Computing Minimal $\mathcal{EL}$-unifiers is Hard

Franz Baader Stefan Borgwardt Barbara Morawska [1]

Theoretical Computer Science, TU Dresden, Germany

Abstract

Unification has been investigated both in modal logics and in description logics, albeit with different motivations. In description logics, unification can be used to detect redundancies in ontologies. In this context, it is not sufficient to decide unifiability, one must also compute appropriate unifiers and present them to the user. For the description logic $\mathcal{EL}$, which is used to define several large biomedical ontologies, deciding unifiability is an NP-complete problem. It is known that every solvable $\mathcal{EL}$-unification problem has a *minimal unifier*, and that every minimal unifier is a *local unifier*. Existing unification algorithms for $\mathcal{EL}$ compute all minimal unifiers, but additionally (all or some) non-minimal local unifiers. Computing only the minimal unifiers would be better since there are considerably less minimal unifiers than local ones, and their size is usually also quite small.
In this paper we investigate the question whether the known algorithms for $\mathcal{EL}$-unification can be modified such that they compute exactly the minimal unifiers without changing the complexity and the basic nature of the algorithms. Basically, the answer we give to this question is negative.

Keywords: Unification, Description Logics, Complexity

1 Introduction

It is well-known that there is a close connection between modal logics (MLs) and description logics (DLs). In fact, many DLs are syntactic variants of classical MLs. Unification has been introduced in both areas [5], with the same formal meaning, but with different applications in mind. In ML, unification [16,17,23] was mainly investigated in the context of the admissibility problem for inference rules [22,18,12]. Unification is simpler than the admissibility problem in the sense that it can easily be reduced to it, but in some cases (e.g., if the unification problem is effectively finitary, i.e., finite complete sets of unifiers can be computed) there is also a reduction in the other direction (see, e.g., [20]). An important open problem in the area is the question whether unification in the basic modal logic K, which corresponds to the DL $\mathcal{ALC}$, is decidable. It is only know that relatively minor extensions of K have an undecidable unification problem [24].

[1] Supported by DFG under grant BA 1122/14-1

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

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

whereas another one represents it as

$$\mathsf{Patient} \sqcap \exists \mathsf{finding}.(\mathsf{Severe\_finding} \sqcap \mathsf{Injury} \sqcap \exists \mathsf{finding\_site}.\mathsf{Head}). \tag{2}$$

Formally, 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 $\mathsf{Head\_injury}$ and $\mathsf{Severe\_finding}$ as variables, and substituting the first one by $\mathsf{Injury} \sqcap \exists \mathsf{finding\_site}.\mathsf{Head}$ and the second one by $\exists \mathsf{severity}.\mathsf{Severe}$. In this case, we say that the descriptions are unifiable, and call the substitution that makes them equivalent a *unifier*. Intuitively, such a unifier proposes definitions for the concept names that are used as variables: in our example, we know that, if we define $\mathsf{Head\_injury}$ as $\mathsf{Injury} \sqcap \exists \mathsf{finding\_site}.\mathsf{Head}$ and $\mathsf{Severe\_finding}$ as $\exists \mathsf{severity}.\mathsf{Severe}$, then the two concept descriptions (1) and (2) are equivalent w.r.t. these definitions.

Of course, this example was constructed such that the unifier actually provides sensible definitions for the concept names used as variables. In general, the existence of a unifier only says that there is a structural similarity between the two concepts. The developer that uses unification as a tool for finding redundancies in an ontology or between two different ontologies needs to inspect the unifier(s) to see whether the definitions it suggests really make sense. Thus, a decision procedure for unifiability is not sufficient in this context. One needs a procedure that also produces appropriate unifiers.

Due to the fact that the decidability status of unification in the DL $\mathcal{ALC}$ is a long-standing open problem (at least in its ML variant of unification in K), the work on unification in DLs has mostly concentrated on sub-Boolean fragments of K. Originally, unification in DLs has been investigated in [10] for the DL $\mathcal{FL}_0$, which offers the constructors conjunction ($\sqcap$), value restrictions ($\forall r.C$), and the top-concept ($\top$). However, the usability of unification in this DL is impaired by the facts that, on the one hand, there are almost no ontologies that use only $\mathcal{FL}_0$, and on the other hand, the complexity of the unification problem is quite high (ExpTime-complete).

In this paper, we consider unification in the DL $\mathcal{EL}$, which differs from $\mathcal{FL}_0$ by offering existential restrictions ($\exists r.C$) in place of value restrictions, and thus corresponds to the fragment of K that uses only diamond, conjunction, and the truth constant "true." $\mathcal{EL}$ has recently drawn considerable attention since, on the one hand, important inference problems such as the subsumption problem are polynomial in $\mathcal{EL}$ [1,13]. On the other hand, though quite inexpressive, $\mathcal{EL}$ can be used to define biomedical ontologies. For example, both the large medical ontology SNOMED CT and the Gene Ontology [2] can be expressed in

[2] see http://www.ihtsdo.org/snomed-ct/ and http://www.geneontology.org/

$\mathcal{EL}$. In [7], we were able to show that unification in $\mathcal{EL}$ is of considerably lower complexity than unification in $\mathcal{FL}_0$: the decision problem for $\mathcal{EL}$ is NP-complete. The main steps in the proof of this statement given in [7] were the following. First, the inverse subsumption order on concept descriptions was used to define an order on substitutions:

$$\sigma \succeq \theta \text{ iff } \sigma(X) \sqsubseteq \theta(X) \text{ holds for all variables } X,$$

and it was shown that this order is well-founded. As an immediate consequence of the well-foundedness of $\succeq$, every solvable unification problem has a minimal unifier. Second, it was shown that every minimal unifier is a local substitution, where local substitutions are built from a polynomial number of so-called atoms determined by the unification problem. Finally, a brute-force "guess and then test" NP-algorithm was described, which guesses a local substitution and then checks (in polynomial time) whether it is a unifier.

An obvious disadvantage of this brute-force algorithm is that it blindly guesses a local substitution and only afterwards checks whether the guessed substitution is a unifier. Thus, in general many substitutions will be generated that only in the subsequent check turn out not to be unifiers. In contrast, the SAT reduction presented in [8] is such that only unifiers are generated. To be more precise, it was shown in [8] how a given unification problem Γ can be translated in polynomial time into a propositional formula ϕ_Γ such that the satisfying valuations of ϕ_Γ correspond to the local unifiers of Γ. The translation into SAT allows us to employ existing highly optimized state-of-the-art SAT solvers for implementing the unification algorithm. While this yields a quite efficient decision procedure for unifiability, the fact that all local unifiers, rather than only minimal ones, are generated turned out to be problematic if one wants to show the unifiers to the user. In fact, even very small unification problems can have hundreds of local unifiers, many of which do not make sense in the application. The set of all minimal unifiers is a subset of the set of all local unifiers, whose cardinality is usually much smaller. [3] Another advantage of minimal unifiers is that they are usually of smaller size (where the size of a substitution is the sum of the sizes of the concept terms substituted for the variables), and are thus easier to read and comprehend.

