"Syntactic" AC-Unification*

Alexandre Boudet and Evelyne Contejean

LRI, CNRS URA 410
Bât 490, Université Paris-Sud, Centre d'Orsay
91405 Orsay Cedex, France

Abstract. The rules for unification in a simple syntactic theory, using Kirchner's *mutation* [15, 16] do not terminate in the case of associative-commutative theories. We show that in the case of a linear equation, these rules terminate, yielding a complete set of solved forms, each variable introduced by the unifiers corresponding to a (trivial) minimal solution of the (trivial) Diophantine equation where all coefficients are 1. A non-linear problem can be first treated as a linear one, that is considering two occurrences of a same variable as two different variables. After this step, one has to solve the equations between the different values that have been obtained for the different occurrences of a same variable. We show that one can restrict the search of the solutions of these latter equations to linear substitutions. This result is based on the analysis of how the minimal solutions of a linear Diophantine equation can be built-up using the solutions of the trivial Diophantine equation associated with the linearized AC-equation. This provides a new AC-unification algorithm which does not make an explicit use of the solving of linear Diophantine equations.

1 Introduction

The *syntactic theories* were defined in '85 by Kirchner as those collapse-free equational theories which admit a finite *resolvent presentation*, this being a presentation where any equality proof can be performed with at most one application of an axiom at the root [15, 16]. At that time, the unification community was not aware that there were many syntactic theories on top of commutativity and its variants. The main advantage of syntacticness is that one can nondeterministically guess which axiom of a resolvent presentation will possibly apply at the root. This provides top-down strategies for solving word problems or unification problems which are complete but not terminating in general.

In '90, it turned out that a lot of theories of interest were syntactic. Kirchner and Klay noticed that a collapse-free theory E over a signature $\mathcal{F}$ is syntactic if and only if all the *general equations* of the form $f(x_1, \ldots, x_n) \stackrel{?}{=} g(y_1, \ldots, y_m)$, where $f, g \in \mathcal{F}$, have a finite complete set of E-unifiers $\Sigma_{f,g}$ [17]. In this case, the equations $f(x_1, \ldots, x_n)\sigma = g(y_1, \ldots, y_m)\sigma$, where σ belongs to $\Sigma_{f,g}$, form a resolvent presentation of E. In particular, the associative-commutative theories

* This research was supported in part by the Esprit Working Group CCL.

having finitary unification, are syntactic, as shown with a nice geometric interpretation by Nipkow [20]. This led to a growing interest in syntactic theories, but Klay showed that it is not decidable whether a theory is syntactic, worse: the word problem is not decidable in general in syntactic theories [18].

A large class of theories for which unification can be decided by syntactic means has been given by Comon, Jouannaud and Haberstrau [6]. The authors dropped the assumption that the theories are collapse-free, and introduced the notion of *cycle-syntacticness* for solving the cycles of the form $x \stackrel{?}{=} u[x]$. They have shown that the *shallow theories*, *i.e.* the theories which have a presentation where all the variables in the axioms are at depth at most 1, are syntactic and cycle-syntactic, and have a decidable first-order theory.

Syntacticness has been used for unification modulo one-sided distributivity [22]. In this work, no mention is made of syntacticness, nevertheless the authors do actually show that the presentation reduced to the distributivity axiom is resolvent and that in this particular case, the unification process terminates.

Surprisingly, while associativity-commutativity is doubtless the theory for which unification has been the most extensively investigated [19, 21, 15, 9, 10, 12, 5, 4, 3, 1, 14, 8, 2], it has not been taken advantage of the syntacticness for AC-unification. The problem is that the syntactic method, while it constructs all the AC-solutions, does not terminate. Actually, an attempt has been made by Franzen and Henschen in '88 who already use a resolvent presentation of AC for unification [11]. There, the authors give a criterion for pruning the search space which ensures the termination, and they conjecture that the completeness is preserved. Their conjecture is that one can avoid the recursive calls which are not strictly smaller than the input problem for the ordering which compares first the maximal number of occurrences of a same variable in the equation, and second the size of the equation. We lack space for developing a counter-example, but the reader can check that their algorithm will not find the solution $\{x \mapsto v_1 + v_1 + v_2 + v_3, y \mapsto v_1 + v_1 + v_1 + v_2, z \mapsto v_2 + v_3 + v_3 + v_3\}$ of the equation $x + x + x \stackrel{?}{=} y + y + z$. In the following, we give another criterion, we prove that the completeness is preserved, and give a control for which the rules terminate, hence providing a new AC-unification algorithm which does not make an explicit use of the solving of linear Diophantine equations.

Let us make clear that our goal was not to give an efficient algorithm which could compete with the existing methods, but instead to *understand* how the axioms of a resolvent presentation of AC do indeed solve linear Diophantine equations.

2 Syntactic Theories and AC-Unification

In this section we recall some basic concepts about syntactic theories and AC-unification. We assume the reader is familiar with term rewriting and unification, and we use the notations of [7]. For instance, $\mathcal{T}(\mathcal{F}, \mathcal{X})$ is the free $\mathcal{F}$-algebra over a set $\mathcal{X}$ of variables, $t|_p$ is the subterm of t at position p and $t[u]_p$ the term t where the subterm at position p has been replaced by u. Λ is the empty (root) position

of a term. For any syntactic object o, $V(o)$ is the set of variables occurring in o. The reader is referred to [13] for a state-of-the-art survey of unification.

2.1 Syntactic Theories

Definition 1. *Given a set $E = \{l_1 = r_1, \ldots, l_n = r_n\}$ of axioms (or identities), where $l_i, r_i \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, the* equational theory $=_E$ generated by E *is the least congruence on $\mathcal{T}(\mathcal{F}, \mathcal{X})$ containing all the instances of the axioms. The theory $=_E$ is* consistent *if the set of its equivalence classes is not a singleton. E is called a* presentation *of the equational theory $=_E$. Equivalently, $=_E$ is the symmetric, reflexive, transitive closure $\overset{*}{\leftrightarrow}_E$ of the relation $\leftrightarrow_E$ defined by $s \leftrightarrow_E t$ if there exists an axiom $l = r$ of E, a position p and a substitution θ such that $s|_p \equiv l\theta$ and $t \equiv s[r\theta]_p$. $s \leftrightarrow_E t$ is called a* proof step *and if $p = \Lambda$, we call it a Λ-*step.

The equational theory that we will study here is $=_{AC}$, the theory presented by the two axioms of associativity (A): $(x + y) + z = x + (y + z)$, and commutativity (C): $x + y = y + x$, on $\mathcal{T}(\{+\}, \mathcal{X})$.

