AC-complete Unification and its Application to Theorem Proving*

Alexandre Boudet, Evelyne Contejean and Claude Marché

LRI, CNRS URA 410
Bât. 490, Université Paris-Sud, Centre d'Orsay
91405 Orsay Cedex, France
Phone:+33 1 69 41 69 05, Fax: +33 1 69 41 65 86
Email: {boudet,contejea,marche}@lri.fr

Abstract. The inefficiency of AC-completion is mainly due to the doubly exponential number of AC-unifiers and thereby of critical pairs generated. We present AC-complete E-unification, a new technique whose goal is to reduce the number of AC-critical pairs inferred by performing unification in a extension E of AC (*e.g.* ACU, Abelian groups, Boolean rings, ...) in the process of normalized completion [21]. The idea is to represent complete sets of AC-unifiers by (smaller) sets of E-unifiers. Not only the theories E used for unification have exponentially fewer most general unifiers than AC, but one can remove from a complete set of E-unifiers those solutions which have no E-instance which is an AC-unifier.
First, we define AC-complete E-unification and describe its fundamental properties. We show how AC-complete E-unification can be done in the elementary case, and how the known combination techniques for unification algorithms can be reused for our purposes. Finally, we give some evidence of the kind of speedup that can be obtained by presenting some experiments with the C*i*ME theorem prover.

Introduction

The completion of rewriting systems modulo an equational theory T was introduced by Jouannaud and Kirchner [13, 3]. This allows to cope with permutative theories (the most popular being associativity and commutativity, denoted hereafter as AC), avoiding the non-termination of rules like $x + y \longrightarrow y + x$. In the case of AC-completion, one will make no distinction between the terms $a + b$ and $b + a$. The price to pay for identifying the terms that are equal modulo an equational theory T, is that T has to be taken into account in the deduction mechanism: standard unification is replaced by T-unification. The main problem with AC-unification is that it may compute doubly exponentially many most general unifiers. A solution to this problem is to consider rewrite rules *constrained*

* This research was supported in part by the EWG CCL, the HCM Network CONSOLE, and the "GDR de programmation du CNRS".

by AC-unification problems, such a constrained rule representing the set of its instances by the AC-unifiers of the constraint. This is the *basic strategy* [22, 4].

On the other hand, Marché has introduced T-normalized completion [21] where the equations are normalized eagerly with a convergent term rewriting system equivalent to T. The theory T may range from AC to Boolean rings. T-normalized completion originated in the solution to the problem of AC-completion with unit [5, 14] (denoted hereafter as ACU). It also gave an elegant solution to the problem raised by Loos and Buchberger [20, 10] of designing a uniform procedure which can specialize to Knuth-Bendix completion [18], AC-completion [23, 19] and to Gröbner basis computation [9]. Unfortunately, T-normalized completion is by essence incompatible with the basic strategy since one needs to apply the unifiers in order to perform T-normalization.

Let us go into details about the motivation of this paper: during the AC-completion process, new equations are computed using AC-unification, in order to reduce the so-called *AC critical peaks*:

$$s \xleftarrow[l_1 \to r_1]{\sigma,p} t \xleftrightarrow[\mathrm{AC}]{pq} t' \xrightarrow[l_2 \to r_2]{\sigma,pq} u$$

Indeed, with such a peak we have $l_1|_p\sigma =_{\mathrm{AC}} l_2\sigma$ hence there is a θ in the complete set of AC-unifiers of $l_1|_p$ and l_2 such that $\sigma =_{\mathrm{AC}} \theta\sigma'$. The peak is then transformed into the equational proof:

$$s \xleftrightarrow[AC]{p} s' \xleftrightarrow[l_1|_p\theta = l_2\theta]{\sigma',p} u' \xleftrightarrow[AC]{p} u$$

where the AC steps come from $\sigma =_{\mathrm{AC}} \theta\sigma'$. In T-normalized completion, it has been shown correct to use E-unification instead of AC-unification, for any E ranging from AC to T, since E is valid in the theory to be completed. One can then pick a θ' in the complete set of E-unifiers of $l_1|_p$ and l_2 such that $\sigma =_E \theta'\sigma'$, and transform the peak into:

$$s \xleftrightarrow[E]{p} s' \xleftrightarrow[l_1|_p\theta' = l_2\theta']{\sigma',p} u' \xleftrightarrow[E]{p} u$$

The choice of E between AC and T is very important because one can choose E such that E-unification behaves better than AC-unification:

- E must be a sub-theory of T so as to ensure the soundness of the deductions;
- E must be rich enough to avoid the complexity of AC unification: for instance ACU is preferable to AC;
- E must be decidable and finitary wrt the unification.

In practice, we are interested in ACU-unification which is simply exponential whereas AC-unification is doubly exponential, and in Abelian Group and Boolean Ring unification which are unitary (in the elementary case) [17].