In [9] we describe 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. By construction, this algorithm can only compute local unifiers, and it is shown in [9] that all minimal ones are among the ones computed by it. Though in our initial tests the number of unifiers computed by the goal-oriented algorithm turned out to be usually much smaller than of the ones computed by the SAT reduction, the goal-oriented algorithm is not guaranteed to compute only minimal unifiers.

[3] In the above example, the unifier we have described is the only minimal unifier, but the SAT-translation computes 64 local unifiers, albeit first the minimal one.

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-concept	$\top$	$\top^{\mathcal{I}} = \Delta^{\mathcal{I}}$
conjunction	$C \sqcap D$	$(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
existential restriction	$\exists r.C$	$(\exists r.C)^{\mathcal{I}} = \{x \mid \exists y:\ (x,y) \in r^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$

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

Following the assumption that it is desirable to compute only minimal unifiers rather than all (or some additional non-minimal) local ones, this paper asks the question whether the NP decision procedures for unification in $\mathcal{EL}$ presented in [8] and [9] can be appropriately modified such that the successful runs of the procedure produce exactly the minimal unifiers of the given $\mathcal{EL}$-unification problem. We show in Section 4 that the answer to this question is negative if we use a slightly more general definition of the order $\succeq$, where the subsumption test $\sigma(X) \sqsubseteq \theta(X)$ can be restricted to a subset of all variables. This restriction is justified by the fact that the user may be interested only in the substitution-images of some of the variables. In fact, the algorithms in [8] and [9] first flatten the input problem, which introduces auxiliary variables. These auxiliary variables are internal to the unification procedure and are not shown to the user.

All three $\mathcal{EL}$-unification algorithms mentioned above (the brute-force algorithm, the goal-oriented algorithm, and the one based on a reduction to SAT) actually do not directly compute local unifiers, but so-called acyclic assignments, which can be seen as compact representations of local unifiers. In Section 3 we ask what properties of the acyclic assignment make the induced unifiers small. To this purpose, we introduce a natural order on acyclic assignments and compare it with the order $\succeq$ on the induced unifiers.

2 Unification in $\mathcal{EL}$

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 using the concept constructors *top-concept* ($\top$), *conjunction* ($C \sqcap D$), and *existential restriction* ($\exists r.C$ for every $r \in N_R$).

An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a nonempty domain $\Delta^{\mathcal{I}}$ and an interpretation function $\cdot^{\mathcal{I}}$ that assigns binary relations on $\Delta^{\mathcal{I}}$ to role names and subsets of $\Delta^{\mathcal{I}}$ to concept descriptions, as shown in the semantics column of Table 1.

The concept description C *is subsumed by* the concept description D (written $C \sqsubseteq D$) iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds for all interpretations $\mathcal{I}$. We say that C *is equivalent to* D (written $C \equiv D$) iff $C \sqsubseteq D$ and $D \sqsubseteq C$, i.e., iff $C^{\mathcal{I}} = D^{\mathcal{I}}$ holds for all interpretations $\mathcal{I}$.

We will also need the notion of an acyclic TBox $\mathcal{T}$, which is a finite set of

concept definitions of the form $A \equiv C$, where A is a concept name and C a concept description, that is unambiguous and acyclic (see [4] for details). The interpretation $\mathcal{I}$ is a model of $\mathcal{T}$ iff it satisfies all concept definitions in $\mathcal{T}$, i.e., $A^{\mathcal{I}} = C^{\mathcal{I}}$ holds for all $A \equiv C$ in $\mathcal{T}$. The concept description C *is subsumed by* the concept description D w.r.t. the acyclic TBox $\mathcal{T}$ (written $C \sqsubseteq_{\mathcal{T}} D$) iff $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ holds for all models $\mathcal{I}$ of $\mathcal{T}$.

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 the top-concept $\top$.

When defining unification in $\mathcal{EL}$, we assume that the set of concept names is partitioned 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 is a mapping from N_v into the set of all $\mathcal{EL}$-concept descriptions. This mapping is extended to concept descriptions in the usual way, i.e., by replacing all occurrences of variables in the description by their σ-images. Unification tries to make concept descriptions equivalent by applying a substitution.

Definition 2.1 *An* $\mathcal{EL}$-unification problem *is of the form* $\Gamma = \{C_1 \equiv^? D_1, \ldots, C_n \equiv^? D_n\}$, *where* $C_1, D_1, \ldots C_n, D_n$ *are* $\mathcal{EL}$*-concept descriptions. The substitution* σ *is a* unifier *(or* solution*) of* Γ *iff* $\sigma(C_i) \equiv \sigma(D_i)$ *for* $i = 1, \ldots, n$. *In this case,* Γ *is called* solvable *or* unifiable.

We will sometimes use the subsumption $C \sqsubseteq^? D$ as abbreviation for the equivalence $C \sqcap D \equiv^? C$. Obviously, the substitution σ solves this subsumption iff $\sigma(C) \sqsubseteq \sigma(D)$.

Flattening

As mentioned before, the algorithms in [8] and [9] first flatten the unification problem. 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. 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. [4]

Let Γ be a unification problem. By introducing auxiliary variables, Γ can be transformed in polynomial time into a flat unification problem Γ' such that the unifiability status remains unchanged, i.e., Γ has a unifier iff Γ' has a unifier. More precisely, it can be shown that, restricted to the variables of Γ, every unifier of Γ' is also a unifier of Γ. Conversely, every unifier of Γ can be extended to a unifier of Γ' by defining appropriate images for the auxiliary variables. Thus, we may assume without loss of generality that our input $\mathcal{EL}$-unification problems are flat.

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

Local unifiers

Let Γ be a flat unification problem. The atoms of Γ are the atoms of all the concept descriptions occurring in Γ. We define

$$\mathrm{At} := \{C \mid C \text{ is an atom of } \Gamma\} \text{ and}$$
$$\mathrm{At}_{\mathsf{nv}} := \mathrm{At} \setminus N_v \quad \text{(non-variable atoms)}.$$

Every assignment S of subsets S_X of $\mathrm{At}_{\mathsf{nv}}$ to the variables X in N_v induces the following relation $>_S$ on N_v: $>_S$ is the transitive closure of

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

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

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 is a local substitution, then we call it a *local unifier* of Γ.

Theorem 2.2 ([7]) *Let Γ be a flat unification problem. If Γ has a unifier, then it also has a local unifier.*

The theorem shows that, in order to decide unifiability of Γ, it is sufficient to guess an acyclic assignment and then check whether the induced substitution is a unifier. The remaining problem is that the induced unifier may be of exponential size. However, in order to check whether a given acyclic assignment S induces a unifier of Γ, one does not need to construct the unifier σ_S explicitly. In fact, S can be turned into an acyclic TBox

$$\mathcal{T}_S := \{X \equiv \prod_{D \in S_X} D \mid X \in N_v\},$$

and it is easy to see that the following holds for arbitrary concept descriptions E, F: $\sigma_S(E) \sqsubseteq \sigma_S(F)$ iff $E \sqsubseteq_{\mathcal{T}_S} F$. Since subsumption in $\mathcal{EL}$ w.r.t. acyclic TBoxes can be decided in polynomial time [1], this obviously yields a way for checking, in polynomial time, whether σ_S solves all equations of Γ.

