J. Symbolic Computation (1996) **22**, 49–82

AC Complement Problems: Satisfiability and Negation Elimination

MARIBEL FERNÁNDEZ[†‡]

DMI - LIENS (CNRS URA 1327),
École Normale Supérieure, 45, rue d'Ulm, 75005 Paris, France.

(Received 24 January 1994)

In general, functional programs are assumed to be non-ambiguous. A simple way to handle ambiguous definitions is to replace them by non-ambiguous ones containing complement problems, which in turn have to be transformed into explicit expressions. Finding an explicit formulation for a complement problem is the same as eliminating negation from the corresponding equational formula. This problem was already shown decidable for complement problems interpreted in $T(\mathcal{F})$, and in $T(\mathcal{F})/_{=_E}$ when E is a permutative theory of Mal'cev. Here, we show that negation elimination is also decidable for linear complement problems interpreted in $T(\mathcal{F})/_{=_{AC}}$, where AC is a set of associative and commutative axioms. We present a system of rules that transforms any linear complement problem into simple formulae, and we give a decision test for simple formulae which serves as a basis for the development of a negation elimination algorithm. In the case of non-linear AC-complement problems even satisfiability is an open problem, but we will show, using the previous results, that satisfiability is decidable for a particular class of non-linear problems: when variables do not occur directly below AC-function symbols. Finally, we generalize the negation elimination results to a wider class of equational formulae modulo AC.

1. Introduction

In general, a functional program is assumed to be a set of left-linear and non-ambiguous definitions. Ambiguity causes problems not only for the definition of efficient evaluators (for instance, the decision algorithms for strong sequentiality apply only to non-ambiguous programs) but also for deciding other properties of programs, such as confluence. An easy way to handle this problem is to replace redundant definitions with non-ambiguous definitions containing *complement problems*. For example, if $l_1 \to r_1$ and $l_2 \to r_2$ are ambiguous rules in a functional program, i.e. there are instances of l_2 that are also instances of l_1, we can replace the second rule by $([\![l_2]\!] - [\![l_1]\!]) \to r_2$, which is no longer ambiguous since the complement problem $[\![l_2]\!] - [\![l_1]\!]$ represents the set of ground instances of l_2 that are not instances of l_1.

Complement problems arise also in other domains in computer science. In the process

† Part of this work was done while the author was at LRI, Université de Paris Sud.
‡ E–mail: maribel@dmi.ens.fr

0747–7171/96/070049 + 34 $25.00/0

of inductive learning (i.e. the process of learning from facts provided by the environment) generalization and term complement are key operations: given a set of terms (or examples) there exists a unique (up to variable renaming) term that represents their most specific generalization; however, usually one needs to restrict this generalization by using counter-examples, which give rise to complement problems. For example, the most specific generalization of the terms $0+1$, $0+2$, $1+2$ is $x+y$ where x and y are variables, but a better generalization is the complement $[\![x+y]\!] - [\![z+z]\!]$, which represents the set of ground instances of $x+y$ such that $x \neq y$, or, in other words, the set of ground instances of $x+y$ which are not instances of $z+z$.

Other areas where complement problems have important applications are: algebraic specifications [the problem of sufficient completeness is nothing but solving a complement problem (Comon, 1988; Thiel, 1984)], logic programming [in particular for the implementation of constructive negation (Lassez *et al.*, 1991)] and constraint programming [to represent solutions of systems of equations and disequations (Lassez and Marriott, 1987)]. Lassez and Marriott (1987) presented the first comprehensive study of complement problems, and showed that it is easy to decide whether a complement problem has solutions or not. See Lassez *et al.* (1991) for a survey of applications of complement problems.

Actually, as remarked by Comon (1988), the computation of term complement is a particular case of solving equational formulae. An *equational formula* is a first order formula constructed over a finite alphabet $\mathcal{F}$ of function symbols and only one relational symbol: equality. The complement problem $[\![t]\!] - [\![t_1, \ldots, t_n]\!]$, which represents the set of ground instances of t that are not instances of $t_1, \ldots, t_n$, is equivalent to the equational formula