By abuse of notation, we will often confuse the equational theory and one of its presentations, hence writing "the theory AC".

Definition 2. *[15, 16] Let $\approx$ be a collapse-free equational theory* i.e. *a theory such that there is no equality $x \approx u[x]_p$ with $x \in \mathcal{X}$ and $p \neq \Lambda$. The theory $\approx$ is* syntactic *if it has a finite* resolvent presentation *E, that is a presentation such that every equality $s \approx t$ has a proof $s \overset{*}{\leftrightarrow}_E t$ with at most one Λ-step.*

It happens that AC is a syntactic theory, but the presentation given above is not a resolvent presentation. Indeed, two Λ-steps are necessary to prove the equality $(x + y) + (z + t) =_{AC} (x + z) + (y + t)$, using only the associativity and commutativity axioms. The following five axioms form a resolvent presentation of AC, which may be obtained from the (most general) AC-solutions of the general equation $x+y = u+v$ [17, 11], or by an analysis of the possible exchanges between the left and right subterms of the root [20]:

$$\begin{array}{cc} (x + y) + z = x + (y + z) & \text{A} \\ x + y = y + x & \text{C} \\ (x + y) + z = (x + z) + y & \text{RC} \\ x + (y + z) = y + (x + z) & \text{LC} \\ (x + y) + (z + t) = (x + z) + (y + t) & \text{MC} \end{array}$$

2.2 *AC*-unification

Definition 3. *A* unification problem *is a (disjunction of) formula(s) of the form T, F, or $P \equiv (\exists y_1, \ldots, y_p)\ s_1 \overset{?}{=} t_1 \wedge \cdots \wedge s_n \overset{?}{=} t_n$, where $s_i, t_i \in \mathcal{T}(\mathcal{F}, \mathcal{X})$ for $1 \leq i \leq n$. A substitution σ is an E-unifier (or E-solution) of P if $P\sigma$ is valid when $\overset{?}{=}$ is interpreted as the equational theory $=_E$. If in addition every E-unifier θ is equal (modulo E) to $\sigma\rho$ for some substitution ρ, then σ is called a*

most general E-unifier of P. *Every substitution is a solution of* T, *while* F *has no solution. Two unification problems are* E-equivalent *if they have the same set of* E*-unifiers.*

A unification problem is in a solved form *if* $P \equiv T$, *or* $P \equiv F$ *or* $P \equiv (\exists y_1, \ldots, y_p)\ x_1 \stackrel{?}{=} t_1 \wedge \cdots \wedge x_n \stackrel{?}{=} t_n$, *where the* x_i*s are free variables and have exactly one occurrence in* P.
A term t *(resp. an equation* $s \stackrel{?}{=} t$*) is* linear *if each of its variables has only one occurrence.*

It is well-known that solved forms (other than T and F) represent their own most general unifier: if $P \equiv (\exists y_1, \ldots, y_p)\ x_1 \stackrel{?}{=} t_1 \wedge \cdots \wedge x_n \stackrel{?}{=} t_n$ is in a solved form, then the substitution $\sigma = \{x_1 \mapsto t_1, \cdots, x_n \mapsto t_n\}$ is a most general E-unifier of P for every consistent equational theory E.

Definition 4. *A* complete set of solved forms *of* P *is a set* $\{P_1, \ldots, P_n\}$ *of solved forms such that* P *and* $P_1 \vee \cdots \vee P_n$ *are equivalent. For* $1 \leq i \leq n$, P_i *is called a* solved form of P. *The set of* E*-unifiers associated with a complete set of solved forms is called a* complete set of E-unifiers of P[1].

We briefly recall how one can derive a *mutation rule* for unification in a syntactic theory [11, 17]. Consider the equation $f(s_1, \ldots, s_n) \stackrel{?}{=} g(t_1, \ldots, t_m)$ to be solved in a theory E. After applying variable abstraction immediately under the root on both sides of the equation, one gets the E-equivalent problem:

$$(\exists \boldsymbol{x}, \boldsymbol{y}) f(x_1, \ldots, x_n) \stackrel{?}{=} g(y_1, \ldots, y_m) \bigwedge_{i=1}^{n} x_i \stackrel{?}{=} s_i \bigwedge_{j=1}^{m} y_j \stackrel{?}{=} t_j$$

where $\boldsymbol{x}, \boldsymbol{y}$ denote the new variables introduced by the variable abstraction. We can solve the general equation $f(x_1, \ldots, x_n) \stackrel{?}{=} g(y_1, \ldots, y_m)$ and, if $\Sigma_{f,g}$ is a finite complete set of E-unifiers, we get the finite disjunction:

$$\bigvee_{\sigma \in \Sigma_{f,g}} ((\exists \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \bigwedge_{i=1}^{n} x_i \stackrel{?}{=} x_i\sigma \bigwedge_{j=1}^{m} y_j \stackrel{?}{=} y_j\sigma \bigwedge_{i=1}^{n} x_i \stackrel{?}{=} s_i \bigwedge_{j=1}^{m} y_j \stackrel{?}{=} t_j)$$

where $\boldsymbol{z}$ denotes the new variables introduced by σ. Term replacement can now be applied to the equations of the two first conjunctions of each disjunct, yielding:

$$\bigvee_{\sigma \in \Sigma_{f,g}} ((\exists \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \bigwedge_{i=1}^{n} x_i \stackrel{?}{=} x_i\sigma \bigwedge_{j=1}^{m} y_j \stackrel{?}{=} y_j\sigma \bigwedge_{i=1}^{n} x_i\sigma \stackrel{?}{=} s_i \bigwedge_{j=1}^{m} y_j\sigma \stackrel{?}{=} t_j)$$

Now, the existentially quantified variables appearing only as a left-hand side of an equation $x \stackrel{?}{=} s$ are irrelevant, and we can remove such equations, thus obtaining the following problem which is E-equivalent to the original equation:

$$\bigvee_{\sigma \in \Sigma_{f,g}} ((\exists \boldsymbol{z}) \bigwedge_{i=1}^{n} x_i\sigma \stackrel{?}{=} s_i \bigwedge_{j=1}^{m} y_j\sigma \stackrel{?}{=} t_j)$$

Hence, the mutation rule for the theories having finite complete sets

[1] Here, we restrict our attention to the theories having finite complete sets of unifiers.

of unifiers $\Sigma_{f,g}$ of the general equations $f(x_1,\ldots,x_n) \stackrel{?}{=} g(y_1,\ldots,y_m)$:

Mutate

$$f(s_1,\ldots,s_n) \stackrel{?}{=} g(t_1,\ldots,t_m) \wedge P \;\rightarrow \bigvee_{\sigma\in\Sigma_{f,g}} ((\exists \boldsymbol{z}) \bigwedge_{i=1}^{n} x_i\sigma \stackrel{?}{=} s_i \bigwedge_{j=1}^{m} y_j\sigma \stackrel{?}{=} t_j \wedge P)$$

where $\Sigma_{f,g}$ is a complete set of E-unifiers of $f(x_1,\ldots,x_n) \stackrel{?}{=} g(y_1,\ldots,y_m)$, and the variables of $\boldsymbol{z}$ are the new variables introduced by σ.