The original proof of Theorem 2.2 in [7] was based on the notion of a minimal unifier, though subsequent simpler proofs [8,3] no longer need this notion.

For readers familiar with unification theory [11], it should be pointed out that the order we use to define minimality of local unifiers (see below) is *not* the instantiation pre-order on substitutions. In fact, it is an easy consequence of the definition of local substitutions that they are ground (i.e., the images of variables under these substitutions do not contain variables), and thus there is no further instantiation possible.

Minimal unifiers

Given a set of variables $\mathcal{X} \subseteq N_v$, we define

$\sigma \succeq_{\mathcal{X}} \theta$ iff $\sigma(X) \sqsubseteq \theta(X)$ holds for all variables $X \in \mathcal{X}$,

$\sigma \succ_{\mathcal{X}} \theta$ iff $\sigma \succeq_{\mathcal{X}} \theta$ and $\theta \not\succeq_{\mathcal{X}} \sigma$.

We say that the unifier σ of Γ is *$\mathcal{X}$-minimal* iff there is no unifier θ of Γ such that $\sigma \succ_{\mathcal{X}} \theta$. We say that two substitutions σ, θ are *equivalent* ($\sigma \equiv \theta$) iff $\sigma(X) \equiv \theta(X)$ holds for all $X \in N_v$. Note that we have $\sigma \equiv \theta$ iff $\sigma \succeq_{N_v} \theta$ and $\theta \succeq_{N_v} \sigma$.

Lemma 2.3 ([9]) *Let Γ be a flat unification problem.*

(i) *If Γ is solvable, then it also has an N_v-minimal unifier.*

(ii) *Every N_v-minimal unifier is equivalent to a local unifier.*

The first part of the lemma is an immediate consequence of the fact [9] that $\succ_{N_v}$ is well-founded, whereas the proof of the second part in [9] is rather long and intricate. Theorem 2.2 is an immediate consequence of Lemma 2.3.

3 Minimal unifiers versus minimal assignments

As mentioned before, we are interested in computing only the N_v-minimal unifiers rather than all local unifiers of a given unification problem. All three $\mathcal{EL}$-unification algorithms mentioned in the introduction (the brute-force algorithm, the goal-oriented algorithm, and the one based on a reduction to SAT) actually compute acyclic assignments rather than directly local unifiers. Thus, one can ask what properties of the assignment make unifiers small w.r.t. $\succeq_{N_v}$. To answer this question, we define an order similar to $\succeq_{N_v}$ on acyclic assignments. Let S, T be acyclic assignments of subsets S_X, T_X of $\mathrm{At}_{\mathsf{nv}}$ to the variables X in N_v. We define

$$S \geq T \text{ iff } S_X \supseteq T_X \text{ holds for all } X \in N_v.$$

As usual, we write $S > T$ if $S \geq T$ and $S \neq T$. First, we show that smaller assignments indeed yield smaller unifiers.

Lemma 3.1 *If $S \geq T$, then $\sigma_S \succeq_{N_v} \sigma_T$.*

Proof. Obviously, $S \geq T$ implies $O_S \supseteq O_T$, and thus $>_S \supseteq >_T$. We show $\sigma_S(X) \sqsubseteq \sigma_T(X)$ for all $X \in N_v$ by induction along $>_S$.

If X is a minimal element of N_v w.r.t. $>_S$, then it is also a minimal element of N_v w.r.t. $>_T$ since $>_S \supseteq >_T$. Thus, $\sigma_S(X) = \bigsqcap_{D \in S_X} D$ and $\sigma_T(X) = \bigsqcap_{E \in T_X} E$. Consequently, $S_X \supseteq T_X$ implies that $\sigma_S(X) \sqsubseteq \sigma_T(X)$.

Assume that $\sigma_S(Y) \sqsubseteq \sigma_T(Y)$ holds for all Y such that $X >_S Y$. Since $>_S \supseteq >_T$, this implies that $\sigma_S(Z) \sqsubseteq \sigma_T(Z)$ holds for all Z such that $X >_T Z$. Since the concept constructors of $\mathcal{EL}$ are monotonic w.r.t. subsumption and $S_X \supseteq T_X$, this implies

$$\sigma_S(X) = \bigsqcap_{D \in S_X} \sigma_S(D) \sqsubseteq \bigsqcap_{D \in T_X} \sigma_S(D) \sqsubseteq \bigsqcap_{D \in T_X} \sigma_T(D) = \sigma_T(X). \qquad \square$$

As an easy consequence of this lemma we obtain that minimal unifiers are induced by minimal acyclic assignments.

Theorem 3.2 *Let Γ be a flat unification problem. Then the set*

$$\{\sigma_S \mid \sigma_S \text{ is a unifier of } \Gamma \text{ and there is no acyclic assignment } T \text{ for } \Gamma \text{ such that } \sigma_T \text{ is a unifier of } \Gamma \text{ and } S > T\}$$

contains all N_v-minimal unifiers of Γ up to equivalence.

Proof. Let θ be an N_v-minimal unifier of Γ. By Lemma 2.3, θ is (equivalent to) a local unifier, and thus there exists an acyclic assignment T such that $\theta \equiv \sigma_T$. Let S be minimal among all assignments that induce a substitution equivalent to θ, i.e., $\theta \equiv \sigma_S$ and there is no acyclic assignment T for Γ such that $\sigma_T \equiv \theta$ and $S > T$.

We claim that this implies that there is no acyclic assignment T for Γ such that σ_T is a unifier of Γ and $S > T$. Assume that such an assignment T exists. Then Lemma 3.1 implies that $\sigma_S \succeq_{N_v} \sigma_T$. Minimality of S among all assignments that induce a unifier equivalent to θ implies that $\sigma_S \not\equiv \sigma_T$, and thus $\sigma_S \succ_{N_v} \sigma_T$. This contradicts the assumption that $\theta \equiv \sigma_S$ is N_v-minimal.□

Thus, if one wants to generate *all* minimal unifiers, it is enough to generate only the minimal acyclic assignments yielding unifiers. If the converse of Lemma 3.1 were true, we could also show that these assignments yield *only* minimal unifiers. Unfortunately, the converse of Lemma 3.1 is not true, as demonstrated by the following example.

Example 3.3 Let

$$\Gamma := \{X \equiv^? \exists r.Y, X \equiv^? \exists r.Z, Y \equiv^? A, Z \equiv^? A\}.$$

Consider the acyclic assignments S, T with

$$S_X := \{\exists r.Y\},\ S_Y := \{A\},\ S_Z := \{A\};$$
$$T_X := \{\exists r.Z\},\ T_Y := \{A\},\ T_Z := \{A\}.$$

Then $\sigma_S(Y) = A = \sigma_T(Y)$, $\sigma_S(Z) = A = \sigma_T(Z)$, and $\sigma_S(X) = \exists r.A = \sigma_T(X)$, i.e., $\sigma_S = \sigma_T$ and this substitution is a unifier of Γ. In particular, this implies $\sigma_S \succeq_{N_v} \sigma_T$. However, $S \geq T$ obviously does *not* hold since $S_X \not\supseteq T_X$.

