcal{O}} D$.

– Mutation 3: Then $\mathfrak{s}$ is of the form $\exists r.X \sqsubseteq^? \exists s.Y$ and the subsumption $\exists r_1.A_1 \sqcap \cdots \sqcap \exists r_k.A_k \sqsubseteq_{\mathcal{O}} \exists u.B$ holds between atoms of $\mathcal{O}$ with $r \trianglelefteq_{\mathcal{O}} r_1$, …, $r \trianglelefteq_{\mathcal{O}} r_k$, $u \trianglelefteq_{\mathcal{O}} s$. As above, we can show that the produced subsumptions $\exists r.X \sqsubseteq^? \exists r_\eta.A_\eta$ for $\eta \in \{1, \ldots, k\}$ are solved by σ. We now consider the remaining subsumption $\exists u.B \sqsubseteq^? \exists s.Y$.

  Again, one or more of the Decomposition rules applies to it, and thus the subsumption $\mathfrak{s}' = B \sqsubseteq^? Y$ or $\mathfrak{s}'' = B \sqsubseteq^? \exists t.Y$ with $u \trianglelefteq_{\mathcal{O}} t \trianglelefteq_{\mathcal{O}} s$ must be in $\widehat{\Gamma}$. Both are smaller than $\mathfrak{s}$ w.r.t. $\succ$ since they are both small while $\mathfrak{s}$ is not. By induction, the produced subsumption is solved by σ, and thus we have $\exists u.B \sqsubseteq_{\mathcal{O}} \sigma(\exists s.Y)$ by Lemma 2.

  We can conclude that $\sigma(\exists r.X) \sqsubseteq_{\mathcal{O}} \exists r_1.A_1 \sqcap \cdots \sqcap \exists r_k.A_k \sqsubseteq_{\mathcal{O}} \exists u.B \sqsubseteq_{\mathcal{O}} \sigma(\exists s.Y)$.

– Mutation 4: Then $\mathfrak{s}$ is of the form $C \sqsubseteq^? \exists s.Y$, C is ground, and there is an atom $\exists u.B$ of $\mathcal{O}$ with $u \trianglelefteq_{\mathcal{O}} s$ such that either $C \sqsubseteq_{\mathcal{O}} \exists u.B$ or $C \sqsubseteq_{\mathcal{O}} \exists t.B$ for a transitive role t with $u \trianglelefteq_{\mathcal{O}} t \trianglelefteq_{\mathcal{O}} s$. Furthermore, the subsumption $\mathfrak{s}' = B \sqsubseteq^? Y$ is in $\widehat{\Gamma}$. Both $\mathfrak{s}$ and $\mathfrak{s}'$ are small and $m_2(\mathfrak{s}) =$

$Y = m_2(\mathfrak{s}')$. Since B is a constant, i.e., has depth 0, we have $m_3(\mathfrak{s}) \geq m_3(\mathfrak{s}')$. Finally, $m_4(\mathfrak{s}) = \mathsf{rd}(\sigma(\exists s.Y)) > \mathsf{rd}(\sigma(Y)) = m_4(\mathfrak{s}')$, and thus $\mathfrak{s} \succ \mathfrak{s}'$. By induction, $B \sqsubseteq_{\mathcal{O}} \sigma(Y)$ holds, and thus Lemma 2 implies that either $C \sqsubseteq_{\mathcal{O}} \exists u.B \sqsubseteq_{\mathcal{O}} \sigma(\exists s.Y)$ or $C \sqsubseteq_{\mathcal{O}} \exists t.B \sqsubseteq_{\mathcal{O}} \sigma(\exists s.Y)$.

- The case that $\mathfrak{s}$ has a *variable as its right-hand side* can again be handled exactly as in the proof of Lemma 30 in [2]. □

5.2 Completeness

Assume that Γ_0 is unifiable w.r.t. $\mathcal{O}$ and let γ be a ground unifier of Γ_0 w.r.t. $\mathcal{O}$. As in [2], we can use this unifier to find a successful computation path of Algorithm 5 on Γ_0 such that the following invariants are preserved by the successive rule applications for the current set of subsumptions Γ and the current assignment S:

(I) γ is a unifier of Γ.