In this article, the first idea is to pursue this idea more deeply: complete sets of E-unifiers are in fact used as a compact representation of complete sets of AC-unifiers, hence for completion it is still complete to compute only E-unifiers *which have at least one AC-unifier as an instance*: this is what we call *AC-complete* unification. The second idea is that an algorithm for computing such AC-complete sets of E-unifiers should be much simpler than E-unification algorithm: since we are only interested in E-unifiers which have instances which are AC unifiers, the rules *Clash* ($f(t_1, \ldots, t_n) = g(u_1, \ldots, u_k)$ has no solutions if $f \neq g$) and *Occur-check* ($x = C[x]$ has no solution for any variable x and non-empty context C) should be valid. For example, when computing AC-complete set of unifiers modulo ACU$(+, 0)$ of the equation $x + a = a$, we find immediately no solution, whereas there is in fact an ACU-unifier ($\{x \mapsto 0\}$) but which does not have any AC-solution as instance (there is no AC-solution!). Unfortunately, this idea is not true in general, as shown by example 4. In fact, *Clash* and *Occur-check* are correct only in a first step of the algorithm, that's why we will present an algorithm in two steps.

Our first important result is Proposition 4, where we show that the size of minimal complete sets of E-unifiers can never be greater than the size of minimal complete sets of AC-unifiers nor of minimal complete sets of E-unifiers of the same unification problem. The main result is Theorem 9 where we show the soundness and completeness of our algorithm. The last important result is the validation in practice of our approach, shown by some benchmarks in Section 4.

This paper is organized as follows: we first study some general properties of AC-complete E-unification. Then we consider the elementary case (*i.e.* when there is only one AC operator in the signature), then we show how the known combination techniques for unification algorithms can be used to solve the general problem. Finally, we give some benchmarks obtained with our implementation[2] of normalized completion called C*i*ME, in which we implemented AC-complete unification modulo ACU and Abelian Groups.

1 Basic concepts

1.1 Definitions

As we mentioned above, we intend to represent all AC-unifiers by a set of E-unifiers for a theory E which extends AC. The main theories E we are interested in are the following:

Definition 1.

$$\begin{aligned}
\mathrm{AC}(+) &= \{x + y = y + x; x + (y + z) = (x + y) + z\} \\
\mathrm{ACU}(+, 0) &= \mathrm{AC}(+) \cup \{x + 0 = x\} \\
\mathrm{AG}(+, 0, -) &= \mathrm{ACU}(+, 0) \cup \{x + (-x) = 0\} \\
\mathrm{BR}(+, *, 0, 1) &= \mathrm{ACU}(+, 0) \cup \mathrm{ACU}(*, 1) \cup \\
&\quad \{x + x = 0; x * x = x; x * (y + z) = (x * y) + (x * z)\}
\end{aligned}$$

Whenever a theory E contains several AC-symbols $+_1, \ldots, +_n$, for short, we use AC instead of $\bigcup_{i=1}^{n} \mathrm{AC}(+_i)$.

[2] available as `http://www.lri.fr/~marche/cime.html`.

Definition 2. Let E be an equational theory on $\mathcal{T}(F,X)$ containing AC. Let P be a unification problem. An E-solution θ of P is *AC-compatible* if there exist σ and ρ such that $\sigma =_E \theta\rho$ and $\sigma \in \mathrm{Sol}_{\mathrm{AC}}(P)$. $\mathrm{Sol}_{E/\mathrm{AC}}(P)$ denotes the set of AC-compatible E-unifiers of P.

A set Θ of substitutions is an *AC-complete* set of E-solutions of P if

(i) $\forall \theta \in \Theta \;\; \theta \in \mathrm{Sol}_E(P)$. (Soundness)
(ii) $\forall \sigma \in \mathrm{Sol}_{\mathrm{AC}}(P) \;\; \exists \theta, \rho \;\;\; \theta \in \Theta \;\; \wedge \;\; \theta\rho =_E \sigma$. (Completeness)

An AC-complete set of E-solutions of P is said to be *minimal* if it has no proper subset that is still AC-complete.

Example 1. Consider the equation

$$f(x+y, z+t) = f(a+b, u+v)$$

where $+$ is associative-commutative, f is a free function symbol, a, b are free constants and x, y, z, t, u, v are variables. This problem has 14 most general *AC* unifiers corresponding to the conjunctions of one of the two solutions

$$\{x \mapsto a, y \mapsto b\} \quad \{x \mapsto b, y \mapsto a\}$$

of the equation $x + y = a + b$, and one of the seven solutions

$$\begin{array}{l}
\{z \mapsto w_1 + w_2, t \mapsto w_3 + w_4, u \mapsto w_1 + w_3, v \mapsto w_2 + w_4\} \\
\{z \mapsto w_1, t \mapsto w_3 + w_4, u \mapsto w_1 + w_3, v \mapsto w_4\} \\
\{z \mapsto w_2, t \mapsto w_3 + w_4, u \mapsto w_3, v \mapsto w_2 + w_4\} \\
\{z \mapsto w_1 + w_2, t \mapsto w_4, u \mapsto w_1, v \mapsto w_2 + w_4\} \\
\{z \mapsto w_1 + w_2, t \mapsto w_3, u \mapsto w_1 + w_3, v \mapsto w_2\} \\
\{z \mapsto w_2, t \mapsto w_3, u \mapsto w_3, v \mapsto w_2\} \\
\{z \mapsto w_1, t \mapsto w_4, u \mapsto w_1, v \mapsto w_4\}
\end{array}$$

of the equation $z + y = u + v$.