It is easy to see that S and T are minimal among the acyclic assignments generating unifiers of Γ. This shows that the same N_v-minimal unifier can be generated by different minimal assignments.

We can also use a unifier σ of Γ to define an acyclic assignment S^σ:

$$S^\sigma_X := \{D \in \mathrm{At}_{\mathsf{nv}} \mid \sigma(X) \sqsubseteq \sigma(D)\}.$$

As shown in [3], this assignment is indeed acyclic.

Surprisingly, the analog of Lemma 3.1 does not hold: going from unifiers to the induced acyclic assignments is neither monotone nor antitone.

Example 3.4 We introduce an $\mathcal{EL}$-unification problem that demonstrates that $\theta \succeq_{N_v} \sigma$ implies neither $S^\theta \geq S^\sigma$ nor $S^\sigma \geq S^\theta$, even if σ and θ are equivalent to local unifiers of the given unification problem Γ. For this purpose, consider the unification problem

$$\Gamma := \{A \sqsubseteq^? X,\ \exists r.X \sqsubseteq^? Y,\ Y \sqsubseteq^? \exists r.X'\}.$$

The following substitutions are obviously unifiers of Γ:

$$\sigma := \{X \mapsto \top, X' \mapsto \top, Y \mapsto \exists r.\top\},$$
$$\theta := \{X \mapsto A, X' \mapsto \top, Y \mapsto \exists r.\top\}.$$

and they satisfy $\theta \succeq_{N_v} \sigma$.

The non-variable atoms of Γ are A, $\exists r.X$, and $\exists r.X'$, and thus

$$S^\sigma_X = \emptyset, \quad S^\sigma_{X'} = \emptyset, \quad S^\sigma_Y = \{\exists r.X, \exists r.X'\},$$
$$S^\theta_X = \{A\}, \quad S^\theta_{X'} = \emptyset, \quad S^\theta_Y = \{\exists r.X'\}.$$

Since $S^\sigma_Y \not\subseteq S^\theta_Y$, we do *not* have $S^\theta \geq S^\sigma$; and since $S^\theta_X \not\subseteq S^\sigma_X$ we do *not* have $S^\sigma \geq S^\theta$.

Finally, note that (up to equivalence) σ and θ are local since they are the substitutions respectively induced by S^σ and S^θ, i.e., $\sigma \equiv \sigma_{S^\sigma}$ and $\theta \equiv \sigma_{S^\theta}$.

This example actually strengthens Example 3.3 in the sense that it shows that the converse of Lemma 3.1 is not even true if we assume $\sigma_S \succ_{N_v} \sigma_T$ rather than $\sigma_S \succeq_{N_v} \sigma_T$. Just take $S = S^\theta$ and $T = S^\sigma$. Indeed, we have $\sigma_{S^\theta} \succ_{N_v} \sigma_{S^\sigma}$, but $S^\theta \not\geq S^\sigma$.

A similar example can be used to show that the set

$$\{\sigma_S \mid \sigma_S \text{ is a unifier of } \Gamma \text{ and there is no acyclic assignment } T \text{ for } \Gamma \text{ such that } \sigma_T \text{ is a unifier of } \Gamma \text{ and } S > T\}$$

may in general contain unifiers that are not N_v-minimal.

Example 3.5 Let

$$\Gamma := \{A \sqsubseteq^? X,\ Y \equiv^? \exists r.X,\ \exists r.A \sqsubseteq^? Y\}.$$

The non-variable atoms of Γ are A, $\exists r.X$, and $\exists r.A$. The acyclic assignments

$$S_X = \emptyset, \quad S_Y = \{\exists r.X\},$$
$$T_X = \{A\}, \quad T_Y = \{\exists r.A\}$$

generate the unifiers $\sigma_S = \{X \mapsto \top, Y \mapsto \exists r.\top\}$ and $\sigma_T = \{X \mapsto A, Y \mapsto \exists r.A\}$ of Γ. We have $\sigma_T \succ_{N_v} \sigma_S$, and thus σ_T is not N_v-minimal. However, it is easy to see that there is no acyclic assignment $U < T$ such that σ_U is a unifier of Γ.

In fact, assume that $U < T$. If $U_Y = \emptyset$, then $\sigma_U(Y) = \top$, and thus σ_U does not solve the equivalence $Y \equiv^? \exists r.X$ independent of whether $U_X = \{A\}$ or $U_X = \emptyset$. Consequently, we must have $U_Y = \{\exists r.A\} = T_Y$, and thus $\sigma_U(Y) = \exists r.A$. However, then $U < T$ implies $U_X = \emptyset$, i.e., $\sigma_U(X) = \top$. But then σ_U again does not solve the equivalence $Y \equiv^? \exists r.X$.

This example shows that, even if we only generate minimal acyclic assignments that induce unifiers, this may yield additional local unifiers that are not N_v-minimal.

We finish this section by investigating what happens if we compose the two transformations $S \mapsto \sigma_S$ and $\sigma \mapsto S^\sigma$.

Lemma 3.6 *Let S be an assignment and σ a substitution. Then $S^{\sigma_S} \geq S$ and $\sigma \succeq_{N_v} \sigma_{S^\sigma}$. If σ is a local substitution, then we even have $\sigma \equiv \sigma_{S^\sigma}$.*

Proof. If $D \in S_X$, then $\sigma_S(D)$ is a top-level conjunct of $\sigma_S(X)$, and thus $\sigma_S(X) \sqsubseteq \sigma_S(D)$, which shows $D \in S_X^{\sigma_S}$.

We show the other inequality by induction along $>_{S^\sigma}$. If X is minimal, then no variables occur in S_X^σ, and thus $\sigma(D) = D$ for all $D \in S_X^\sigma$. This yields $\sigma_{S^\sigma}(X) = \bigsqcap_{D \in S_X^\sigma} D \sqsupseteq \sigma(X)$ since all $D \in S_X^\sigma$ satisfy $\sigma(X) \sqsubseteq \sigma(D) = D$.

Assume that for all $Y \in N_v$ with $X >_{S^\sigma} Y$ we have $\sigma_{S^\sigma}(Y) \sqsupseteq \sigma(Y)$. Consider $\sigma_{S^\sigma}(X) = \bigsqcap_{D \in S_X^\sigma} \sigma_S^\sigma(D)$. Since $D \in S_X^\sigma$ contains only variables that are smaller than X w.r.t. $>_{S^\sigma}$ and the concept constructors of $\mathcal{EL}$ are monotone w.r.t. subsumption, the induction assumption yields

$$\bigsqcap_{D \in S_X^\sigma} \sigma_S^\sigma(D) \sqsupseteq \bigsqcap_{D \in S_X^\sigma} \sigma(D).$$

Finally, since all $D \in S_X^\sigma$ satisfy $\sigma(X) \sqsubseteq \sigma(D)$, we have

$$\bigsqcap_{D \in S_X^\sigma} \sigma(D) \sqsupseteq \sigma(X).$$

Since the subsumption relation is transitive, this completes the proof that $\sigma \succeq_{N_v} \sigma_{S^\sigma}$.