Applied to AC, this rule allows us to transform the equation $s_1+s_2 \stackrel{?}{=} t_1+t_2$ into the AC-equivalent disjunction of the seven problems:

$(\exists v_1, v_2)$	$s_1 \stackrel{?}{=} v_1 \wedge s_2 \stackrel{?}{=} v_2 \wedge t_1 \stackrel{?}{=} v_1 \wedge t_2 \stackrel{?}{=} v_2$	(Id)
$(\exists v_1, v_2)$	$s_1 \stackrel{?}{=} v_1 \wedge s_2 \stackrel{?}{=} v_2 \wedge t_1 \stackrel{?}{=} v_2 \wedge t_2 \stackrel{?}{=} v_1$	(C)
$(\exists v_1, v_2, v_3)$	$s_1 \stackrel{?}{=} v_1 + v_2 \wedge s_2 \stackrel{?}{=} v_3 \wedge t_1 \stackrel{?}{=} v_1 \wedge t_2 \stackrel{?}{=} v_2 + v_3$	($A_\rightarrow$)
$(\exists v_1, v_2, v_3)$	$s_1 \stackrel{?}{=} v_1 \wedge s_2 \stackrel{?}{=} v_2 + v_3 \wedge t_1 \stackrel{?}{=} v_1 + v_2 \wedge t_2 \stackrel{?}{=} v_3$	($A_\leftarrow$)
$(\exists v_1, v_2, v_3)$	$s_1 \stackrel{?}{=} v_1 + v_2 \wedge s_2 \stackrel{?}{=} v_3 \wedge t_1 \stackrel{?}{=} v_1 + v_3 \wedge t_2 \stackrel{?}{=} v_2$	(RC)
$(\exists v_1, v_2, v_3)$	$s_1 \stackrel{?}{=} v_1 \wedge s_2 \stackrel{?}{=} v_2 + v_3 \wedge t_1 \stackrel{?}{=} v_2 \wedge t_2 \stackrel{?}{=} v_1 + v_3$	(LC)
$(\exists v_1, v_2, v_3, v_4)$	$s_1 \stackrel{?}{=} v_1 + v_2 \wedge s_2 \stackrel{?}{=} v_3 + v_4 \wedge t_1 \stackrel{?}{=} v_1 + v_3 \wedge t_2 \stackrel{?}{=} v_2 + v_4$	(MC)

The reader has recognized the use of the five axioms of the resolvent presentation of AC given above in paramodulations followed by decomposition. (The first problem corresponds to the case where no Λ-step is needed, and associativity can be used from left to right or from right to left.)

Of course, the **Mutate** rule is not sufficient by itself for computing a (disjunction of) solved form(s). In order to have a unification algorithm, we need some additional rules (given in figure 1) which are all well-known and whose correctness is straightforward. Some of these rules are redundant, but we want to define the largest class of algorithms, depending on the possible controls. These rules are complete for the *simple* syntactic theories, a simple theory being a theory which has no equality of the form $u =_E s[u]_p$ with $p \neq \Lambda$.

Lemma 1. *The irreducible problems for $\mathcal{S}$ are solved forms.*

Unfortunately, the rules of $\mathcal{S}$ do not terminate in general. Consider the problem: $x + x \stackrel{?}{=} y + y$. **Mutate** will yield (among others) the problem obtained by applying MC:

$$(\exists v_1, v_2, v_3, v_4)\;\; x \stackrel{?}{=} v_1 + v_2 \;\wedge\; x \stackrel{?}{=} v_3 + v_4 \;\wedge\; y \stackrel{?}{=} v_1 + v_3 \;\wedge\; y \stackrel{?}{=} v_2 + v_4$$

Merge, applied to x will build the equation $v_1 + v_2 \stackrel{?}{=} v_3 + v_4$. One solved form of this equation is $v_1 \stackrel{?}{=} v_3$ and $v_2 \stackrel{?}{=} v_4$. But now, identifying v_1 and v_3 on one hand, v_2 and v_4 on the other hand in the remaining equations turns them into $y \stackrel{?}{=} v_1 + v_1 \;\wedge\; y \stackrel{?}{=} v_2 + v_2$. **Merge**, applied to these equations yields a renaming of the original problem.

Mutate

$$f(s_1,\ldots,s_n) \stackrel{?}{=} g(t_1,\ldots,t_m) \wedge P \;\rightarrow\; \bigvee_{\sigma\in\Sigma_{f,g}} ((\exists \boldsymbol{z}) \bigwedge_{i=1}^{n} x_i\sigma \stackrel{?}{=} s_i \bigwedge_{j=1}^{m} y_j\sigma \stackrel{?}{=} t_j \wedge P)$$

where $\Sigma_{f,g}$ is a complete set of E-unifiers of $f(x_1,\ldots,x_n) \stackrel{?}{=} g(y_1,\ldots,y_m)$, and the variables of $\boldsymbol{z}$ are the new variables introduced by σ.

Merge

$x \stackrel{?}{=} s \;\wedge\; x \stackrel{?}{=} t \;\rightarrow\; x \stackrel{?}{=} s \;\wedge\; s \stackrel{?}{=} t$
if $x \in \mathcal{X}$ and $s, t \notin \mathcal{X}$

Var-Rep (Variable Replacement)

$(\exists z_1,\ldots,z_n)\;\; x \stackrel{?}{=} y \;\wedge\; P \;\rightarrow\; (\exists z_1,\ldots,z_n)\;\; x \stackrel{?}{=} y \;\wedge\; P\{x \mapsto y\}$
if $x, y \in V(P)$ and x is existentially quantified or y is free.

Rep (Replacement)

$(\exists y_1,\ldots,y_n)\;\; x \stackrel{?}{=} s \;\wedge\; P \;\rightarrow\; (\exists y_1,\ldots,y_n)\;\; x \stackrel{?}{=} s \;\wedge\; P\{x \mapsto s\}$
if $x \in V(P)$, and $s \notin \mathcal{X}$, and $x \notin V(s)$ or
if $s \in \mathcal{X}$, and $s \not\equiv x$ and $x, s \in V(P)$.