Now, the same equation has 4 most general ACU$(+, 0)$-unifiers, corresponding to the conjunctions of one of the four solutions

$$\{x \mapsto a+b, y \mapsto 0\} \;\; \{x \mapsto a, y \mapsto b\} \;\; \{x \mapsto b, y \mapsto a\} \;\; \{x \mapsto 0, y \mapsto a+b\}$$

of the equation $x + y = a + b$ and the unique most general ACU-unifier

$$\{z \mapsto w_1 + w_2, t \mapsto w_3 + w_4, u \mapsto w_1 + w_3, v \mapsto w_2 + w_4\}$$

of the equation $z + y = u + v$.

But the two ACU solutions which respectively map x and y onto 0 are not AC-compatible. In this case, a minimal, AC-complete set of ACU-unifiers is reduced to the two substitutions

$$\begin{array}{l}
\{x \mapsto a, y \mapsto b, z \mapsto w_1 + w_2, t \mapsto w_3 + w_4, u \mapsto w_1 + w_3, v \mapsto w_2 + w_4\} \\
\{x \mapsto b, y \mapsto a, z \mapsto w_1 + w_2, t \mapsto w_3 + w_4, u \mapsto w_1 + w_3, v \mapsto w_2 + w_4\}
\end{array}$$

The reader can check that the 14 AC-unifiers of the original equation can be obtained by mapping some of the variables introduced by these two substitutions onto 0.

Remark. In no way, AC-compatible E-unification can be seen as unification in a quotient algebra. Indeed two terms s and t can be equal modulo E while the first unifies with a term u and the second doesn't. For instance $x+y$ is equal to $1*(x+y)$ in Boolean rings but the first term has two most general AC-compatible BR-unifiers with $a+b$, and the second has none.

1.2 General Properties

As in the case of standard E-unification, a minimal, AC-complete set of E-unifiers is unique in a certain way:

Lemma 3. *If Σ_1 and Σ_2 are two minimal AC-complete sets of E-solutions of a problem P, then they are equivalent modulo the E-subsumption ordering. In particular they have the same cardinality.*

The proof is the same as in the standard E-unification case. As a consequence, every AC-complete set of E-solutions of P has a larger cardinality. The following property shows that AC-complete E-unification cannot be worse than AC-unification or E-unification, provided one can compute a *minimal* AC-complete set of E-unifiers.

Proposition 4. *For an arbitrary unification problem P, if Σ is a minimal AC-complete set of E-unifiers of P, then $|\mathrm{CSU}_{\mathrm{AC}}(P)| \geq |\Sigma|$ and $|\mathrm{CSU}_E(P)| \geq |\Sigma|$.*

Proof. We simply remark that $\mathrm{CSU}_{\mathrm{AC}}(P)$ is itself an AC-complete set of E-solutions of P, hence the first inequality is true by minimality of Σ (Lemma 3). The set $\mathrm{CSU}_E(P) \cap \mathrm{Sol}_{E/\mathrm{AC}}$ is also an AC-complete set of E-solutions of P hence the second inequality.

Removing from a minimal complete set of E-unifiers those solutions which are not AC-compatible always yields an AC-complete set of E-unifiers, but unfortunately, this does not guarantee minimality.

Example 2. Let E be the Boolean ring theory $\mathrm{BR}(+,*,0,1)$ on the term algebra $\mathcal{T}(\{+,,0,1\} \cup \{f\}, \mathcal{X})$. Consider the equation

$$f(x) + f(y) = f(u) + f(v),$$

where x, y, u and v are variables. This equation has only two most general AC-unifiers:

$$\sigma_1 \equiv \{x \mapsto z, y \mapsto w, u \mapsto z, v \mapsto w\} \quad \sigma_2 \equiv \{x \mapsto z, y \mapsto w, u \mapsto w, v \mapsto z\}$$

but it has three minimal E-unifiers which are AC-compatible: σ_1, σ_2 and

$$\sigma_3 \equiv \{x \mapsto z, y \mapsto z, u \mapsto w, v \mapsto w\}$$

But σ_3 is useless since its E-instances which are AC-solutions are not minimal, hence $\{\sigma_1, \sigma_2\}$ is an AC-complete set of E-unifiers.

