32nd International Workshop on Unification (UNIF 2018), Informal Proceedings
Oxford, 7th July, 2018

ACUI Unification modulo Ground Theories

Franz Baader[1], Pavlos Marantidis[1]*, and Antoine Mottet[2]*
firstname.lastname@tu-dresden.de

[1] Theoretical Computer Science, TU Dresden, Germany
[2] Institute for Algebra, TU Dresden, Germany

Abstract

It is well-known that the unification problem for a binary associative-commutative-idempotent function symbol with a unit (*ACUI*-unification) is polynomial for unification with constants and NP-complete for general unification. We prove that the same is true if we add a finite set of ground identities. To be more precise, we first show that not only unification with constants, but also unification with linear constant restrictions is in P for any extension of *ACUI* with a finite set of ground identities. Using well-known combination results for unification algorithms, this then yields an NP-upper bound for general unification modulo such a theory. The matching lower bound can be shown as in the case without ground identities.

1 Introduction

As shown by Kapur and Narendran [7], general *ACUI*-unification is NP-complete, while elementary unification and unification with free constants is in P. In particular, this also implies that the word problem in *ACUI* is decidable in polynomial time. Marché proved in [8] that the word problem remains decidable if *ACUI* is extended with a finite set of ground identities, but no complexity bounds are given. This result actually holds for a signature possibly containing several *ACUI* symbols and free function symbols.

We are interested in whether decidability of unification for *ACUI* is also stable under adding ground identities. In this paper, we answer this question affirmatively for the case of a single *ACUI* symbol. The ground identities may contain this symbol and additional constant symbols, but no additional function symbols of arity greater than 0. In this setting, we can actually prove that not only decidability of unification, but also the complexity results transfer. To this purpose, we show that unification with linear constant restrictions, a notion that generalizes unification with free constants, is decidable in P. Then, using known combination results by Baader and Schulz [4], we can conclude that general unification is decidable in NP.

Our interest in *ACUI*-unification modulo an additional ground theory stems from the fact that the theory *ACUI* is a common subtheory of the equational theories corresponding to the Description Logics $\mathcal{FL}_0$ and $\mathcal{EL}$: for $\mathcal{FL}_0$, *ACUI* is extended with unary function symbols that behave like homomorphisms and for $\mathcal{EL}$ the additional unary function symbols behave like

*Supported by DFG Graduiertenkolleg 1763 (QuantLA)

monotone operators. Unification in $\mathcal{FL}_0$ is known to be ExpTime-complete [3] and NP-complete in $\mathcal{EL}$ [2]. However, it is not known how to extend these decidability results to unification in the presence of so-called general TBoxes, though for $\mathcal{EL}$ there are some positive results for a restricted form of TBoxes [1]. Since, from an equational theory point of view, general TBoxes correspond to finite sets of ground identities, we are interested in equational theories for which decidability of unification is stable under adding finite sets of ground identities. We will show in this paper that *ACUI* is such a theory.

2 *ACUIG*-unification with linear constant restriction

We assume in the following that the reader is familiar with basic notions of unification theory, and in particular the difference between unification with constants, unification with linear constant restriction, and general unification. Detailed definitions and a discussion of this difference can be found in [5].

Let $\Sigma = \{+, \mathbf{0}\}$ for a binary function symbol $+$ and a constant symbol $\mathbf{0}$. We denote the equational theory that states that $+$ is an associative, commutative, and idempotent symbol with unit $\mathbf{0}$ by *ACUI*. Furthermore, let F be a countably infinite set of constants and V a countably infinite set of variables. The set of terms built from Σ, F and V is denoted by $T_\Sigma(V, F)$, and the set of ground terms, i.e., terms that do not contain variables, by $T_\Sigma(F)$. If G is a finite set of ground identities using terms in $T_\Sigma(F)$, then we denote the equational theory $ACUI \cup G$ with *ACUIG*. The constants from F not occurring in G are called *free constants*. For example, if $a, b, c \in F$ and $x, y \in V$, then $x + y$ and $a + x$ belong to $T_\Sigma(V, F)$, and $a + b + b$ and $b + a + a$ are elements of $T_\Sigma(F)$. The latter two terms are actually equivalent modulo *ACUI*. If G contains the identity $a + b = c$, then these terms are also equivalent to $a + b + c$. In fact, $b + a + a =_{ACUI} a + b + b =_{ACUI} a + b =_{ACUI} a + b + a + b =_{ACUIG} a + b + c$.