Check*

$x_1 \stackrel{?}{=} t_1[x_2]_{p_1} \;\wedge\; x_2 \stackrel{?}{=} t_2[x_3]_{p_2} \;\wedge\; \cdots \;\wedge\; x_n \stackrel{?}{=} t_n[x_1]_{p_n} \;\rightarrow\; F$
if some $p_i \neq \Lambda$.

EQE (Existential Quantifiers Elimination)

$(\exists x, y_1,\ldots,y_n)\;\; x \stackrel{?}{=} s \;\wedge\; P \;\rightarrow\; (\exists y_1,\ldots,y_n)\;\; P$
if $x \notin V(s) \cup V(P)$.

Fig. 1. The set of rules $\mathcal{S}$ for unification in simple syntactic theories

Before we show how the rules of $\mathcal{S}$ can be used, after all, for AC-unification, we introduce a convenient notation for AC-unification problems and we recall the well-known result [19, 21], about the "semantic" method for AC-unification.

Definition 5. *Consider the table:*

	x_1	$\cdots$	x_i	$\cdots$	x_n
t_1	a_{11}	$\cdots$	a_{1i}	$\cdots$	a_{1n}
t_2	a_{21}	$\cdots$	a_{2i}	$\cdots$	a_{2n}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
t_m	a_{m1}	$\cdots$	a_{mi}	$\cdots$	a_{mn}

where $x_1, \ldots, x_n \in \mathcal{X}$, $t_1, \ldots, t_m \in \mathcal{T}(\{+\}, \mathcal{X})$, and the a_{ij}s are natural numbers, and for $1 \leq i \leq n$, $\Sigma_{j=1}^{m} a_{ij} \neq 0$.
We shall use such a table as a notation for the unification problem:

$$P \equiv \bigwedge_{i=1}^{n} x_i \stackrel{?}{=} \underbrace{t_1 + \cdots + t_1}_{a_{1i} \text{ times}} + \cdots + \underbrace{t_m + \cdots + t_m}_{a_{mi} \text{ times}}$$

The admissible subproblems of P *are the problems corresponding to a subtable (in the sense that some lines have been removed), still verifying that the sum of the coefficients of each column is non-zero.*

Using the above notation we can reformulate the well-known result on AC-unification [19, 21].

Theorem 1. *Given a unification problem $s \stackrel{?}{=} t$, where s,t are terms of $\mathcal{T}(\{+\}, \mathcal{X})$, let $V(s,t) = \{x_1, \ldots, x_n\}$, and for $x_i \in V(s,t)$, let α_i be the number of occurrences of x_i in s minus its number of occurrences in t. Let $\{s_1, \ldots, s_m\}$ be the set of minimal (for the ordering $>^n$), positive, non-null solutions of the linear Diophantine equation $\Sigma_{i=1}^{n} \alpha_i z_i = 0$, where each s_j is a vector of integers $(a_{j1}, \ldots, a_{jn})$. Then, the admissible subproblems of*

	x_1	$\cdots$	x_i	$\cdots$	x_n
z_1	a_{11}	$\cdots$	a_{1i}	$\cdots$	a_{1n}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
z_j	a_{j1}	$\cdots$	a_{ji}	$\cdots$	a_{jn}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
z_m	a_{m1}	$\cdots$	a_{mi}	$\cdots$	a_{mn}

where $z_1, \ldots, z_m$ are new variables, form a complete set of solved forms of $s \stackrel{?}{=} t$.

3 Syntactic *AC*-Unification

fact[theorem]Fact assumption[theorem]Assumption

We take advantage of the fact that the unification problems $x + s \stackrel{?}{=} x + t$ and $s \stackrel{?}{=} t$ are AC-equivalent. Indeed, adding (or removing) an occurrence of a variable to (from) both sides of an equation does not change the coefficient of the corresponding integer variable in the associated linear Diophantine equation.

We assume that the input problem is an equation $s \stackrel{?}{=} t$ with $V(s) \cap V(t) = \emptyset$.

Definition 6. *Two unification problems P and Q are* strongly equivalent *if they are the same modulo*

- *the associativity and commutativity of $\wedge$ and $\vee$,*
- *the distributivity of $\wedge$ with respect to $\vee$,*
- *the commutativity of $\stackrel{?}{=}$,*
- *the associativity and commutativity of $+$,*
- *a renaming of the existentially quantified variables.*

Lemma 2. *Let $P \equiv x_1 + \cdots + x_n \stackrel{?}{=} y_1 + \cdots + y_m$ be an equation where $x_1, \ldots, x_n, y_1, \ldots, y_m$ are pairwise distinct variables.*
Assume that some rules of $\mathcal{S}$ are applied to P, yielding a problem Q. Assume that two rules $\mathcal{R}_1$ and $\mathcal{R}_2$ of $\mathcal{S}$, other than **Check*** *may be applied to Q, yielding the problems Q_1 and Q_2, respectively. Then there exist two strongly equivalent problems Q'_1 and Q'_2 which are respectively obtainable from Q_1 and Q_2 by applying one rule of $\mathcal{S}$.*

Proof. Let us first show by induction on the number of applications of the rules that every variable has at most two occurrences in the non-quasi-solved part of Q. The quasi-solved part of Q is of the form $v_1 \stackrel{?}{=} t_1 \ \wedge \ \ldots v_k \stackrel{?}{=} t_k$ where the v_is $(1 \leq i \leq k)$ are pairwise distinct and can only occur in some t_j, $j < i$, and nowhere else in Q. By hypothesis, this is true for P. For the inductive step, note first that the quasi-solved part is never made unsolved by an application of a rule.

- When **Mutate** is applied to a problem satisfying the property, the new variables introduced by the rule have exactly two occurrences, and the already existing variables keep the same number of occurrences.
- When **Merge** turns two equations $x \stackrel{?}{=} s$ and $x \stackrel{?}{=} t$ into $x \stackrel{?}{=} s$ and $s \stackrel{?}{=} t$, by induction hypothesis, x occurs nowhere else in the non-quasi-solved part. Hence the equation $x \stackrel{?}{=} s$ is now in the quasi-solved part, and the previous occurrences of the variables of s and t in $x \stackrel{?}{=} s$ and $s \stackrel{?}{=} t$ are now in the equation $s \stackrel{?}{=} t$.
- When **Rep** (or **Var-Rep**) is applied to $x \stackrel{?}{=} s \ \wedge \ P$ (or $x \stackrel{?}{=} y \ \wedge \ P$), the equation had to be in the non-quasi-solved part and it is now in the quasi-solved part. In both cases, by induction hypothesis, x had at most one occurrence in the non-quasi-solved part, and the new occurrences of the variables of s (or y) replace those in the equation $x \stackrel{?}{=} s$ (or $x \stackrel{?}{=} y$).
- Finally, the result is straightforward for **Check*** and **EQE** which do not create any new equation.