2 The elementary case

As we already said, it is sound and complete to just replace AC-unification by E-unification, provided $\mathrm{AC} \subseteq E \subseteq T$. In this section we show how one can further reduce the number of unifiers in the elementary AC case (*i.e.* when the signature is reduced to one AC-operator and free constants), by discarding those E-unifiers which are not AC-compatible. We can even obtain a minimal AC-complete set of E-unifiers in two particular cases: ACU and unitary theories. This turns out to be sufficient for our purpose since besides ACU, the theories that we have in mind are those of Abelian groups and Boolean rings which happen to be unitary for unification.

2.1 AC-complete ACU-unification

It is easy to see if an ACU-unifier σ is AC-compatible. Indeed, if σ maps some variable x onto the unit, then no further instantiation will turn σ into an AC-unifier. On the other hand, if no variable is mapped onto the unit, then σ *is* an AC-unifier. Example 2 showed that discarding the useless E-unifiers from a minimal complete set of E-unifiers does not yield a minimal AC-complete set of E-unifiers in general. But in the case of ACU, the remaining substitutions are all incomparable AC-unifiers, hence the minimality of the obtained set.

2.2 Unitary theories

The case of unitary theories is even easier since if there exists a most general E-unifier σ and at least an AC-unifier, then necessarily $\{\sigma\}$ is a minimal AC-complete set of E-unifiers.

Lemma 5. *Assume that a unification problem has a most general E-unifier σ and is AC-unifiable. Then $\{\sigma\}$ is a minimal AC-complete set of E-unifiers.*

This holds because AC being a sub-theory of E, all the AC-solutions are E-solutions, and σ being a most general E-unifier represents all of them.

This particular case is relevant for us since two theories of interest (BR and AG) are unitary, and in addition meet the additional requirements that we describe below.

2.3 Further useful assumptions

We describe some useful properties that a theory E must have in order to use AC-compatible E-unification in the general case (*i.e.* in presence of several AC-operators and free function symbols). These are essentially the same assumptions that are made on equational theories for the combination of unification algorithms [24, 6, 1, 7].

Definition 6. A theory E defined on a signature $\mathcal{F}$ is *combinable* if

- E has a complete unification algorithm for terms in $\mathcal{T}(\mathcal{F} \cup \mathcal{C}, \mathcal{X})$, where $\mathcal{C}$ is a set of free constants.
- E has a *constant elimination algorithm*.

It is well-known that ACU, BR and AG are combinable theories [24].

Before to address the general case, let us give an example of the kind of speedup that can be obtained with our technique:

Example 3. Assume that $+$ is an AC-operator and that E is the theory of Boolean rings over $\{+, *, 0, 1\}$. All the equations of the form

$$x + s = t$$

for arbitrary terms s and t in which x has no occurrence have the most general E-unifier $\sigma = \{x \mapsto s + t\}$. Now, as soon as such an equation is AC-unifiable, $\{\sigma\}$ is an AC-complete set of E-unifiers.

3 The general case

One of the major difficulties with AC-unification is the combination of the elementary AC-unification algorithms for each AC-operator [25, 16, 11, 8]. Indeed, it took almost a decade before the termination of Stickel's combination algorithm [25] was proved by Fages[3] [11]. Yet AC theories, being *simple theories* (*i.e.* theories in which there is no equality between a term and one of its proper subterms), are in the easiest class for the combination problem.

What we intend to do here is not to re-design a combination procedure for our specific purpose but to reuse the standard combination techniques. A first difficulty is that the combination methods for simple theories are not complete for AC-compatible E-unification.

3.1 Purification

Let us see why we need more than the known combination techniques for AC-unification. The two rules of Figure 1 are well-known to be complete for AC-unification [8, 12]. A naïve attempt would be to use these rules since we can discard the input problems that are not AC-unifiable. This is not complete as exemplified below:

Example 4. Assume that $+$ and $*$ are AC symbols, and that we want to use $\mathrm{ACU}(+, 0) \cup \mathrm{ACU}(*, 1)$ as theory E for solving the problem

$$x + y = u + v \ \wedge \ x * y' = u * v'$$

[3] Stickel's algorithm goes back to '75 and Fages' proof to '84.

Clash
$x = f(s_1, \ldots, s_n) \ \wedge \ x = g(t_1, \ldots, t_m) \quad \rightarrow \ \perp$
if $f \neq g$.

Check*
$x_1 = s_1[x_2] \ \wedge \ x_2 = s_2[x_3] \ \wedge \ \cdots \ \wedge \ x_n = s_n[x_1] \quad \rightarrow \ \perp$
if some s_i is not a variable.