(II) For all $B \in S_X$, we have $\gamma(X) \sqsubseteq_{\mathcal{O}} \gamma(B)$.

In [2], it is shown that these invariants are maintained by expanding Γ and by applying eager rules. We also know that invariant (II) guarantees that the current assignment is always acyclic. We now reprove a variant of Lemma 34 from [2] that deals with the modified nondeterministic rules.

Lemma 8. *Let $\mathfrak{s}$ be an unsolved subsumption of Γ to which no eager rule applies. Then there is a nondeterministic rule that can be successfully applied to $\mathfrak{s}$ while maintaining the invariants.*

Proof. $\mathfrak{s}$ must be of the form $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? D$, where $C_1, \ldots, C_n$ are flat atoms from At or $\top$ and $D \in \mathsf{At}_{\mathsf{nv}}$. By invariant (I), γ solves $\mathfrak{s}$, i.e., we have $\gamma(C_1) \sqcap \cdots \sqcap \gamma(C_n) \sqsubseteq_{\mathcal{O}} \gamma(D)$. By Lemma 2, one of the following alternatives holds:

1. There is an index $i \in \{1, \ldots, n\}$ such that $E \sqsubseteq^{\mathsf{s}}_{\mathcal{O}} \gamma(D)$ for a top-level atom E of $\gamma(C_i)$. We consider the following cases for C_i, which can obviously not be $\top$.
   - If C_i is a concept name, then $C_i = E = D$ and Eager Solving is applicable to $\mathfrak{s}$, which contradicts the assumption.
   - If $C_i = \exists r.C'$, then $\gamma(C_i) = \exists r.\gamma(C') = E$, and thus $D = \exists s.D'$ and $r \trianglelefteq_{\mathcal{O}} s$ and either (i) $\gamma(C') \sqsubseteq_{\mathcal{O}} \gamma(D')$ or (ii) $\gamma(C') \sqsubseteq_{\mathcal{O}} \exists t.\gamma(D')$ for a transitive role t with $r \trianglelefteq_{\mathcal{O}} t \trianglelefteq_{\mathcal{O}} s$. In case (i), Decomposition 1 can be successfully applied to $\mathfrak{s}$ and results in a new subsumption $C' \sqsubseteq^? D'$ that is solved by γ. Similarly, in case (ii) Decomposition 2 can be applied.

- If $C_i = X$ is a variable, then invariant (II) is preserved by adding D to S_X since $\gamma(X) \sqsubseteq E \sqsubseteq_{\mathcal{O}} \gamma(D)$. This implies that S stays acyclic, and thus we can successfully apply Extension to $\mathfrak{s}$.

2. There are atoms $A_1, \dots, A_k, B$ of $\mathcal{O}$ such that for all $\eta \in \{1, \dots, k\}$ there is $i \in \{1, \dots, n\}$ with $\gamma(C_i) \sqsubseteq_{\mathcal{O}} E \sqsubseteq^{\mathsf{s}}_{\mathcal{O}} A_\eta$ for a top-level atom E of $\gamma(C_i)$, $A_1 \sqcap \dots \sqcap A_k \sqsubseteq_{\mathcal{O}} B$, and $B \sqsubseteq^{\mathsf{s}}_{\mathcal{O}} \gamma(D)$. If $n > 1$, we can apply Mutation 1 in such a way that all created subsumptions are solved by γ. The possible subsequent applications of Decomposition rules again preserve the invariants and cannot fail.