A substitution is a mapping $\sigma : V \to T_\Sigma(V, F)$, which is the identity for all but finitely many variables. It can be homomorphically extended to a mapping from $T_\Sigma(V, F)$ to $T_\Sigma(V, F)$ in the obvious way.

Definition 1 (*ACUIG*-unification problem with linear constant restriction)**.**

Input: *A finite system $\Gamma = \{s_1 =^? t_1, \ldots, s_k =^? t_k\}$ of equations between terms in $T_\Sigma(V, F)$, a finite set of ground identities $G = \{g_1 = h_1, \ldots, g_\ell = h_\ell\}$ between terms of $T_\Sigma(F)$, and a linear order $\prec$ on $X \cup D$, where $X \subseteq V$ is the set of variables occurring in Γ and $D \subseteq F$ is the set of* free *constants occurring in Γ.*

Question: *Is there a substitution σ such that $\sigma(s_i) =_{ACUIG} \sigma(t_i)$ for every $i = 1, \ldots, k$ and for every $x \in X$ and $d \in D$ we have that d does not occur in $\sigma(x)$ if $x \prec d$. Such a substitution is called an ACUIG-unifier of Γ w.r.t. $\prec$.*

Now, let Γ be an *ACUIG*-unification problem with linear constant restriction $\prec$, and $C = \{a_1, \ldots, a_n\}$ be the finite set of constants occurring in Γ and in G, and $X = \{x_1, \ldots, x_m\}$ the finite set of variables occurring in Γ. In order to check whether Γ has a unifier w.r.t. $\prec$ it is sufficient to consider substitutions that are the identity on $V \setminus X$ and replace every $x \in X$ by a term in $T_\Sigma(C)$, i.e., a ground term containing (in addition to $\mathbf{0}$) only constants from C. In fact, any *ACUIG*-unifier of Γ w.r.t. $\prec$ can be turned into one satisfying this property by replacing variables and constants in $F \setminus C$ with $\mathbf{0}$. If we apply such a substitution to the terms s_i, t_i occurring in Γ, then we obtain terms in $T_\Sigma(C)$.

Modulo *ACUI*, terms in $T_\Sigma(C)$ can be represented as subsets of C. These sets just consist of the constants occurring in the terms. Two ground terms are equivalent modulo *ACUI* iff

the corresponding sets are equal. For this reason, we often assume in the following that ground terms are represented as such sets. However, in the presence of a set of ground identities G, different sets may represent terms that are equivalent modulo $ACUIG$. In our above example, the terms $a+b+b$ and $b+a+a$ are both represented by the set $\{a,b\}$, whereas $a+b+c$ is represented by $\{a,b,c\}$. Intuitively, the identity $a+b=c$ can be used to add c to the set $\{a,b\}$.

We will now show how we can decide whether two sets represent terms that are equivalent modulo $ACUIG$. For this purpose we saturate the sets by using the identities in G to add constants to them, as we have done with c in our example. For each identity $g_i = h_i$ in G, let $G_i, H_i \subseteq C$ be the sets corresponding to g_i, h_i.

Given a set $A \subseteq C$, its *saturation* is obtained by iteratively applying the identities of G as follows: begin with setting $A^* := A$; as long as there is an identity $g_i = h_i$ in G such that $G_i \subseteq A^*$ and $H_i \not\subseteq A^*$ (or $H_i \subseteq A^*$ and $G_i \not\subseteq A^*$), extend A^* by setting $A^* := A^* \cup H_i$ (respectively, by setting $A^* := A^* \cup G_i$). This saturation process terminates after a number of iterations that is bounded by the cardinality of G. In fact, once an identity $g_i = h_i$ was applied in the saturation process, it is no longer applicable since the set A^* then contains $G_i \cup H_i$. It is also easy to see that the result of the saturation does not depend on the order in which rules are applied. Thus, each set A has a unique saturation A^*, which can be computed in polynomial time.

As an example, consider the set of ground identities

$$G = \{a+b+c=d, \quad b+c+e=f\}$$

and the term $s = a+f$. The saturation process for s starts with setting $A_s^* := A_s = \{a,f\}$. For the second identity, we have that $A_f = \{f\} \subseteq A_s^*$, but $A_{b+c+e} = \{b,c,e\} \not\subseteq A_s^*$. Hence, we can extend A_s^* to $A_s^* := A_s^* \cup A_{b+c+e} = \{a,b,c,e,f\}$. Now, for the first identity, we have that $A_{a+b+c} = \{a,b,c\} \subseteq A_s^*$, but $A_d = \{d\} \not\subseteq A_s^*$, and thus we obtain $A_s^* := A_s^* \cup A_d = \{a,b,c,d,e,f\}$. This is the final saturated set since it cannot be further extended using the identities in G.