Finally, assume that σ is local, i.e., there is an acyclic assignment T such that $\sigma = \sigma_T$. We must show $\sigma_{S^\sigma} \succeq_{N_v} \sigma$. Because of the first statement of the lemma, we have $S^{\sigma_T} \geq T$, and thus $\sigma_{S^\sigma} = \sigma_{S^{\sigma_T}} \succeq_{N_v} \sigma_T = \sigma$ by Lemma 3.1.□

4 The complexity of computing exactly the minimal unifiers

The three $\mathcal{EL}$-unification algorithms mentioned in the introduction (the brute-force algorithm, the goal-oriented algorithm, and the one based on a reduction to SAT) are NP-decision procedures for unifiability that additionally compute local unifiers in the following sense: each successful run of the nondeterministic algorithm generates an acyclic assignment that induces a unifier. The brute-force algorithm and the SAT-based algorithm generate all local unifiers, whereas the goal-oriented algorithm generates all N_v-minimal unifiers, but may also generate some additional, non-minimal local unifiers. In [2] we sketch a variant of the SAT reduction of [8] that generates exactly the minimal assignments that induce unifiers. It is based on a reduction into a special case of the partial MAX-SAT problem [21]. By Lemma 3.2, this procedure generates all N_v-minimal unifiers, but Example 3.5 shows that—like the goal-oriented

algorithm—it may also produce additional, non-minimal local unifiers. Unlike the other three algorithms, this is no longer an NP-algorithm.

In this section we investigate the question whether there can exist an NP-algorithm that produces exactly the minimal unifiers in the sense that the successful runs of this algorithm yield a set of acyclic assignments that induces exactly the set of minimal unifiers. For the general case of $\mathcal{X}$-minimality for an arbitrary subset $\mathcal{X}$ of N_v, we give a negative answer to this question. We also show an analogous result for the problem of computing $\mathcal{X}$-minimal assignments that induce unifiers, where $\mathcal{X}$-minimality for assignments is defined in the obvious way.

To show these negative results, we consider the following decision problem, which we call the *minimal unifier containment problem*:

Given: A flat $\mathcal{EL}$-unification problem Γ, a set $\mathcal{X} \subseteq N_v$, a concept constant $A \in N_c$, and a concept variable $X \in \mathcal{X}$.

Question: Is there an $\mathcal{X}$-minimal unifier σ of Γ such that $\sigma(X) \sqsubseteq A$?

Theorem 4.1 *The minimal unifier containment problem is Σ_2^p-complete.*

Proof. Containment in Σ_2^p is easy to see. Guess an acyclic assignment S of Γ and check (in polynomial time, using $\mathcal{T}_S$) whether it induces a unifier σ_S of Γ that satisfies $\sigma_S(X) \sqsubseteq A$. If this check succeeds, then use an NP-oracle to check whether σ_S is $\mathcal{X}$-minimal. In fact, an NP procedure for testing whether σ_S is *not* $\mathcal{X}$-minimal guesses an acyclic assignment T, and then uses subsumption tests w.r.t. $\mathcal{T}_S$ to check whether $\sigma_S \succ_{\mathcal{X}} \sigma_T$.

To show Σ_2^p-hardness, we use a reduction from the *minimal model deduction problem*:

Given: A propositional formula ϕ in conjunctive normal form and a propositional variable x.

Question: Is there a minimal model M of ϕ such that $M \models x$?

Here, minimality of propositional models is defined w.r.t. the following order on propositional valuations: $V' \geq V$ iff for all propositional variables x, $V \models x$ implies $V' \models x$. Σ_2^p-completeness of the minimal model deduction problem is an immediate consequence of Lemma 3.1 in [15].

In order to reduce the minimal model deduction problem (as specified above) to the minimal unifier containment problem, we adapt the proof of NP-hardness of $\mathcal{EL}$-matching given in [6]. Let $\phi = \phi_1 \wedge \cdots \wedge \phi_m$ be a propositional formula in conjunctive normal form and let $\{x_1, \ldots, x_n\}$ be the propositional variables of this problem. Assume without loss of generality that $x = x_1$.

For the propositional variables, we introduce the concept variables

$$\{X_1, \ldots, X_n, \overline{X}_1, \ldots, \overline{X}_n\},$$

which encode x_i and $\neg x_i$, respectively. In addition, we introduce concept variables $\{Y_1, \ldots, Y_n\}$, which are used for minimization, i.e.,

$$\mathcal{X} = \{Y_1, \ldots, Y_n\}.$$

Furthermore, we need concept constants A and B (encoding the truth values) and a role name r. The unification problem $\Gamma_{\phi,x}$ constructed from the given minimal model deduction problem consists of the equations introduced below.

First, we specify equations that ensure that A, B encode the truth values. For all $i, 1 \leq i \leq n$, we add the equation

$$\exists r.X_i \sqcap \exists r.\overline{X}_i \equiv^? \exists r.A \sqcap \exists r.B.$$

Obviously, any solution of this equation replaces

- either X_i by a concept description equivalent to A and $\overline{X}_i$ by a concept description equivalent to B (corresponding to $x_i = \mathit{true}$),
- or X_i by a concept description equivalent to B and $\overline{X}_i$ by a concept description equivalent to A (corresponding to $x_i = \mathit{false}$).

In order to encode ϕ, we introduce an equation for every conjunct ϕ_j of ϕ, where we view ϕ_j as the set of its disjuncts:

$$B \sqcap \bigsqcap_{x_i \in \phi_j} X_i \sqcap \bigsqcap_{\neg x_i \in \phi_j} \overline{X}_i \equiv^? A \sqcap B$$

For example, if $\phi_j = x_1 \vee \overline{x}_2 \vee x_3 \vee \overline{x}_4$, then the corresponding equation is $B \sqcap X_1 \sqcap \overline{X}_2 \sqcap X_3 \sqcap \overline{X}_4 \equiv^? A \sqcap B$. The above equation ensures that, among all the concept variables occurring on the left-hand side, at least one must be replaced by a concept description equivalent to A. This corresponds to the fact that, in the conjunct ϕ_j, there must be at least one literal that evaluates to *true*. Note that we need the concept name B on the left-hand side to cover the case where all the variables occurring in it are substituted with A.

It is easy to see that (up to equivalence) the unifiers of the equations introduced until now (which do not contain the variables in $\mathcal{X}$) correspond exactly to the propositional models of ϕ. Given a propositional valuation V of ϕ, we define the corresponding substitution σ_V as follows:

- if $V \models x_i$, then $\sigma_V(X_i) := A$ and $\sigma_V(\overline{X}_i) := B$;
- if $V \models \neg x_i$, then $\sigma_V(X_i) := B$ and $\sigma_V(\overline{X}_i) := A$.