Now, the reader can easily check that neither Q_1 and Q_2 are in normal form, and two different applications of a rule to Q commute, modulo strong equivalence, since $\mathcal{S} \setminus \{\textbf{Check}^*\}$ does not contain any failure rule. The only non-straightforward case is two different possible applications of **Merge** on a same variable. This case can never occur, since if **Merge** can apply to $x \stackrel{?}{=} s$

and $x \stackrel{?}{=} t$, by the previous result, x has no further occurrence in Q, and this is the only possible application of **Merge** on x.

Definition 7. *The similarity is the reflexive, symmetric and transitive closure of the variables renaming and the variable abstraction $\equiv_{VA}$ defined on the unification problems:*

$$\exists \boldsymbol{v} P \equiv_{VA} \exists u, \boldsymbol{v} P\{y \mapsto u\} \ \wedge \ y \stackrel{?}{=} u$$

if $y \in \mathcal{V}ar(\exists \boldsymbol{v} P)$ and y does not appear in P as a member of an equation.

Lemma 3. *Let $P \equiv x_1 + \cdots + x_n \stackrel{?}{=} y_1 + \cdots + y_m$ be a linear equation with $n, m \geq 2$.*
The rules of $\mathcal{S}$ terminate, starting with P, and the disjuncts in the irreducible problem are (up to similarity) all the admissible subproblems of the problem:

	x_1	x_2	$\cdots$	x_n	y_1	y_2	$\cdots$	y_m
v_1	1	0	$\cdots$	0	1	0	$\cdots$	0
v_2	1	0	$\cdots$	0	0	1	$\cdots$	0
$\vdots$			$\cdots$				$\ddots$	
v_m	1	0	$\cdots$	0	0	0	$\cdots$	1
v_{m+1}	0	1	$\cdots$	0	1	0	$\cdots$	0
v_{m+2}	0	1	$\cdots$	0	0	1	$\cdots$	0
$\vdots$			$\cdots$				$\ddots$	
v_{2m}	0	1	$\cdots$	0	0	0	$\cdots$	1
$\cdots$			$\cdots$				$\cdots$	
$\cdots$			$\cdots$				$\cdots$	
$v_{(n-1)m+1}$	0	0	$\cdots$	1	1	0	$\cdots$	0
$v_{(n-1)m+2}$	0	0	$\cdots$	1	0	1	$\cdots$	0
$\vdots$			$\cdots$				$\ddots$	
v_{nm}	0	0	$\cdots$	1	0	0	$\cdots$	1

where the v_is are new variables.

Proof. We proceed by induction on the number of variables $n+m$ of the equation. If $n = m = 2$, **Mutate** yields exactly the admissible subproblems of

	x_1	x_2	y_1	y_2
v_1	1	0	1	0
v_2	1	0	0	1
v_3	0	1	1	0
v_4	0	1	0	1

By lemma 2, it suffices to show the result for a particular control. Assume $n > 2$ and $m \geq 2$.
Let us treat the equation $x_1 + \cdots + x_{n-2} + (x_{n-1} + x_n) \stackrel{?}{=} y_1 + \cdots + y_m$ like $x_1 + \cdots + x_{n-2} + x' \stackrel{?}{=} y_1 + \cdots + y_m$, that is considering $(x_{n-1} + x_n)$ as a variable. By induction hypothesis, we get admissible subproblems of

	x_1	x_2	$\cdots$	$(x_{n-1}+x_n)$	y_1	y_2	$\cdots$	y_m
v_1	1	0	$\cdots$	0	1	0	$\cdots$	0
v_2	1	0	$\cdots$	0	0	1	$\cdots$	0
$\vdots$			$\cdots$				$\ddots$	
v_m	1	0	$\cdots$	0	0	0	$\cdots$	1
v_{m+1}	0	1	$\cdots$	0	1	0	$\cdots$	0
v_{m+2}	0	1	$\cdots$	0	0	1	$\cdots$	0
$\vdots$			$\cdots$				$\ddots$	
v_{2m}	0	1	$\cdots$	0	0	0	$\cdots$	1
$\cdots$			$\cdots$				$\cdots$	
$\cdots$			$\cdots$				$\cdots$	
$v'_{(n-2)m+1}$	0	0	$\cdots$	1	1	0	$\cdots$	0
$v'_{(n-2)m+2}$	0	0	$\cdots$	1	0	1	$\cdots$	0
$\vdots$			$\cdots$				$\ddots$	
$v'_{(n-1)m}$	0	0	$\cdots$	1	0	0	$\cdots$	1

Now, all the equations are solved, except maybe an equation between $(x_{n-1}+x_n)$ and a subterm of $v'_{(n-2)m+1}+v'_{(n-2)m+2}+\cdots+v'_{(n-1)m}$. By induction hypothesis the rules applied to this equation yield the admissible subproblems of

	x_{n-1}	x_n	$v'_{(n-2)m+1}$	$v'_{(n-2)m+2}$	$\cdots$	$v'_{(n-1)m}$
$v_{(n-2)m+1}$	1	0	1	0	$\cdots$	0
$v_{(n-2)m+2}$	1	0	0	1		0
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\ddots$	$\vdots$
$v_{(n-1)m}$	1	0	0	0		1
$v_{(n-1)m+1}$	0	1	1	0	$\cdots$	0
$v_{(n-1)m+2}$	0	1	0	1		0
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\ddots$	$\vdots$
v_{nm}	0	1	0	0		1

Now, applying **Rep** and **EQE** to $v'_{(n-2)m+1}, v'_{(n-2)m+1}, \ldots, v'_{(n-1)m}$ yields some admissible subproblems of the problem given in the lemma. The set of rules being complete (*i.e.* all the minimal solutions are constructed), all the admissible subproblems are obtained since they form a minimal complete set of AC-unifiers as shown in theorem 1.