Fig. 1. rules **Clash** and **Check***

Replacing each equation by an ACU-solved form results in the problem:

$$\begin{array}{c} x = z_1 + z_2 \ \wedge \ y = z_3 + z_4 \ \wedge \ u = z_1 + z_3 \ \wedge \ v = z_2 + z_4 \\ \bigwedge \\ x = w_1 * w_2 \ \wedge \ y' = w_3 * w_4 \ \wedge \ u = w_1 * w_3 \ \wedge \ v' = w_2 * w_4 \end{array}$$

Clash applies to x and u but the theory conflicts can be solved by applying the substitution $\{z_2 \mapsto 0, w_3 \mapsto 1\}$. This produces the problem

$$\begin{array}{c} x = z_1 \ \wedge \ y = z_3 + z_4 \ \wedge \ u = z_1 + z_3 \ \wedge \ v = z_4 \\ \bigwedge \\ x = w_1 * w_2 \ \wedge \ y' = w_4 \ \wedge \ u = w_1 \ \wedge \ v' = w_2 * w_4 \end{array}$$

to which **Check*** now applies since there is a cycle

$$x = w_1 * w_2 \ \wedge \ w_1 = u \ \wedge \ u = z_1 + z_3 \ \wedge \ z_1 = x$$

Yet, the original problem is surely AC-unifiable since it unifies in the free theory.

What we want to point out is that the combination rules of AC-unification are not correct *after* solving a subproblem in E. This is because E-unification of a subproblem does not yield an AC-equivalent one. Conversely these rules are obviously correct *before* E-unification since they just transform an AC-unification problem into an equivalent one.

Hence the set of rules **Purify** of Figure 2 can be applied as a first step of an AC-complete E-unification algorithm. These rules terminate as a strict subset of the rules of [8, 6] for AC-unification.

What example 4 shows is that we need all the power of the combination techniques for arbitrary equational theories. We now show how to use these techniques for solving our problem.

3.2 The disjunction lemma

After applying the rules of **Purify** as long as possible, we obtain a problem which is either $\perp$, or a DAG-solved form [12], or a unification problem that we will write under the form

Decompose
$f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n) \quad \to \quad s_1 = t_1 \ \wedge \ \cdots \ \wedge \ s_n = t_n$
if f is not an AC operator.

Clash
$f(s_1, \ldots, s_n) = g(t_1, \ldots, t_m) \quad \to \ \perp$
if $f \neq g$.

Coalesce
$x = y \ \wedge \ P \quad \to \ x = y \ \wedge \ P\{x \mapsto y\}$
if $x, y \in \mathcal{V}(P)$ and x is existentially quantified or y is free.

Variable Abstraction
$s[u]_p = t \quad \to \ (\exists x) \ s[x]_p = t \ \wedge \ x = u$
if u is an alien subterm of s at position p with respect to the disjoint signatures $\{+_1\}, \ldots, \{+_n\}, (\mathcal{F} \setminus \{+_1, \ldots, +_n\})$.

Merge
$x = s \ \wedge \ x = t \quad \to \ x = s \ \wedge \ s = t$
If $s, t \notin \mathcal{X}$ and $|s| \leq |t|$.

Check*
$x_1 = s_1[x_2] \ \wedge \ x_2 = s_2[x_3] \ \wedge \ \cdots \ \wedge \ x_n = s_n[x_1] \quad \to \ \perp$
if some s_i is not a variable.

Fig. 2. the set of rules **Purify** for AC-complete E-unification

$$P \equiv (\exists y_1, \ldots, y_m) P_V \ \wedge \ P_\emptyset \ \wedge \ P_{+_1} \ \wedge \ \cdots \ \wedge \ P_{+_n}$$

where

- P_V contains only equations of the form $x = y$ with $x, y \in \mathcal{X}$ and x or y has no other occurrence in P.
- $P_\emptyset \equiv x_1 = s_1 \ \wedge \ \cdots \ \wedge \ x_n = s_n$ where s_i is a non-variable term with no occurrence of an AC-operator.
- P_{+_i} is a conjunction of pure equations in $AC(+_i)$.
- **Clash**, **Merge** and **Check*** do not apply.

Now, we wish to use a theory E, extending $\bigcup_i \mathrm{AC}(+_i)$ instead of AC-unification for solving the pure AC-subproblems P_{+_i}. But this is no longer unification in a combination of disjoint equational theories: if the AC-symbols are $+$ and $*$, and the theory E is a Boolean ring over $\{+, *, 0, 1\}$, AC($+$)-complete BR-unification and AC($*$)-complete BR-unification will not be performed with disjoint alphabets. We show that we can simulate the disjoint case by a technical trick which reflects the fact that AC($+$) and AC($*$) *are* disjoint theories. We shall use different *copies* of E for each AC-operator.

Definition 7. Let E be an equational theory defined on $\mathcal{T}(\mathcal{F}, \mathcal{X})$ and containing $\mathrm{AC} = \bigcup_{i=1}^{n} \mathrm{AC}(+_i)$. A copy of $\mathcal{F}$ with respect to $+_i$ is a one-to-one mapping κ_i from $\mathcal{F}$ onto a set $\mathcal{F}_i$ such that

- $\kappa_i(+_i) = +_i$,
- $\kappa_i(f) = f$ if f is a free function symbol,
- $\kappa_i(f) = f^i \quad (\neq f)$ otherwise.