According to our observations above, V is a model of ϕ iff σ_V is a unifier of the equations introduced above. In addition, any unifier of these equations is equivalent to a unifier of the form σ_M for a model M of ϕ.

It remains to express minimal models as $\mathcal{X}$-minimal unifiers. For this purpose, we add the equations

$$A \sqcap B \equiv^? B \sqcap \overline{X}_i \sqcap Y_i \tag{3}$$

for all $i, 1 \leq i \leq n$. This completes the description of the unification problem $\Gamma_{\phi,x}$.

The effect of equation (3) is the following:

- If X_i is substituted with a concept description equivalent to A (corresponding to x_i being evaluated to *true*), then $\overline{X}_i$ is substituted with a concept

description equivalent to B, and thus Y_i must be substituted by a concept description equivalent to A or $A \sqcap B$. In an $\mathcal{X}$-minimal unifier, it is thus substituted with a concept description equivalent to A.

- If X_i is substituted with a concept description equivalent to B (corresponding to x_i being evaluated to *false*), then $\overline{X}_i$ is substituted with a concept description equivalent to A, and thus Y_i can be substituted by a concept description equivalent to $\top$, A, B, or $A \sqcap B$. In an $\mathcal{X}$-minimal unifier, it is thus substituted with a concept description equivalent to $\top$.

We extend the definition of the substitution σ_V induced by a propositional valuation V by setting:

- if $\sigma_V(X_i) = A$, then $\sigma_V(Y_i) := A$;
- if $\sigma_V(X_i) = B$, then $\sigma_V(Y_i) := \top$.

We claim that the minimal models of ϕ correspond to the $\mathcal{X}$-minimal unifiers of $\Gamma_{\phi,x}$.

Let M be a minimal model of ϕ, and σ_M the corresponding unifier of $\Gamma_{\phi,x}$, as defined above. Assume that σ_M is not $\mathcal{X}$-minimal. Then there is an $\mathcal{X}$-minimal unifier θ of $\Gamma_{\phi,x}$ such that $\sigma_M \succ_{\mathcal{X}} \theta$. Define the propositional valuation U by setting $U(x_i) := \mathit{true}$ iff $\theta(X_i) \equiv A$. We claim that $\theta \equiv \sigma_U$. For $X \in \{X_1, \ldots, X_n, \overline{X}_1, \ldots, \overline{X}_n\}$, we clearly have $\theta(X) \equiv \sigma_U(X)$. For $X \in \{Y_1, \ldots, Y_n\}$, $\theta(X) \equiv \sigma_U(X)$ is a consequence of the fact that, for a $\mathcal{X}$-minimal unifier θ, $\theta(Y_i) \equiv A$ iff $\theta(X_i) \equiv A$ and $\theta(Y_i) \equiv \top$ iff $\theta(X_i) \equiv B$. Since θ is a unifier of $\Gamma_{\phi,x}$, the same is true for σ_U, and thus U is a model of ϕ. However, it is easy to see that $\sigma_M \succ_{\mathcal{X}} \theta \equiv \sigma_U$ implies that $M > U$, which contradicts minimality of M. In fact, assume that $U \models x_i$, i.e., $\sigma_U(X_i) = A$. Then $\sigma_U(Y_i) = A$, which implies $\sigma_M(Y_i) = A$ (due to $\sigma_M \succeq_{\mathcal{X}} \sigma_U$), and thus $M \models x_i$. This shows $M \geq U$. Since $\sigma_M \succ_{\mathcal{X}} \sigma_U$, there is an index i such that $\sigma_M(Y_i) \sqsubset \sigma_U(Y_i)$. This is only possible if $\sigma_M(Y_i) = A$ and $\sigma_U(Y_i) = \top$. But then $\sigma_M(X_i) = A$ and $\sigma_U(X_i) = B$, and thus $M \models x_i$ and $U \not\models x_i$. This yields $M > U$. To sum up, we have shown:

If M is a minimal model of ϕ, then σ_M is an $\mathcal{X}$-minimal unifier of $\Gamma_{\phi,x}$.

Conversely, assume that θ is a minimal unifier of $\Gamma_{\phi,x}$. As shown above, the propositional valuation U defined as $U(x_i) := \mathit{true}$ iff $\theta(X_i) \equiv A$ is such that U is a model of ϕ and $\theta \equiv \sigma_U$. We claim that U is a *minimal* model of ϕ. Assume that M is a model of ϕ such that $U > M$. First, note that $U \geq M$ implies $\sigma_U \succeq_{\mathcal{X}} \sigma_M$, i.e., $\sigma_U(Y_i) \sqsubseteq \sigma_M(Y_i)$ for all $i, 1 \leq i \leq n$. To see this, it is enough to show that $\sigma_M(Y_i) = A$ implies $\sigma_U(Y_i) = A$. However, $\sigma_M(Y_i) = A$ implies $\sigma_M(X_i) = A$, which in turn implies $M \models x_i$. But then $U \geq M$ yields $U \models x_i$, and thus $\sigma_U(X_i) = A$, which finally implies $\sigma_U(Y_i) = A$. Since $U > M$, there is an index i such that $U \models x_i$, but $M \not\models x_i$. But then $\sigma_U(Y_i) = A$ and $\sigma_M(Y_i) = \top$, and thus $\sigma_U(Y_i) \sqsubset \sigma_M(Y_i)$. This shows $\sigma_U \succ_{\mathcal{X}} \sigma_M$, which contradicts the $\mathcal{X}$-minimality of $\theta \equiv \sigma_U$. To sum up, we have shown:

If θ is an $\mathcal{X}$-minimal unifier of $\Gamma_{\phi,x}$, then there is a minimal model M of ϕ

such that $\theta \equiv \sigma_M$.

To finish the proof of the theorem, first assume that there is a minimal model M of ϕ such that $M \models x_1$. Then the $\mathcal{X}$-minimal unifier σ_M of $\Gamma_{\phi,x}$ satisfies $\sigma_M(Y_1) = A$, and thus $\sigma_M(Y_1) \sqsubseteq A$. Conversely, assume that there is an $\mathcal{X}$-minimal unifier θ of $\Gamma_{\phi,x}$ such that $\theta(Y_1) \sqsubseteq A$. Then there is a minimal model M of ϕ such that $\theta \equiv \sigma_U$. But then $\sigma_U(Y_1) \equiv \theta(Y_1) \sqsubseteq A$ yields $\sigma_U(Y_1) = A$, which implies $M \models x_1$.

To sum up, we have described a polynomial-time reduction of the minimal model deduction problem to the minimal unifier containment problem. Since the former problem is known to be Σ_2^p-hard, this shows Σ_2^p-hardness of the latter problem. □

As an immediate consequence of this theorem, we can show that there cannot be an NP-algorithm that generates exactly the minimal unifiers of the given $\mathcal{EL}$-unification problem.