The following lemma is an easy consequence of the definition of saturation.

Lemma 2. *Let $A, B \subseteq C$. Then the following holds:*

$$A \subseteq A^*, \quad A^{**} = A^*, \quad A \subseteq B \Rightarrow A^* \subseteq B^*, \quad A^* \cup B^* \subseteq (A \cup B)^*.$$

Proposition 3. *Let $A_s, A_t \subseteq C$ be sets respectively representing the terms $s, t \in T_\Sigma(C)$. Then $s =_{ACUIG} t$ iff $A_s^* = A_t^*$. In particular this implies that the word problem for $ACUIG$ is decidable in polynomial time.*

Proof. Decidability in polynomial time obviously follows from the equivalence in the first statement since the saturation A^* of a set $A \subseteq C$ can be computed in polynomial time.

To show the equivalence, first assume that $A_s^* = A_t^*$. To conclude from this that $s =_{ACUIG} t$, it is sufficient to show that saturation steps correspond to rewrite steps in $ACUIG$. Thus, assume that $l \in T_\Sigma(C)$ has the corresponding set A_l, and that $g_i = h_i$ is an identity in G such that $G_i \subseteq A_l$. Then l is of the form $l = g_i + l'$. We now have $l = g_i + l' =_{ACUIG} g_i + g_i + l' =_{ACUIG} h_i + l$, and the set corresponding to the term $h_i + l$ is $A_l \cup H_i$.

Second, assume that $A_s^* \neq A_t^*$. To show that this implies $s \neq_{ACUIG} t$, we construct a model $\mathcal{A}$ of $ACUIG$ in which this identity does not hold. As interpretation domain, we use all saturated sets, i.e., $\Delta := \{A^* \mid A \subseteq C\}$. The binary symbol $+$ is interpreted as union followed by saturation, i.e., $A^* + B^* := (A^* \cup B^*)^*$, $\mathbf{0}$ as $\emptyset^*$, and $c \in C$ as $\{c\}^*$. Given a term $u \in T_\Sigma(C)$

with corresponding set A_u, its interpretation in this algebra is A_u^*. This can easily be shown by induction on the structure of u, where the induction step uses the fact that

$$(A^* \cup B^*)^* = (A \cup B)^*, \tag{1}$$

which is an easy consequence of Lemma 2. Thus, $A_s^* \neq A_t^*$ implies that the terms s, t have different interpretations in $\mathcal{A}$. To show $s \neq_{ACUIG} t$, it is thus sufficient to show that $\mathcal{A}$ satisfies all identities of $ACUIG$. For the identities in $ACUI$ this is an easy consequence of (1) and the fact that set union is associative, commutative and idempotent and has $\emptyset$ as unit. Now consider an identity $g_i = h_i \in G$. When saturating the corresponding sets G_i and H_i, one can in a first step go both from G_i and from H_i to $G_i \cup H_i$ (unless this step is void due to an inclusion). Saturating further, one thus obtains identical saturated sets, which shows that g_i and h_i are interpreted in $\mathcal{A}$ by the same saturated set. □

Continuing our example, recall that the term $s = a + f$ has the saturated set $A_s^* = \{a, b, c, d, e, f\}$. It is easy to see that for $t = b + d + e$ saturation produces the sequence of

$$A_t = \{b, d, e\} \to \{a, b, c, d, e\} \to \{a, b, c, d, e, f\} = A_t^*,$$

where in the first step the identity $d = a + b + c$ is applied, and in the second the identity $b + c + e = f$. Thus, we have $A_s^* = A_t^*$, which shows that $s = a + f =_{ACUIG} b + d + e = t$.

Next, we introduce an algorithm that solves $ACUIG$-unification with linear constant restriction in polynomial time. Intuitively, it starts with a maximal substitution that respects the linear order $\prec$. Next, whenever an equation is not satisfied, that is, when an element appears on one side but not on the other, we trim the substitution, so that it no longer introduces this violation. Upon termination, the algorithm provides a solution if one exists, or outputs **Fail** otherwise. By a slight abuse of notation we assume that the substitutions σ considered in the algorithm actually map to sets of constants rather than ground terms. In addition, for a term $t \in T_\Sigma(X, C)$ we use $\sigma(t)$ to denote also the set corresponding to the term $\sigma(t)$.