3.1 A new (inefficient) algorithm for solving a linear Diophantine equation

The following is the key theorem. Indeed, we have seen that when a non-linear equation is treated like a linear one, a problem is obtained, where the new variables correspond to the solutions of the trivial Diophantine equation where all the coefficients are 1. Now, a solution of the linear Diophantine equation associated with a non-linear AC-equation can be obtained as the sum of a *subset* of the solutions of the trivial Diophantine equation of the linearized AC equation:

Theorem 2. *Let e be the linear Diophantine equation*

$$a_1x_1 + \cdots + a_nx_n = b_1y_1 + \cdots + b_my_m$$

where the a_is and b_js are positive integers. Let us write e in the form

$$\underbrace{x_1 + \cdots + x_1}_{a_1 \text{ times}} + \cdots + \underbrace{x_n + \cdots + x_n}_{a_n \text{ times}} = \underbrace{y_1 + \cdots + y_1}_{b_1 \text{ times}} + \cdots + \underbrace{y_m + \cdots + y_m}_{b_m \text{ times}}$$

and let us write a solution $(c_1, \ldots, c_n, c_{n+1}, \ldots, c_{n+m})$ under the form

$$(\underbrace{c_1, \ldots, c_1}_{a_1 \text{ times}}, \ldots, \underbrace{c_n, \ldots, c_n}_{a_n \text{ times}}, \underbrace{c_{n+1}, \ldots, c_{n+1}}_{b_1 \text{ times}}, \ldots, \underbrace{c_{n+m}, \ldots, c_{n+m}}_{b_m \text{ times}})$$

Let $\mathcal{E}$ be the set of $(\Sigma_{i=1}^n a_i + \Sigma_{j=1}^m b_j)$-tuples of integers of the form

$$(0, \ldots, 0, 1, 0, \ldots, 0, 1, 0, \ldots, 0)$$

where all the components are 0, except on the kth and lth components which are 1, with $1 \leq k \leq \Sigma_{i=1}^n a_i$ and $\Sigma_{i=1}^n a_i < l \leq \Sigma_{i=1}^n a_i + \Sigma_{j=1}^m b_j$).

The minimal solutions of e are the componentwise sums of the minimal (wrt inclusion) subsets of $\mathcal{E}$, which have a same value in each component corresponding to a same variable.

Proof. First, every componentwise sum of a subset of $\mathcal{E}$ which agrees on different components correponding to a same variable is a solution since each vector of $\mathcal{E}$ adds 1 to both sides of the equation. Second, every greater (wrt inclusion) subset satisfying the same condition will obviously correspond to a non-minimal solution.

Let us write a minimal solution c of e in the form

$$x_1 = s^{c_1}(0), \ldots, x_n = s^{c_n}(0), y_1 = s^{c_{n+1}}(0), \ldots, y_m = s^{c_{n+m}}(0)$$

Since c is a solution, there is a one-to-one mapping β between the occurrences of the symbol s in

$$\underbrace{s^{c_1}(0) + \cdots + s^{c_1}(0)}_{a_1 \text{ times}} + \cdots + \underbrace{s^{c_n}(0) + \cdots + s^{c_n}(0)}_{a_n \text{ times}}$$

and in $\quad \underbrace{s^{c_{n+1}}(0) + \cdots + s^{c_{n+1}}(0)}_{b_1 \text{ times}} + \cdots + \underbrace{s^{c_{n+m}}(0) + \cdots + s^{c_{n+m}}(0)}_{b_m \text{ times}}$

Let β_{ij} be the number of occurrences of the symbol s in an x_i which are mapped by β onto an occurrence of the symbol s in an y_j. Since c is a minimal solution, it is not greater (for $>^{n+m}$) than the solution

$$\begin{array}{ccccccc} (0, \ldots, 0, & b_j & , 0, \ldots, 0, & & a_i & & , 0, \ldots, 0) \\ & \uparrow & & & \uparrow & & \\ & i\text{-th} & & & (n+j)\text{-th} & & \end{array}$$

hence $\beta_{ij} \leq a_i \times b_j$. But there are exactly $a_i \times b_j$ vectors of $\mathcal{E}$ which have the value 1 in a component corresponding to an x_i and in a component corresponding to an y_j. For every (x_i, y_j), let $\mathcal{E}_{ij}$ be a set of β_{ij} such vectors. Now c is the componentwise sum of $\bigcup_{ij} \mathcal{E}_{ij}$.

3.2 The criterion

Theorem 3. *Let $P \equiv s \stackrel{?}{=} t$ be a unification problem, with $V(s) \cap V(t) = \emptyset$. Assume that in a first step the rules of $\mathcal{S}$ are applied to P as long as possible, yielding Q, except that two distinct occurrences of a variable in $V(s) \cup V(t)$ are considered as distinct variables. (By lemma 3, this first step terminates.)*
Let $\mathcal{V}$ be the set $V(Q) \setminus V(P)$ of variables introduced by this first step. Assume now that the rules of $\mathcal{S}$ are now applied to Q with the additional rule:

Prune
$x \stackrel{?}{=} t \quad \rightarrow \; F$
if $x \in \mathcal{V}$ and t is not a linear term.

The process described above is still complete, i.e. *every solved form of a complete set of solved forms of P will be computed.*

In other words, if in a first step **Merge** is never applied to the variables of the input equation, in a second step, the equations built by **Merge** can be solved while discarding the non-linear substitutions for the variables introduced at the first step.

Proof. We develop an example (using this font) in parallel with the proof.
Let $P \equiv x_1 + \cdots + x_n \stackrel{?}{=} y_1 + \cdots + y_m$ be an equation where the x_is (and the y_js) are not necessarily pairwise different variables.

Our example will be $P \equiv x + x + x \stackrel{?}{=} y + y + u$.

By lemma 3, the first step terminates and the disjuncts in the irreducible problem are all the admissible subproblems of the problem:

$Q \equiv$

	x_1	x_2	$\cdots$	x_n	y_1	y_2	$\cdots$	y_m
v_1	1	0	$\cdots$	0	1	0	$\cdots$	0
v_2	1	0	$\cdots$	0	0	1	$\cdots$	0
$\vdots$			$\cdots$				$\ddots$	
v_m	1	0	$\cdots$	0	0	0	$\cdots$	1
v_{m+1}	0	1	$\cdots$	0	1	0	$\cdots$	0
v_{m+2}	0	1	$\cdots$	0	0	1	$\cdots$	0
$\vdots$			$\cdots$				$\ddots$	
v_{2m}	0	1	$\cdots$	0	0	0	$\cdots$	1
$\cdots$			$\cdots$				$\cdots$	
$\cdots$			$\cdots$				$\cdots$	
$v_{(n-1)m+1}$	0	0	$\cdots$	1	1	0	$\cdots$	0
$v_{(n-1)m+2}$	0	0	$\cdots$	1	0	1	$\cdots$	0
$\vdots$			$\cdots$				$\ddots$	
v_{nm}	0	0	$\cdots$	1	0	0	$\cdots$	1

where several occurrences of a same variable may occur in $x_1, \ldots, x_n$ and $y_1, \ldots, y_m$.