   If $n = 1$, we distinguish several cases for C_1 and D:

   - If both C_1 and D are ground, then the Eager Ground Solving rule is applicable to $\mathfrak{s}$, which contradicts our assumption.
   - If $C_1 = X$ is a variable, then the Eager Extension rule is applicable to $\mathfrak{s}$, which again contradicts our assumption.
   - If $C_1 = \exists r.X$ for a variable X, then we have $A_\eta = \exists r_\eta.A'_\eta$ with $r \trianglelefteq_{\mathcal{O}} r_\eta$ and $\gamma(\exists r.X) \sqsubseteq_{\mathcal{O}} A_\eta$ for every $\eta \in \{1, \dots, k\}$. Thus, we can add the subsumptions $C \sqsubseteq^? A_1$, $\dots$, $C \sqsubseteq^? A_k$ to Γ $A'_1, \dots, A'_k$ without violating invariant (I). If D is ground, we have $A_1 \sqcap \dots \sqcap A_k \sqsubseteq_{\mathcal{O}} B \sqsubseteq_{\mathcal{O}} D$, and thus we can successfully apply Mutation 2 to $\mathfrak{s}$. Otherwise, $D = \exists s.Y$ for a variable Y, which implies that $B = \exists u.B'$ with $u \trianglelefteq_{\mathcal{O}} s$ and $B \sqsubseteq_{\mathcal{O}} \gamma(D)$. In this case, we can apply Mutation 3 to $\mathfrak{s}$ while preserving the invariants.
   - If C_1 is ground and $D = \exists s.Y$ for a variable Y, then we have

     $$C_1 \sqsubseteq_{\mathcal{O}} A_1 \sqcap \dots \sqcap A_k \sqsubseteq_{\mathcal{O}} B = \exists u_1.B_1$$

     and $u_1 \trianglelefteq_{\mathcal{O}} s$ and either (i) $B_1 \sqsubseteq_{\mathcal{O}} \gamma(Y)$ or (ii) $B_1 \sqsubseteq_{\mathcal{O}} \exists t_1.\gamma(Y)$ for some transitive role t_1 with $u_1 \trianglelefteq_{\mathcal{O}} t_1 \trianglelefteq_{\mathcal{O}} s$. In case (i), we can successfully apply Mutation 4 to $\mathfrak{s}$ while maintaining the invariants.

     In case (ii), note that the resulting subsumption $B_1 \sqsubseteq_{\mathcal{O}} \gamma(\exists t_1.Y)$ is similar to the original $C_1 \sqsubseteq_{\mathcal{O}} \gamma(\exists s.Y)$ since B_1 is also ground. Since B_1 is a concept name, again the second case of Lemma 2 applies and we can derive the same consequences as above. In particular, there is an atom $\exists u_2.B_2$ of $\mathcal{O}$ such that $B_1 \sqsubseteq_{\mathcal{O}} \exists u_2.B_2$ and $u_2 \trianglelefteq_{\mathcal{O}} t_1$ and either (i) $B_2 \sqsubseteq_{\mathcal{O}} \gamma(Y)$ or (ii) $B_2 \sqsubseteq_{\mathcal{O}} \exists t_2.\gamma(Y)$ for some transitive role t_2 with $u_2 \trianglelefteq_{\mathcal{O}} t_2 \trianglelefteq_{\mathcal{O}} t_1$. An important consequence is that

     $$C_1 \sqsubseteq_{\mathcal{O}} \exists u_1.B_1 \sqsubseteq_{\mathcal{O}} \exists u_1.\exists u_2.B_2 \sqsubseteq_{\mathcal{O}} \exists t_1.\exists t_1.B_2 \sqsubseteq_{\mathcal{O}} \exists t_1.B_2$$

     since both u_1 and u_2 are subroles of the transitive role t_1. In case (i), we can thus again successfully apply Mutation 4 to $\mathfrak{s}$.