κ_i can be uniquely extended to $\mathcal{T}(\mathcal{F}, \mathcal{X})$. $\kappa_i(E)$ is the equational theory defined on $\mathcal{T}(\mathcal{F}_i, \mathcal{X})$ by replacing the symbols f in the equations of E by $\kappa_i(f)$. $\kappa(E)$ is the equational theory $\bigcup_{i=1}^{n} \kappa_i(E)$ defined on $\mathcal{T}(\mathcal{F} \cup \bigcup_{i=1}^{n} \mathcal{F}_i), \mathcal{X})$. κ^{-1} retrieves the original function symbols:

- $\kappa^{-1}(f) = f$ if f is in $\mathcal{F}$,
- $\kappa^{-1}(f^i) = \kappa_i^{-1}(f^i)$ otherwise (f^i is in $\mathcal{F}_i$).

Lemma 8 Disjunction lemma. *Let Θ be an AC-complete set of $\kappa(E)$-unifiers for the unification problem*

$$P \equiv (\exists y_1, \ldots, y_m) \ \ P_V \ \wedge \ P_\emptyset \ \wedge \ P_{+_1} \ \wedge \ \cdots \ \wedge \ P_{+_n}$$

obtained from the original problem by applying as long as possible the rules of ***Purify****. Then $\kappa^{-1}(\Theta)$ is an AC-complete set of E-unifiers for P.*

Proof. First, it is obvious that if θ is a $\kappa(E)$-unifier for P, $\kappa^{-1}(\theta)$ is an E-unifier: any proof modulo $\kappa(E)$ can still be performed modulo E with the "original" function symbols.

Now let σ be an AC-unifier of P. We can assume without loss of generality that σ is minimal, hence σ contains only functions symbols occuring in $\mathcal{F}$: this implies that $\kappa^{-1}(\sigma) = \sigma$. Since Θ is AC-complete, there exist θ and ρ such that $\theta \in \Theta$ and $\sigma =_{\kappa(E)} \theta\rho$. Hence $\sigma = \kappa^{-1}(\sigma) =_E \kappa^{-1}(\theta)\kappa^{-1}(\rho)$.

This simple result allows us to reuse the machinery of unification in combination of disjoint equational theories. Elementary AC-unification will be replaced by E-unification in the associated copy. For this, recall that we must be able to compute AC-complete sets of E-unifiers with the traditional additional restrictions on E.

Theorem 9. *Let $\mathrm{AC} = \bigcup_{i=1}^{n} \mathrm{AC}(+_i)$. Let $E \supseteq \mathrm{AC}$ be a combinable equational theory. The following algorithm*

1. *apply as long as possible the rules of* ***Purify****,*
2. *run a combination algorithm for arbitrary combinable equational theories replacing elementary $\mathrm{AC}(+_i)$-unification by AC-complete $\kappa_i(E)$-unification,*
3. *apply κ^{-1} to the solutions computed at step 2,*

is an AC-complete E-unification algorithm.

4 Some experiments

In this section we present some experiments with our prototype CiME. We have used the technique due to Boudet [7] for combining the elementary AC-complete unification algorithms. In the tables below, T is the normalization theory used for rewriting, and E is the unification theory. For each case we give the number of calls to unification, the number of solutions generated, the total computation time of completion, and the time spent in unification (computation times are in seconds, on a Sparc station 5 running Solaris 2.4).

4.1 Abelian Group theory

T	E	# calls to unif.	# of sols	total CPU time	unif. time
AC(+)	AC(+)	37	227	15.7	8.1
ACU(+, 0)	AC(+)	30	206	14.2	6.7
	ACU(+, 0)	37	47	7.53	5.03

In this case, things get clearly better with ACU unification.

4.2 Ring theory

The Ring theory is presented by

$$\begin{array}{rcl} x + 0 = x & x * 1 = x & (x * y) * z = x * (y * z) \\ x + (-x) = 0 & 1 * x = x & x * (y + z) = (x * y) + (x * z) \\ & & (x + y) * z = (x * z) + (y * z) \end{array}$$

where + is AC.