Algorithm 1: Computation of unifier

Input: An $ACUIG$-unification problem with linear constant restriction $\prec$, as introduced in Definition 1.
Output: A unifier $\sigma : X \to 2^C$ or **Fail**.
Set $\sigma(x) := \{c \in C \mid c \prec x$ or c occurs in $G\}$ for all $x \in X$
while *some equation* $s =^? t$ *in* Γ *is not satisfied by* σ **do**
  if *there is a variable* x *in* s *such that* $\sigma(x) \not\subseteq \sigma(t)^*$ *or* y *in* t *such that* $\sigma(y) \not\subseteq \sigma(s)^*$ **then**
    Set $\sigma(x) := \sigma(x) \cap \sigma(t)^*$ for all variables x in s
    Set $\sigma(y) := \sigma(y) \cap \sigma(s)^*$ for all variables y in t
  else
    return Fail
  end
end
return σ

Before proving correctness of this algorithm, we give an example that illustrates how it tests for the existence of an $ACUIG$-unifier w.r.t. a linear constant restriction. Consider the system of equations

$$\Gamma = \{g + x_2 =^? a + x_1,\ b + x_1 =^? c + f + g,\ c + x_2 =^? a + c + e\},$$

the set of ground identities

$$G = \{a + b + c = d,\ b + c + e = f\}$$

considered in our previous examples, and the linear order

$$x_2 \prec g \prec x_1.$$

Note that g is the only *free* constant occurring in Γ, and thus it is the only constant occurring in this linear constant restriction. Also note that without $g \prec x_1$, the second equation of Γ would not be solvable. In addition, without G this second equation would not be solvable either: b belongs to the left-hand side, but could never belong to the right-hand side without additional ground identities.

The algorithm begins by setting

$$\sigma(x_1) := \{a, b, c, d, e, f, g\} \quad \text{and} \quad \sigma(x_2) := \{a, b, c, d, e, f\}.$$

Next, the algorithm enters the while loop and picks in each iteration an equation that is not satisfied:

- The second equation is not satisfied by σ. In fact, we have that $\sigma(c+f+g)^* = \{b, c, e, f, g\}$, and hence $\sigma(x_1) \not\subseteq \sigma(c+f+g)^*$. The algorithm proceeds to set $\sigma(x_1) := \{a, b, c, d, e, f, g\} \cap \{b, c, e, f, g\} = \{b, c, e, f, g\}$.
- The third equation is not satisfied by σ. We have that $\sigma(a+c+e)^* = \{a, c, e\}$, and hence $\sigma(x_2) \not\subseteq \sigma(a+c+e)^*$. The algorithm proceeds to set $\sigma(x_2) := \{a, b, c, d, e, f\} \cap \{a, c, e\} = \{a, c, e\}$.
- The first equation is not satisfied by σ. We have that $\sigma(x_1) = \{b, c, e, f, g\} \not\subseteq \sigma(g+x_2)^* = \{a, c, e, g\}$. The algorithm proceeds to set $\sigma(x_1) := \{b, c, e, f, g\} \cap \{a, c, e, g\} = \{c, e, g\}$.

The algorithm then terminates since all equations are satisfied, and yields the substitution $\sigma = \{x_1 \mapsto c + e + g, x_2 \mapsto a + c + e\}$ as output.

Proposition 4. *Algorithm 1 terminates in polynomial time. If Γ has a unifier, then it provides a unifier as output, and otherwise it fails.*

Proof. Termination in polynomial time is an easy consequence of the fact that in each iteration of the while-loop, at least one constant is removed from the image of a variable, or the loop is exited.

Since the algorithm only returns a substitution if the while-loop is exited regularly, this substitution satisfies all the equations of Γ. It satisfies the linear constant restriction due to the fact that the original substitution satisfies it and that constants are only removed from, but never added to, the image of variables during the run of the algorithm. Consequently, if the algorithm returns a substitution, then this substitution is a unifier of Γ w.r.t. $\prec$. This shows that the algorithm must return **Fail** in case Γ has no unifier w.r.t. $\prec$.

To prove the completeness of the algorithm, assume that $\widehat{\sigma}$ is a unifier of Γ, and that the algorithm terminates during the rth iteration of the while-loop. Let $\sigma^{(0)}$ be the substitution σ before the first iteration of the while-loop. For $i \in \{1, \ldots, r-1\}$, let $\sigma^{(i)}$ be the substitution obtained at the end of the ith iteration of the while-loop.