     In case (ii), we can apply the same case analysis as before. This process cannot go on indefinitely since at some point the same concept name B_i

would occur twice and then we would have a subsumption $B_i \sqsubseteq_{\mathcal{O}} \exists t_1.B_i$, which is impossible since $\mathcal{O}$ is cycle-restricted. Thus, at some point we must find an atom $\exists u_k.B_k$ of $\mathcal{O}$ such that $u_k \trianglelefteq_{\mathcal{O}} t_1$, $C_1 \sqsubseteq_{\mathcal{O}} \exists t_1.B_k$, and $B_k \sqsubseteq_{\mathcal{O}} \gamma(Y)$, i.e., we can successfully apply Mutation 4 to $\mathfrak{s}$. □

To summarize, we have shown that for any unifiable input problem Γ_0 there is a non-failing run of Algorithm 5 on Γ_0 during which the invariants (I) and (II) are satisfied. Together with the fact that any run of the algorithm terminates (see below), this shows completeness, i.e., whenever Γ_0 has a unifier w.r.t. $\mathcal{O}$, the algorithm computes one.

5.3 Termination and Complexity

Similar termination arguments as in [2] also hold for the algorithm with the new nondeterministic rules. The only remark we have to make concerns the number and the form of the subsumptions created in the course of a computation. These subsumptions can have one of the following forms:

1. subsumptions from Γ_0;
2. subsumptions created by expansion of Γ_0: these are of the form $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? A$ for a subsumption $C_1 \sqcap \cdots \sqcap C_n \sqsubseteq^? X \in \Gamma_0$ and $A \in \mathsf{At_{nv}}$,
3. subsumptions of the form $C \sqsubseteq D$, where $C \in \mathsf{At}, D \in \mathsf{At_{tr}}$.

Hence for the modified algorithm, the set of atoms appearing in the subsumptions is larger since the right-hand sides can be from $\mathsf{At_{tr}}$ and not only from At. But the increase is only polynomial in the size of the input and therefore there are only polynomially many subsumptions of the above forms.

Hence, since each rule application solves one subsumption and each such application takes only polynomial time, the modified algorithm is sound and complete and terminates in time polynomial in the size of the input Γ_0 and $\mathcal{O}$.

Theorem 9. *Algorithm 5 is an* NP*-decision procedure for unifiability in* $\mathcal{ELH}_{R^+}$ *w.r.t. cycle-restricted ontologies.*

6 Conclusions

We have presented a goal-oriented NP-algorithm for unification in $\mathcal{ELH}_{R^+}$ w.r.t. cycle-restricted ontologies. In [4], we have developed a reduction of this problem to SAT, which is based on a characterization of subsumption different from the one

in Lemma 2. Though clearly better than the brute-force algorithm introduced in [3], both algorithms suffer from a high degree of nondeterminism due to having to guess true subsumptions between concepts built from atoms of the background cycle-restricted ontology. We must find optimizations to tackle this problem before an implementation becomes feasible.

On the theoretical side, the main topic for future research is to consider unification w.r.t. unrestricted $\mathcal{EL}\mathcal{H}_{R^+}$-ontologies. In order to generalize the brute-force algorithm in this direction, we need to find a more general notion of locality. Starting with the goal-oriented algorithm, one idea could be not to fail when a cyclic assignment is generated, but rather to add rules that can break such cycles, similar to what is done in procedures for general E-unification [14].

Another idea could be to use just the rules of our goal-oriented algorithm, and not fail when a cyclic assignment S is generated. Our conjecture is that then the background ontology $\mathcal{O}$ together with the cyclic TBox $\mathcal{T}_S := \{X \equiv \bigsqcap_{C \in S_X} C \mid X \in N_v\}$ induced by S satisfies $C \sqsubseteq_{\mathcal{O} \cup \mathcal{T}_S} D$ for all subsumptions $C \sqsubseteq^? D$ in Γ_0 if an appropriate hybrid semantics [12] for the combined ontology $\mathcal{O} \cup \mathcal{T}_S$ is used.

All the results on unification in Description Logics mentioned here are restricted to relatively inexpressive logics that do not support all Boolean operators. If we close $\mathcal{EL}$ under negation, then we obtain the DL $\mathcal{ALC}$, which corresponds to the modal logic K [15]. Whether unification in K is decidable is a long-standing open problem. It is only known that relatively minor extensions of K have an undecidable unification problem [16].