Corollary 4.2 *Unless the polynomial hierarchy collapses, there cannot exist an NP-decision procedure for unifiability in* $\mathcal{EL}$ *that, given a flat* $\mathcal{EL}$*-unification problem* Γ *and a subset* $\mathcal{X}$ *of the concept variables occurring in* Γ*, not only decides unifiability of* Γ*, but additionally computes exactly the* $\mathcal{X}$*-minimal unifiers of* Γ *in the following sense:*

- *each successful run of the nondeterministic procedure generates an acyclic assignment* S *such that the induced local unifier* σ_S *is an* $\mathcal{X}$*-minimal unifier of* Γ*, and*
- *for every* $\mathcal{X}$*-minimal unifier* θ *of* Γ *there is a successful run of the nondeterministic procedure that generates an acyclic assignment* S *such that* $\sigma_S \equiv \theta$.

Proof. Assume that there exists an NP-decision procedure for unifiability in $\mathcal{EL}$ that computes exactly the $\mathcal{X}$-minimal unifiers of Γ in the sense introduced above. Then we could decide the minimal unifier containment problem within NP. In fact, the NP-procedure for deciding this problem is obtained by using the one that computes exactly the $\mathcal{X}$-minimal unifiers, but for every successful path of that procedure checks whether the generated acyclic assignment S satisfies $X \sqsubseteq_{\mathcal{T}_S} A$. This test can be performed in polynomial time, and it yields the same result as testing whether $\sigma_S(X) \sqsubseteq A$. Since the acyclic assignments generated by the original NP-procedure correspond exactly to the $\mathcal{X}$-minimal unifiers, there is a successful path of the extended NP-procedure iff there is an $\mathcal{X}$-minimal unifier θ satisfying $\theta(X) \sqsubseteq A$. Thus, this extended procedure decides the minimal unifier containment problem within NP. Obviously, membership of a Σ_2^p-complete problem in NP would imply $\Sigma_2^p = \text{NP}$. It is well-known that this would imply that the whole polynomial hierarchy collapses. □

Although in general minimal assignments need not induce minimal unifiers, this is not the case for the unification problem constructed in the proof of Theorem 4.1. For this reason, $\mathcal{X}$-minimal assignments that yield unifiers can also not be computed with an NP-procedure. Here, $\mathcal{X}$-minimality of assignments is defined with the following variant of the order on assignments introduced in

Section 3. Let S, T be acyclic assignments of subsets S_X, T_X of $\mathrm{At}_{\mathsf{nv}}$ to the variables X in N_v, and $\mathcal{X} \subseteq N_v$ a set of variables. We define

$S \geq_{\mathcal{X}} T$ iff $S_X \supseteq T_X$ holds for all variables $X \in \mathcal{X}$, and

$S >_{\mathcal{X}} T$ iff $S \geq_{\mathcal{X}} T$ and $T \not\geq_{\mathcal{X}} S$.

If we consider the unification problem Γ constructed in the proof of Theorem 4.1, then its non-variable atoms are $\exists r.X_i$, $\exists r.\overline{X}_i$, $\exists r.A$, $\exists r.B$, A, and B. However, assignments that induce unifiers can obviously only assign subsets of $\{A, B\}$ to variables. As an easy consequence, we have

$$S \geq_{\mathcal{X}} T \text{ iff } \sigma_S \succeq_{\mathcal{X}} \sigma_T \quad \text{and} \quad S >_{\mathcal{X}} T \text{ iff } \sigma_S \succ_{\mathcal{X}} \sigma_T$$

for all acyclic assignments S, T such that σ_S, σ_T are unifiers. This actually holds for all subsets $\mathcal{X}$ of the set of variables of Γ, and thus in particular for the one employed in the proof of Theorem 4.1. In addition, since unifiers can only substitute the variables with $\top$, A, B, or $A \sqcap B$, all unifiers are local.

Lemma 4.3 *Let Γ be the unification problem constructed in the proof of Theorem 4.1 and let $\mathcal{X} := \{Y_1, \ldots, Y_n\}$. Then the set*

$$\{\sigma_S \mid \sigma_S \text{ is a unifier of } \Gamma \text{ and there is no acyclic assignment } T \text{ for } \Gamma \text{ such that } \sigma_T \text{ is a unifier of } \Gamma \text{ and } S >_{\mathcal{X}} T\}$$

consists of exactly the $\mathcal{X}$-minimal unifiers of Γ up to equivalence.

Proof. First, assume that S is an acyclic assignment such that σ_S is a unifier of Γ and there is no acyclic assignment T for Γ such that σ_T is a unifier of Γ and $S >_{\mathcal{X}} T$. If σ_S is not $\mathcal{X}$-minimal, then there is a unifier τ of Γ such that $\sigma_S \succ_{\mathcal{X}} \tau$. Since all unifiers are local, there is an acyclic assignment T such that $\tau \equiv \sigma_T$. But then σ_T is a unifier of Γ, and $\sigma_S \succ_{\mathcal{X}} \tau \equiv \sigma_T$ implies $S >_{\mathcal{X}} T$, which contradicts our assumption on S.

Conversely, let us assume that σ is an $\mathcal{X}$-minimal unifier of Γ. Since all unifiers are local, there is an acyclic assignment S such that $\sigma_S \equiv \sigma$. Assume that there is an acyclic assignment T for Γ such that σ_T is a unifier of Γ and $S >_{\mathcal{X}} T$. But then $\sigma_S \succ_{\mathcal{X}} \sigma_T$, which contradicts our assumption that σ is $\mathcal{X}$-minimal. □

We say that S is an *$\mathcal{X}$-minimal assignment that induces a unifier of Γ* if σ_S is a unifier of Γ and there is no acyclic assignment T for Γ such that σ_T is a unifier of Γ and $S >_{\mathcal{X}} T$. Obviously, the lemma implies that an NP-algorithm that computes all $\mathcal{X}$-minimal assignments that induce unifiers also computes all $\mathcal{X}$-minimal unifiers in the sense defined in Corollary 4.2. Thus, Corollary 4.2 implies that such an NP-algorithm cannot exist.

Corollary 4.4 *Unless the polynomial hierarchy collapses, there cannot exist an NP-decision procedure for unifiability in $\mathcal{EL}$ that, given a flat $\mathcal{EL}$-unification problem Γ and a subset $\mathcal{X}$ of the concept variables occurring in Γ, not only decides unifiability of Γ, but additionally computes exactly the $\mathcal{X}$-minimal assignments for Γ that induces unifiers in the following sense:*

- *each successful run of the nondeterministic procedure generates an $\mathcal{X}$-minimal assignment that induces a unifier of Γ, and*
- *for every $\mathcal{X}$-minimal assignment that induces a unifier of Γ, there is a successful run of the nondeterministic procedure that generates it.*

5 Conclusion

The results of this paper indicate that it is not easy to compute all and only the minimal unifiers of a given $\mathcal{EL}$-unification problem. On the one hand, while it is sufficient to compute only minimal acyclic assignment to obtain all minimal unifiers, this restriction does not guarantee that only minimal unifiers are generated. In addition, one cannot even compute with an NP-algorithm all $\mathcal{X}$-minimal assignments for subsets $\mathcal{X}$ of the set of all variables. On the other hand, NP-procedures cannot generate exactly the $\mathcal{X}$-minimal unifiers for subsets $\mathcal{X}$ of the set of all variables. It is an open problem whether this last fact is also true if $\mathcal{X}$ is required to be the set of all variables. As mentioned in the introduction, the restriction to a subset of all variables can be justified by the fact that the user may be interested only in the substitution-images of some of the variables. In fact, the algorithms in [8] and [9] first flatten the input problem, which introduces auxiliary variables. These auxiliary variables are internal to the unification procedure and their substitution-images are not shown to the user [2]. Thus, the unifiers need not be minimized w.r.t. these auxiliary variables. However, since the auxiliary variables are introduced in very specific way, leaving them out of the minimization process can probably not produce the effects responsible for the hardness result in Theorem 4.1.