We extend $\subseteq$ to substitutions in a natural way, by using pointwise comparison. We prove by induction on i that $\widehat{\sigma} \subseteq \sigma^{(i)}$, for all $i \in \{0, \ldots, r-1\}$. Since $\widehat{\sigma}$ satisfies the linear constant

restriction, we have $\widehat{\sigma} \subseteq \sigma^{(0)}$. Let now $i \in \{0, \ldots, r-2\}$, and assume that we already know that $\widehat{\sigma} \subseteq \sigma^{(i)}$. We must prove $\widehat{\sigma} \subseteq \sigma^{(i+1)}$.

Since the algorithm does not exit the while-loop at this stage, there is an equation $s =^? t$ in Γ that is not satisfied by $\sigma^{(i)}$. In addition, since the algorithm does not fail at iteration i, there exists a variable x in s such that $\sigma(x) \not\subseteq \sigma(t)^*$ or y in t such that $\sigma(y) \not\subseteq \sigma(s)^*$. Clearly, for every $x \in X$ that does not appear in this equation, we have $\widehat{\sigma}(x) \subseteq \sigma^{(i)}(x) = \sigma^{(i+1)}(x)$. Let now x be a variable occurring in s (variables in t can be treated analogously). To prove that $\widehat{\sigma}(x) \subseteq \sigma^{(i+1)}(x)$, it suffices to prove that $\widehat{\sigma}(x) \subseteq \sigma^{(i)}(x)$ and that $\widehat{\sigma}(x) \subseteq \sigma^{(i)}(t)^*$. The first statement is true by the induction hypothesis. Now, we have

$$\widehat{\sigma}(x) \overset{(1)}{\subseteq} \widehat{\sigma}(s) \overset{(2)}{\subseteq} \widehat{\sigma}(s)^* \overset{(3)}{=} \widehat{\sigma}(t)^* \overset{(4)}{\subseteq} \sigma^{(i)}(t)^*,$$

where (1) holds because x occurs in s, (2) by Lemma 2, (3) because $\widehat{\sigma}$ is a unifier of Γ, and (4) by Lemma 2 since $\widehat{\sigma} \subseteq \sigma^{(i)}$. This finishes the induction proof.

Therefore, we now know that $\widehat{\sigma} \subseteq \sigma^{(r-1)}$. There are two possible reasons for the algorithm terminating in the rth iteration. Either the while-loop is exited regularly or the algorithm returns **Fail**. In the first case, $\sigma^{(r-1)}$ is a unifier and the algorithm returns this substitution.

It remains to show that the second case cannot occur. In this case, we have $\sigma^{(r-1)}(s)^* \neq \sigma^{(r-1)}(t)^*$ for some equation $s =^? t$ in Γ, but $\sigma^{(r-1)}(x) \subseteq \sigma^{(r-1)}(t)^*$ for all variables x in s and $\sigma^{(r-1)}(y) \subseteq \sigma^{(r-1)}(s)^*$ for all variables y in t. This can only be the case if there is a constant $c \in C$ such that c occurs in s, but $c \notin \sigma^{(r-1)}(t)^*$; or c occurs in t, but $c \notin \sigma^{(r-1)}(s)^*$. We show that this is impossible.

Thus assume that c occurs in s (the case where c occurs in t can be treated symmetrically). We have

$$c \overset{(1)}{\in} \widehat{\sigma}(s) \overset{(2)}{\subseteq} \widehat{\sigma}(s)^* \overset{(3)}{=} \widehat{\sigma}(t)^* \overset{(4)}{\subseteq} \sigma^{(r-1)}(t)^*,$$

where (1) holds since c occurs in s, (2) by Lemma 2, (3) since $\widehat{\sigma}$ is a unifier of Γ, and (4) by Lemma 2 since $\widehat{\sigma} \subseteq \sigma^{(r-1)}$. □

The following theorem is an immediate consequence of Proposition 4.

Theorem 5. *Let ACUI be the equational theory that states that the binary function symbol $+$ is associative, commutative, and idempotent, and has the constant symbol $\mathbf{0}$ as unit, and let ACUIG be an extension of ACUI by finitely many ground identities built using $+, \mathbf{0}$, and additional constant symbols. Then the ACUIG-unification problem with linear constant restrictions is decidable in polynomial time.*

3 General *ACUIG*-unification

General ACUIG-unification problems differ from the ones we have considered until now in that the terms used in Γ may contain "free" function symbols, i.e., function symbols not occurring in the identities of *ACUIG*. For example, $\{f(x+a, a+b) =^? f(b+y, x)\}$ is such a general *ACUIG*-unification problem since it contains the additional function symbol f that does not occur in the identities of *ACUIG*.