T	E	# calls to unif.	# of sols	total CPU time	unif.time
AC(+)	AC(+)	143	407	34.7	12.8
ACU(+, 0)	AC(+)	128	372	32.7	11.55
	ACU(+, 0)	509	495	106.0	20.8
ACU(+, 0) AU($*$, 1)	ACU(+, 0)	337	375	77.0	16.9
AG(+, 0, −)	AC(+)	75	100	8.05	1.5
	ACU(+, 0)	75	56	5.35	0.92
	AG(+, 0, −)	75	52	5.13	0.90
AG(+, 0, −) AU($*$, 1)	AC(+)	25	67	7.77	1.07
	ACU(+, 0)	25	23	4.53	0.57
	AG(+, 0, −)	25	19	3.40	0.50

In this case, it's getting worse when using ACU-normalized completion with ACU-unification : in fact, there are problems with CiME's completion strategy, many useless rules (like $0.0 \to 0$ or $x.-(1) \to -(x)$) are considered. An evidence of this strategy problem is obtained if we try to complete the already known convergent set of rules for commutative rings, that is if we simply want to prove the confluence. The results are then as follows:

T	E	# calls to unif.	# of sols	total CPU time	unif.time
ACU$(+, 0)$	AC$(+)$	81	169	14.7	2.7
	ACU$(+, 0)$	81	71	7.5	1.6

In fact, it is obvious that when completion is simply used for proving confluence, ACU unification is always better, since there is no strategy to choose in that case.

4.3 Commutative Ring theory

The Commutative Ring theory is presented by the equations

$$\begin{array}{cc} x + 0 = x & x * 1 = x \\ x + (-x) = 0 & x * (y + z) = (x * y) + (x * z) \end{array}$$

where $+$ and $*$ are AC.

T	E	# calls to unif.	# of sols	total CPU time	unif.time
AC$(+)$ AC$(*)$	AC$(+)$ AC$(*)$	69	324	24.7	11.9
ACU$(+, 0)$ ACU$(*, 1)$	AC$(+)$ AC$(*)$	66	340	28.3	11.9
	ACU$(+, 0)$ ACU$(*, 1)$	98	141	42.0	29.7
AG$(+, 0, -)$ AC$(*)$	AC$(+)$ AC$(*)$	12	46	3.30	0.77
	ACU$(+, 0)$ AC$(*)$	12	24	2.65	0.92
	AG$(+, 0, -)$ AC$(*)$	12	22	2.07	0.75
AG$(+, 0, -)$ ACU$(*, 1)$	AC$(+)$ AC$(*)$	8	39	3.30	0.78
	ACU$(+, 0)$ ACU$(*, 1)$	8	12	1.87	0.58
	AG$(+, 0, -)$ ACU$(*, 1)$	8	10	1.35	0.45

Again, there are some problems with the completion strategy in the case of ACU-normalized completion with ACU unification, but in the other cases ACU-unification and AG-unification are better.

4.4 A finitely generated Abelian group

Let us complete the presentation of the Abelian group generated by a, b, c and the relations

$$2a - 3b + c = 0 \quad -3a + 2b + 3c = 0 \quad 2a + 2b - 2c = 0$$

T	E	# calls to unif.	# of sols	total CPU time	unif. time
AC(+)	AC(+)	369	1531	278.0	46.0
ACU(+, 0)	AC(+)	334	1466	288.0	44.05
	ACU(+, 0)	176	419	240.0	19.8
AG(+, 0, −)	AC(+)	17	13	4.97	0.22
	ACU(+, 0)	17	13	5.07	0.22
	AG(+, 0, −)	17	13	5.10	0.25

Here, ACU unification is clearly an enhancement for ACU-normalized completion. AG-normalized completion is much more efficient, so we are not able to see any difference between the different unification theories.

4.5 A Gröbner basis of a polynomial ideal

It has been remarked that CR-normalized completion (where CR is commutative rings theory) can be used for computing polynomial ideals over $\mathbb{Z}$. Here is an example:

$$2X^2Y - Y = 0 \quad 3XY^2 - X = 0$$

which we complete modulo CR$(+, 0, -, *, 1)$.

T	E	# calls to unif.	# of sols	total CPU time	unif. time
CR	AC(+) AC(∗)	176	108	38.8	6.4
	ACU(+, 0) ACU(∗, 1)	201	89	34.8	7.0
	AG(+, 0, −) ACU(∗, 1)	201	90	36.95	6.22

In this case, ACU unification is slightly better. Note that the problem with the completion strategy arises here also (there are more calls to unification), hence we expect better results with a clever strategy.

We have obtained similar results with polynomial ideals over finite fields.

5 Conclusion

Thanks to the use of AC-complete E-unification, we succeeded to reduce significantly the number of critical pairs generated during normalized completion. AC-complete E-unification computes less critical pairs than AC unification or E-unification. The implementation, though, highlightened some lacks in the completion strategy of our prover C*i*ME. From an algebraic point of view, it is not clear what minimal AC-complete sets of E-unifiers are. We saw that they are *not* complete sets of unifiers modulo an equational theory. From the practical point of view, we do not know how to compute minimal sets, although our experiments on the prototype implementation shows that we do compute small sets of unifiers in practice. Note however that the standard combination techniques for

AC-unification do not compute minimal sets either in general. One important thing that remains to do is to study how to combine this new method with other known methods for reducing the number of critical pairs : critical pair criteria [2, 15]. Another future work is to extend our results to Horn clauses, that is to study the use of AC-complete unification with paramodulation inference rules.