In the example, we have

$$Q \equiv \begin{array}{c|cccccc} & x & x & x & y & y & z \\ \hline v_1 & 1 & 0 & 0 & 1 & 0 & 0 \\ v_2 & 1 & 0 & 0 & 0 & 1 & 0 \\ v_3 & 1 & 0 & 0 & 0 & 0 & 1 \\ v_4 & 0 & 1 & 0 & 1 & 0 & 0 \\ v_5 & 0 & 1 & 0 & 0 & 1 & 0 \\ v_6 & 0 & 1 & 0 & 0 & 0 & 1 \\ v_7 & 0 & 0 & 1 & 1 & 0 & 0 \\ v_8 & 0 & 0 & 1 & 0 & 1 & 0 \\ v_9 & 0 & 0 & 1 & 0 & 0 & 1 \end{array}$$

By theorem 1, a complete set of solved forms of P is made of problems of the form

$$S \equiv \begin{array}{c|ccccccc} & x_1 & \cdots & x_n & & y_1 & \cdots & y_m \\ \hline z_1 & c_{11} & \cdots & c_{1n} & c_{1(n+1)} & & \cdots & c_{1(n+m)} \\ z_2 & c_{21} & \cdots & c_{2n} & c_{2(n+1)} & & \cdots & c_{2(n+m)} \\ \cdots & \cdots & \cdots & \cdots & \cdots & & \cdots & \cdots \\ z_p & c_{p1} & \cdots & c_{pn} & c_{p(n+1)} & & \cdots & c_{p(n+m)} \end{array}$$

where for $1 \leq i \leq p$, $s_i = (c_{i1}, \ldots, c_{i(n+m)})$ is a minimal positive, non-null solution of the linear Diophantine equation e associated with P, and the componentwise sum of the coefficients in each column is non-zero. (Here the columns corresponding to the different occurrence of a same variable have been duplicated).

In the example, we consider the solution obtained from the minimal solutions $(2, 3, 0)$ and $(1, 1, 1)$ of the Diophantine equation $3\alpha = 2\beta + \gamma$. The table

$$S \equiv \begin{array}{c|cccccc} & x & x & x & y & y & z \\ \hline z_1 & 2 & 2 & 2 & 3 & 3 & 0 \\ z_2 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}$$

corresponds to the unifier $\{x \mapsto z_1 + z_1 + z_2, y \mapsto z_1 + z_1 + z_1 + z_2, z \mapsto z_2\}$.

The lines of Q being the vectors of the set $\mathcal{E}$ described in theorem 2, each s_i (written as an $n + m$-uple) is the componentwise sum of the lines correponding to a subset $\mathcal{V}(s_i)$ of $\{v_1, \ldots, v_{nm}\}$.

In the example, $s_1 = (2, 2, 2, 3, 3, 0)$, and $s_2 = (1, 1, 1, 1, 1, 1)$.
For instance, we choose $\mathcal{V}(s_1) = \{v_1, v_2, v_4, v_5, v_7, v_8\}$ and $\mathcal{V}(s_2) = \{v_1, v_5, v_9\}$.
Note that for $\mathcal{V}(s_2)$, we could as well have chosen the subset $\{v_3, v_4, v_8\}$.

For $1 \leq i \leq n \times m$, let $\mathcal{W}(v_i) = \{z_j \in \{z_1, \ldots, z_p\} \mid v_i \in \mathcal{V}(s_j)\}$.

In the example, $\mathcal{W}(v_3) = \mathcal{W}(v_6) = \emptyset$, $\mathcal{W}(v_1) = \mathcal{W}(v_5) = \{z_1, z_2\}$, $\mathcal{W}(v_2) = \mathcal{W}(v_4) = \mathcal{W}(v_7) = \mathcal{W}(v_8) = \{z_1\}$ and $\mathcal{W}(v_9) = \{z_2\}$.

Let us call *useful* the variables v_i of $\{v_1, \ldots, v_{nm}\}$ such that $\mathcal{W}(v_i) \neq \emptyset$. For each useful variable v_i, let t_i be a term of $\mathcal{T}(\{+\}, \mathcal{X})$ such that $V(t_i) = \mathcal{W}(v_i)$, each variable having one occurrence. The subproblem Q' obtained by keeping the lines of Q correponding to useful variables is, by construction, an admissible subproblem of Q.

In the example, The variables that are not useful are v_3 and v_6, and Q' is obtained from Q by removing the two corresponding lines.
We have $t_1 = t_5 = z_1 + z_2$, $t_2 = t_4 = t_7 = t_8 = z_1$ and $t_9 = z_2$.

By construction, the values for the input variables in the problem obtained from Q' by replacing each v_i by t_i are equal modulo AC as those in S. Note that the t_is are linear terms because the minimal solutions of the linear Diophantine equation are the sum of a *subset* of the lines of Q, and not of a multiset.

In the example,we obtain the problem

	x	x	x	y	y	z
$z_1 + z_2$	1	0	0	1	0	0
z_1	1	0	0	0	1	0
z_1	0	1	0	1	0	0
$z_1 + z_2$	0	1	0	0	1	0
z_1	0	0	1	1	0	0
z_1	0	0	1	0	1	0
z_2	0	0	1	0	0	1

At this point, we have shown that the equations created by applying **Merge** to the variables of the original problem can be solved while restricting the values of the new variables of $\mathcal{V}$ to linear terms, without loosing the completeness. Since the rules of $\mathcal{S}$ are complete, these linear solutions will be computed, hence the result.

3.3 An algorithm for "syntactic" AC-unification

We give a control, using our criterion for pruning some branches which might not terminate otherwise, and show that each step terminates with our control.
To solve the equation $s \stackrel{?}{=} t$, where s and t have no common variable

1. Apply as long as possible the rules of $\mathcal{S}$ except that two distinct occurrences of a variable in $V(s) \cup V(t)$ are considered as distinct. By lemma 3, this step terminates, and as we have seen in the proof of lemma 2, there are at most two occurrences of each new variable.
2. As long as possible, apply (**Var-Rep**)* in order to make possibl;e an application of **Merge** to a variable of $V(s) \cup V(t)$ and solve the resulting equation. The equation $s \stackrel{?}{=} t$ created by applying **Merge** to $x \stackrel{?}{=} s \ \wedge \ x \stackrel{?}{=} t$ is linear, as shown by lemma 3. Hence, this step terminates.