The following was proved by Baader and Schulz [4] and provides an upper bound for general *ACUIG*-unification.

Theorem 6 ([4])**.** *If solvability of E_i-unification problems with linear constant restrictions is decidable in NP for $i = 1, 2$, then unifiability in the combined theory $E_1 \cup E_2$ is also decidable in NP.*

In particular, this implies that, if E-unification with linear constant restriction is decidable in NP, the same is true for general unification, by choosing as the second theory the empty theory.

Theorem 7. *For every finite set of ground identities G, general $ACUIG$-unification is NP-complete.*

Proof. Membership in NP is an immediate consequence of Theorem 5 together with Theorem 6.

NP-hardness can be shown by the same reduction from the set-matching problem as used in [6] to show that general ACI-unification is NP-hard. To be more precise, this reduction yields ACI-unification problems of the form

$$\Gamma = \{g(s_1) + \ldots + g(s_m) \stackrel{?}{=} g(t_1) + \ldots + g(t_m)\},$$

where $+$ is an associative, commutative, and idempotent function symbol, g is a unary free function symbol and the terms $s_1, \ldots, s_m, t_1, \ldots, t_n$ contain only free function symbols and variables. The presence of a unit and of ground identities in $ACUIG$ do not change solvability of such problems compared to ACI since

- in the top-level sum the additional identities cannot be used due to the fact that all terms on this level start with the free function symbol g;
- while the variables occurring in the terms $g(s_1), \ldots, g(s_m), g(t_1), \ldots, g(t_n)$ may be replaced by terms containing $+$ and constant symbols from G, these "alien" subterms can be abstracted away by free constants.

This shows that such a problem Γ is solvable modulo ACI iff it is solvable modulo $ACUIG$, which completes our proof of NP-hardness of general $ACUIG$-unification. □

4 Conclusion

We have shown that $ACUIG$-unification with linear constant restrictions is decidable in P, and general $ACUIG$-unification is NP-complete. Note, however, that according to our definition of $ACUIG$ this result holds for a single $ACUI$-symbol $+$ and ground identities G built using only $+$ and free constant symbols. Due to the combination results of Baader and Schulz [4], we can also deal with several $ACUI$-symbols $+_1, \ldots, +_n$ and sets of ground identities $G_1, \ldots, G_n$, where the identities in G_i are built using only $+_i$ and free constant symbols not occurring in any of the other sets G_j $(i \neq j)$. The combination results show that unification in the union of such theories can be decided in NP. However, this result cannot deal with situations where the ground identities share constants, or contain several $ACUI$-symbols, or contain free function symbols. It is an open problem whether unification in such "mixed" theories remains decidable. It is only known from Marché's results [8] that the word problem is decidable in this setting, and it would be interesting to see whether the same is true for unification.

Motivated by the applications in Description Logics mentioned in the introduction, we also intend to investigate what effect adding ground identities to extensions of $ACUI$ has on the unification problem.

References

[1] F. Baader, 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)*, pages 568–572. AAAI Press/The MIT Press, 2012.

[2] F. Baader 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)*, volume 5595 of *Lecture Notes in Computer Science*, pages 350–364. Springer, 2009.

[3] F. Baader and P. Narendran. Unification of concept terms in description logics. *J. of Symbolic Computation*, 31(3):277–305, 2001.

[4] F. Baader and K. U. Schulz. General A- and AX-unification via optimized combination procedures. In H. Abdulrab and J.-P. Pécuchet, editors, *Word Equations and Related Topics*, volume 677 of *Lecture Notes in Computer Science*, pages 23–42. Springer, 1993.

[5] F. Baader and W. Snyder. Unification theory. In J. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume I, pages 447–533. Elsevier Science Publishers, 2001.

[6] D. Kapur and P. Narendran. NP-completeness of the set unification and matching problems. In J. H. Siekmann, editor, *Proceedings of the 8th International Conference on Automated Deduction*, volume 230 of *Lecture Notes in Computer Science*, pages 489–495, Oxford, UK, 1986. Springer.

[7] D. Kapur and P. Narendran. Complexity of unification problems with associative-commutative operators. *Journal of Automated Reasoning*, 9(2):261–288, Oct 1992.

[8] C. Marché. Normalized rewriting: an alternative to rewriting modulo a set of equations. *Journal of Symbolic Computation*, 21(3):253 – 288, 1996.