References

1. F. Baader and K. Schulz. Unification in the union of disjoint equational theories: Combining decision procedures. In D. Kapur, editor, *Proc. 11th Int. Conf. on Automated Deduction, Saratoga Springs, NY, LNAI 607*, 1992.
2. L. Bachmair and N. Dershowitz. Critical pair criteria for completion. *Journal of Symbolic Computation*, 6(1):1–18, 1988.
3. L. Bachmair and N. Dershowitz. Completion for rewriting modulo a congruence. *Theoretical Comput. Sci.*, 67(2&3):173–201, Oct. 1989.
4. L. Bachmair, H. Ganzinger, C. Lynch, and W. Snyder. Basic paramodulation and superposition. In D. Kapur, editor, *Proc. 11th Int. Conf. on Automated Deduction, Saratoga Springs, NY, LNAI 607*. Springer-Verlag, June 1992.
5. T. Baird, G. Peterson, and R. Wilkerson. Complete sets of reductions modulo Associativity, Commutativity and Identity. In *Proc. 3rd Rewriting Techniques and Applications, Chapel Hill, LNCS 355*, pages 29–44. Springer-Verlag, Apr. 1989.
6. A. Boudet. *Unification dans les Mélanges de Théories équationnelles.* Thèse de doctorat, Université Paris-Sud, Orsay, France, Feb. 1990.
7. A. Boudet. Combining unification algorithms. *Journal of Symbolic Computation*, 16:597–626, 1993.
8. A. Boudet, E. Contejean, and H. Devie. A new AC-unification algorithm with a new algorithm for solving diophantine equations. In *Proc. 5th IEEE Symp. Logic in Computer Science, Philadelphia*, pages 289–299. IEEE Computer Society Press, June 1990.
9. B. Buchberger. *An Algorithm for Finding a Basis for the Residue Class Ring of a Zero-Dimensional Ideal.* PhD thesis, University of Innsbruck, Austria, 1965. (in German).
10. B. Buchberger and R. Loos. Algebraic simplification. In *Computer Algebra, Symbolic and Algebraic Computation. Computing Supplementum 4*. Springer-Verlag, 1982.
11. F. Fages. Associative-commutative unification. *Journal of Symbolic Computation*, 3(3), June 1987.
12. J.-P. Jouannaud and C. Kirchner. Solving equations in abstract algebras: A rule-based survey of unification. In J.-L. Lassez and G. Plotkin, editors, *Computational Logic: Essays in Honor of Alan Robinson.* MIT-Press, 1991.
13. J.-P. Jouannaud and H. Kirchner. Completion of a set of rules modulo a set of equations. *SIAM J. Comput.*, 15(4):1155–1194, 1986.
14. J.-P. Jouannaud and C. Marché. Termination and completion modulo associativity, commutativity and identity. *Theoretical Comput. Sci.*, 104:29–51, 1992.
15. D. Kapur, D. Musser, and P. Narendran. Only prime superpositions need be considered for the Knuth-Bendix procedure. *Journal of Symbolic Computation*, 4:19–36, 1988.

16. C. Kirchner. Méthodes et outils de conception systématique d'algorithmes d'unification dans les théories equationnelles. Thèse d'Etat, Univ. Nancy, France, 1985.
17. C. Kirchner, editor. *Unification*. Academic Press, 1990.
18. D. E. Knuth and P. B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970.
19. D. S. Lankford and A. M. Ballantyne. Decision procedures for simple equational theories with permutative axioms: Complete sets of permutative reductions. Research Report Memo ATP-37, Department of Mathematics and Computer Science, University of Texas, Austin, Texas, USA, Aug. 1977.
20. R. Loos. Term reduction systems and algebraic algorithms. In *Proceedings of the Fifth GI Workshop on Artificial Intelligence*, pages 214–234, Bad Honnef, West Germany, 1981. Available as *Informatik Fachberichte*, Vol. 47.
21. C. Marché. Normalised rewriting and normalised completion. In *Proceedings of the Ninth Annual IEEE Symposium on Logic in Computer Science*, pages 394–403, Paris, France, July 1994. IEEE Comp. Soc. Press.
22. R. Nieuwenhuis and A. Rubio. Basic superposition is complete. In B. Krieg-Bruckner, editor, *Proc. European Symp. on Programming, LNCS 582*, pages 371–389, Rennes, 1992. Springer-Verlag.
23. G. E. Peterson and M. E. Stickel. Complete sets of reductions for some equational theories. *J. ACM*, 28(2):233–264, Apr. 1981.
24. M. Schmidt-Schauß. Unification in a combination of arbitrary disjoint equational theories. *Journal of Symbolic Computation*, 1990. Special issue on Unification.
25. M. Stickel. A unification algorithm for associative-commutative functions. *J. ACM*, 28(3):423–434, 1981.

This article was processed using the LaTeX macro package with LLNCS style