3. Apply **Prune** to the equations $v \stackrel{?}{=} t$ where $v \notin V(s) \cup V(t)$ and t is not linear, and apply **Check*** if possible. (This can be done in parallel with the previous step). This step obviously terminates. Now, for every variable $v \notin V(s) \cup V(t)$, we have at most two equations $v \stackrel{?}{=} s$ and $v \stackrel{?}{=} t$.
4. As long as possible, apply **Merge** to $v \stackrel{?}{=} s$ and $v \stackrel{?}{=} t$, with $v \notin V(s) \cup V(t)$, solve the resulting equation $s \stackrel{?}{=} t$, apply **Rep** to the equations of the solved form, and apply **Prune** and **Check*** if possible. This step terminates, by lemma 3 since every equation that will be solved is linear, and the number of possible applications of **Merge** decreases.

4 Conclusion

We have given the first AC-unification algorithm based on the use of the resolvent presentation of AC. Again, we do not claim that this algorithm can compete, regarding efficiency, with the existing algorithms based on the solving of linear Diophantine equations. However, our method might turn out to have some advantages

- It is easy to avoid computing some solutions that are symmetric, or to have a compact representation of symmetric solutions. For instance, when computing the solutions of $x + y \stackrel{?}{=} t$, one can avoid to apply the commutativity at the root, and represent sets of substitutions under the form

$$\{\{x, y\} \mapsto \{u_1, u_2\}, \ldots\}$$

  standing for $\{\{x \mapsto u_1, y \mapsto u_2, \ldots\}, \{x \mapsto u_2, y \mapsto u_1, \ldots\}\}$.
- We do not know what would be in practice the efficiency of our method for computing just one solution, rather than a complete set. This would be worth experimenting.
- Our algorithm is easy to implement: it does not require to have a linear Diophantine equations solver, nor to search the subsets of the set of minimal solutions. It may be advantageous to use it, for instance in a prototype.

Some problems remain:

- Some solutions are not minimal: The reader can check that our algorithm, applied to $x + x \stackrel{?}{=} y + z$, using the axiom MC twice, yields the solution $\{x \mapsto v_1 + v_2 + v_3 + v_4, y \mapsto v_1 + v_1 + v_2 + v_3, z \mapsto v_2 + v_3 + v_4 + v_4\}$, which is an instance of $\{x \mapsto v_1 + v + v_4, y \mapsto v_1 + v_1 + v, z \mapsto v + v_4 + v_4\}$.
- Some solutions are computed several times: for instance when two variables simultaneously introduced by **Mutate** are later identified.
- Is it possible to relax the control, and to apply **Prune** as soon as a new variable gets a non-linear value? This is our conjecture.

References

1. Mohamed Adi and Claude Kirchner. AC-unification race: the system solving approach. In *Proc. Int. Symposium on Design and Implementation of Symbolic Computation Systems, LNCS 429*, 1990.
2. Alexandre Boudet. Competing for the AC-unification race. *Journal of Automated Reasoning*, 11:185–212, 1993.
3. Alexandre Boudet, Evelyne Contejean, and Hervé Devie. A new AC-unification algorithm with a new algorithm for solving diophantine equations. In *Proc. 5th IEEE Symp. Logic in Computer Science, Philadelphia*, June 1990.
4. Hans-Jurgen Bürckert, Alexander Herold, Deepak Kapur, Jorg H. Siekmann, Mark E. Stickel, Michael Tepp, and Hantao Zhang. Opening the AC-unification race. *Journal of Automated Reasoning*, 4(4):465–474, December 1988.
5. Jim Christian and Patrick Lincoln. Adventures in associative-commutative unification. Technical Report ACA-ST-275-87, MCC, AI Program Austin, Austin, October 1987.
6. Hubert Comon, Marianne Haberstrau, and Jean-Pierre Jouannaud. Decidable properties of shallow equational theories. In *Proc. 7th IEEE Symp. Logic in Computer Science*, Santa Cruz, 1992.
7. Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 243–309. North-Holland, 1990.
8. Eric Domenjoud. AC unification through order-sorted AC1 unification. *Journal of Symbolic Computation*, 14(6):537–556, December 1992.
9. François Fages. Associative-commutative unification. *Journal of Symbolic Computation*, 3(3), June 1987.
10. A. Fortenbacher. An algebraic approach to unification under associativity and commutativity. In *Proc. Rewriting Techniques and Applications 85, Dijon, LNCS 202*. Springer-Verlag, May 1985.
11. M. Franzen and L.J. Henschen. A new approach to universal unification and its application to AC-unification. In *Lusk and Overbeek eds., Proc. 9th int. Conf. on Automated Deduction*. Springer-Verlag, May 1988.
12. Alexander Herold and Jorg H. Siekmann. Unification in abelian semi-groups. *Journal of Automated Reasoning*, 3(3):247–283, 1987.
13. Jean-Pierre Jouannaud and Claude Kirchner. Solving equations in abstract algebras: A rule-based survey of unification. In Jean-Louis Lassez and Gordon Plotkin, editors, *Computational Logic: Essays in Honor of Alan Robinson*. MIT-Press, 1991.
14. D. Kapur and P. Narendran. Double-exponential complexity of computing a complete set of *ac*-unifiers. In *Proc. 7th IEEE Symp. Logic in Computer Science, Santa Cruz*, June 1992.
15. Claude Kirchner. Méthodes et outils de conception systématique d'algorithmes d'unification dans les théories equationnelles. Thèse d'Etat, Univ. Nancy, France, 1985.
16. Claude Kirchner. Computing unification algorithms. In *Proc. 1st IEEE Symp. Logic in Computer Science, Cambridge, Mass.*, pages 206–216, 1986.
17. Claude Kirchner and Francis Klay. Syntactic theories and unification. In *Proc. 5th IEEE Symp. Logic in Computer Science, Philadelphia*, June 1990.
18. Francis Klay. Undecidable properties of syntactic theories. In *Proc. 4th Rewriting Techniques and Applications, LNCS 488*, Como, Italy, 1991.

19. M. Livesey and Jorg H. Siekmann. Unification of bags and sets. Research report, Institut fur Informatik I, Universität Karlsruhe, West Germany, 1976.
20. T. Nipkow. Proof transformations for equational theories. In *Proc. 5th IEEE Symp. Logic in Computer Science, Philadelphia*, June 1990.
21. M. Stickel. A unification algorithm for associative-commutative functions. *Journal of the ACM*, 28(3):423–434, 1981.
22. Erik Tiden and Stefan Arnborg. Unification problems with one-sided distributivity. *Journal of Symbolic Computation*, 3:183–202, 1987.