The problem considered here is reminiscent of, but not identical to, some problems considered in complexity theory that are concerned with enumerating (minimal or maximal) [5] solutions. *Maximality problems (MAXPs)* [14] are concerned with computing a maximal solution or some maximal solutions (under set inclusion) for a given decision problem. The main difference to our setting is that algorithms solving MAXPs are not required to compute all maximal solutions. Nevertheless, it is an interesting topic for future work to consider MAXP-variants of our problem (e.g., computation of one minimal unifier) and determine the exact complexity of these problems.

For problems that can potentially have an exponential number of solutions, like ours, one can also investigate their so-called *enumeration complexity* [19]. Even though computing all solutions in the worst-case takes exponential time (in the size of the input), one can distinguish between different kinds of algorithms: for polynomial-delay algorithms, the time needed to compute the next solution is polynomial in the size of the input; for incrementally polynomial algorithms, the time needed to compute the next solution is polynomial in the size of the input and the already computed solutions; and for output-polynomial algorithms, the time needed to compute all solutions is polynomial

[5] Note that, with an appropriate change of the representation of solutions, minimal solutions can be seen as maximal solutions and vice versa.

in the size of the input and output. Since existence of a minimal unifier is an NP-complete problem, there cannot be polynomial-delay algorithms or incrementally polynomial algorithms for enumerating all minimal unifiers (unless P = NP) since otherwise the first minimal unifier would be computable in polynomial time, and thus existence of a solution could be decided in polynomial time. However, there could still exist an output-polynomial algorithm, though we conjecture that this is not the case.

References

[1] Baader, F., *Terminological cycles in a description logic with existential restrictions*, in: G. Gottlob and T. Walsh, editors, *Proc. of the 18th Int. Joint Conf. on Artificial Intelligence (IJCAI 2003)* (2003), pp. 325–330.

[2] Baader, F., S. Borgwardt, J. A. Mendez and B. Morawska, *UEL: Unification solver for $\mathcal{EL}$*, in: *Proceedings of the 25th International Workshop on Description Logics (DL'12)*, CEUR Workshop Proceedings **846**, Rome, Italy, 2012.

[3] Baader, F., S. Borgwardt and B. Morawska, *Extending unification in $\mathcal{EL}$ towards general TBoxes*, in: *Proc. of the 13th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2012)* (2012), pp. 568–572, short paper.

[4] Baader, F., D. Calvanese, D. McGuinness, D. Nardi and P. F. Patel-Schneider, editors, "The Description Logic Handbook: Theory, Implementation, and Applications," Cambridge University Press, 2003.

[5] Baader, F. and S. Ghilardi, *Unification in modal and description logics*, Logic Journal of the IGPL **19** (2011), pp. 705–730.

[6] Baader, F. and R. Küsters, *Matching in description logics with existential restrictions*, in: *Proc. of the 7th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR 2000)*, 2000, pp. 261–272.

[7] Baader, F. and B. Morawska, *Unification in the description logic $\mathcal{EL}$*, in: R. Treinen, editor, *Proc. of the 20th Int. Conf. on Rewriting Techniques and Applications (RTA 2009)*, Lecture Notes in Computer Science **5595** (2009), pp. 350–364.

[8] Baader, F. and B. Morawska, *SAT encoding of unification in $\mathcal{EL}$*, in: C. Fermüller and A. Voronkov, editors, *Proc. of the 17th Int. Conf. on Logic for Programming, Artifical Intelligence, and Reasoning (LPAR-17)*, Lecture Notes in Computer Science **6397** (2010), pp. 97–111.

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

[10] Baader, F. and P. Narendran, *Unification of concept terms in description logics*, J. of Symbolic Computation **31** (2001), pp. 277–305.

[11] Baader, F. and W. Snyder, *Unification theory*, in: J. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning, Vol. I*, Elsevier Science Publishers, 2001 pp. 447–533.

[12] Babenyshev, S., V. V. Rybakov, R. Schmidt and D. Tishkovsky, *A tableau method for checking rule admissibility in S4*, in: *Proc. of the 6th Workshop on Methods for Modalities (M4M-6)*, Copenhagen, 2009.

[13] Brandt, S., *Polynomial time reasoning in a description logic with existential restrictions, GCI axioms, and—what else?*, in: R. L. de Mántaras and L. Saitta, editors, *Proc. of the 16th Eur. Conf. on Artificial Intelligence (ECAI 2004)*, 2004, pp. 298–302.

[14] Chen, Z.-Z. and S. Toda, *The complexity of selecting maximal solutions*, Inf. Comput. **119** (1995), pp. 231–239.

[15] Eiter, T. and G. Gottlob, *Propositional circumscription and extended closed world reasoning are Π_2^P-complete*, Theoretical Computer Science **114** (1993), pp. 231–245.

[16] Ghilardi, S., *Unification through projectivity*, J. of Logic and Computation **7** (1997), pp. 733–752.

[17] Ghilardi, S., *Unification in intuitionistic logic*, J. of Symbolic Logic **64** (1999), pp. 859–880.
[18] Iemhoff, R. and G. Metcalfe, *Proof theory for admissible rules*, Ann. Pure Appl. Logic **159** (2009), pp. 171–186.
[19] Johnson, D. S., C. H. Papadimitriou and M. Yannakakis, *On generating all maximal independent sets*, Inf. Process. Lett. **27** (1988), pp. 119–123.
[20] Kracht, M., *Modal consequence relations*, in: P. Blackburn, J. van Benthem and F. Wolter, editors, *The Handbook of Modal Logic*, Elsevier, 2006 pp. 491–545.
[21] Li, C. M. and F. Manyà, *MaxSAT, hard and soft constraints*, in: *Handbook of Satisfiability*, IOS Press, 2009 pp. 613–631.
[22] Rybakov, V. V., "Admissibility of logical inference rules," Studies in Logic and the Foundations of Mathematics **136**, North-Holland Publishing Co., Amsterdam, 1997.
[23] Rybakov, V. V., *Multi-modal and temporal logics with universal formula - reduction of admissibility to validity and unification*, J. of Logic and Computation **18** (2008), pp. 509–519.
[24] Wolter, F. and M. Zakharyaschev, *Undecidability of the unification and admissibility problems for modal and description logics*, ACM Trans. Comput. Log. **9** (2008).