References

[1] Franz Baader, Stefan Borgwardt, Julian Mendez, and Barbara Morawska. UEL: Unification solver for $\mathcal{EL}$. In *Proc. of the 25th Int. Workshop on Description Logics (DL'12)*, volume 846 of *CEUR Workshop Proceedings*, 2012.

[2] Franz Baader, Stefan Borgwardt, and Barbara Morawska. Unification in the description logic $\mathcal{EL}$ w.r.t. cycle-restricted TBoxes. LTCS-Report 11-05, Chair for Automata Theory, Institute for Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany, 2011. See http://lat.inf.tu-dresden.de/research/reports.html.

[3] Franz Baader, Stefan Borgwardt, and Barbara Morawska. Extending unification in $\mathcal{EL}$ towards general TBoxes. In *Proc. of the 13th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'12)*, pages 568–572. AAAI Press, 2012. Short paper.

[4] Franz Baader, Stefan Borgwardt, and Barbara Morawska. SAT-encoding of unification in $\mathcal{EL}\mathcal{H}_{R^+}$ w.r.t. cycle-restricted ontologies. In *Proc. of the*

6th Int. Joint Conf. on Automated Reasoning (IJCAR'12), volume 7364 of *Lecture Notes in Artificial Intelligence*, pages 30–44. Springer-Verlag, 2012.

[5] Franz Baader, Stefan Borgwardt, and Barbara Morawska. SAT encoding of unification in $\mathcal{ELH}_{R^+}$ w.r.t. cycle-restricted ontologies. LTCS-Report 12-02, Chair for Automata Theory, Institute for Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany, 2012. See http://lat.inf.tu-dresden.de/research/reports.html.

[6] Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the $\mathcal{EL}$ envelope. In *Proc. of the 19th Int. Joint Conf. on Artificial Intelligence (IJCAI'05)*, pages 364–369. Professional Book Center, 2005.

[7] Franz Baader and Barbara Morawska. Unification in the description logic $\mathcal{EL}$. In *Proc. of the 20th Int. Conf. on Rewriting Techniques and Applications (RTA'09)*, volume 5595 of *Lecture Notes in Computer Science*, pages 350–364. Springer-Verlag, 2009.

[8] Franz Baader and Barbara Morawska. SAT encoding of unification in $\mathcal{EL}$. In *Proc. of the 17th Int. Conf. on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'10)*, volume 6397 of *Lecture Notes in Computer Science*, pages 97–111. Springer-Verlag, 2010.

[9] Franz Baader and Barbara Morawska. Unification in the description logic $\mathcal{EL}$. *Logical Methods in Computer Science*, 6(3), 2010.

[10] Franz Baader and Paliath Narendran. Unification of concept terms in description logics. *Journal of Symbolic Computation*, 31(3):277–305, 2001.

[11] Sebastian Brandt. Polynomial time reasoning in a description logic with existential restrictions, GCI axioms, and - what else? In *Proc. of the 16th Eur. Conf. on Artificial Intelligence (ECAI'04)*, pages 298–302. IOS Press, 2004.

[12] Sebastian Brandt and Jörg Model. Subsumption in $\mathcal{EL}$ w.r.t. hybrid TBoxes. In *Proc. of the 28th German Conf. on Artificial Intelligence (KI'05)*, volume 3698 of *LNCS*, pages 34–48. Springer, 2005.

[13] Carla P. Gomes, Henry Kautz, Ashish Sabharwal, and Bart Selman. Satisfiability solvers. In *Handbook of Knowledge Representation*, pages 89–134. Elsevier, 2008.

[14] Barbara Morawska. General E-unification with eager variable elimination and a nice cycle rule. *Journal of Automated Reasoning*, 39(1):77–106, 2007.

[15] Klaus Schild. A correspondence theory for terminological logics: Preliminary report. In *Proc. of the 12th Int. Joint Conf. on Artificial Intelligence (IJCAI'91)*, pages 466–471, 1991.

[16] Frank Wolter and Michael Zakharyaschev. Undecidability of the unification and admissibility problems for modal and description logics. *ACM Transactions on Computational Logic*, 9(4), 2008.