$$\forall \vec{y} : t \neq t_1 \wedge \ldots \wedge t \neq t_n$$

(where $\vec{y}$ are the variables appearing in $t_1, \ldots, t_n$) interpreted in the *ground term algebra* $T(\mathcal{F})$. Comon (1988) presented an algorithm for deciding whether an equational formula has solutions in $T(\mathcal{F})$, which solves, in particular, complement problems. The disunification algorithm of Buntine and Bürckert (1994), which solves systems of equations and disequations in the *algebra* $T(\mathcal{F}, \mathcal{X})$ *of terms with variables*, cannot be applied here since by definition complement problems deal with ground instances of terms, i.e. have to be solved in $T(\mathcal{F})$. Note that the behaviour of these two algebras is very different: for instance, when E-unification is decidable and finitary, E-disunification is decidable in the quotient of $T(\mathcal{F}, \mathcal{X})$ modulo E (Buntine and Bürckert, 1994) whereas it is not decidable in the quotient of $T(\mathcal{F})$ modulo E (Fernandez, 1992). Actually, the algorithm of Buntine and Bürckert can be used in any algebra for which a consistency test exists but they gave such a test only for $T(\mathcal{F}, \mathcal{X})$ or its quotients; testing consistency in $T(\mathcal{F})$ is equivalent to solving complement problems.

Complement problems serve, in general, as a means of representing knowledge, but this way of expressing knowledge has an important drawback: it is not *explicit*. The transformation of a complement problem into an equivalent explicit representation, that is, an equivalent finite disjunction of terms, is a very interesting problem. In particular, with this technique ambiguity can be eliminated from functional programs, or from rewrite systems in general. Other applications are described in Lassez *et al.* (1991). Finding a finite explicit representation for a complement problem is the same as *eliminating negation* from the corresponding equational formula. As Lassez and Marriott (1987) showed, this problem is also decidable in $T(\mathcal{F})$: the complement problem $\forall \vec{y} : t \neq t_1 \wedge \ldots \wedge t \neq t_n$

is equivalent to a formula without negation if and only if for $1 \leq i \leq n$, the most general unifier of t and t_i is linear in the variables of t. Hence, there is a simple sufficient and necessary condition for deciding negation elimination when complement problems are interpreted in $T(\mathcal{F})$.

However, as pointed out by Kounalis (1990), Kounalis *et al.* (1991), and Lugiez and Moysset (1993), in most applications of complement problems the involved function symbols are not free. There exists, in general, some "background knowledge" that provides information about the properties of the operators, e.g. we have to consider axioms that express the semantics of the operators, and we have to interpret complement problems in the quotient of $T(\mathcal{F})$ by the equational theory E defined by these axioms (the quotient will be denoted by $T(\mathcal{F})/_{=_E}$).

In Comon and Fernández (1992) we showed that when E is a permutative theory of Mal'cev (1971), negation elimination is decidable for complement problems interpreted in $T(\mathcal{F})/_{=_E}$. Permutative theories of Mal'cev include and generalize commutative theories, but do not include the important case of theories with associative and commutative axioms (AC). AC operators appear very often in applications (notorious examples are the logical connectives $\wedge$, $\vee$, and many algebraic functions), and this motivated the study of satisfiability of AC complement problems (i.e. complement problems interpreted in $T(\mathcal{F})/_{=_{AC}}$) in Kounalis *et al.* (1991) and Lugiez and Moysset (1993). Here we continue that line of research, studying *negation elimination in AC complement problems*, and as a first step towards that, we also develop an algorithm for *solving AC complement problems*. Our results provide a partial answer to the open questions of Buntine and Bürckert (1994) about testing "ground consistency" and representing solutions of E-disunification problems by substitutions only, in the AC-case.

The necessary and sufficient condition of Lassez and Marriott for negation elimination in $T(\mathcal{F})$ that we mentioned above, generalizes easily to $T(\mathcal{F})/_{=_E}$ when E is a permutative theory of Mal'cev (by taking E-unifiers instead of syntactical unifiers), but in the case of $T(\mathcal{F})/_{=_{AC}}$ the condition is no longer sufficient. For example, if $+$ is AC, a is a constant and f is a free unary function symbol, negation cannot be eliminated from $\forall y : x \neq y + a$ although the most general unifier of the terms is linear.

In this paper we show that negation elimination is decidable for *linear* complement problems interpreted in $T(\mathcal{F})/_{=_{AC}}$. A complement problem $[\![t]\!] - [\![t_1, \ldots, t_n]\!]$ is linear if $t_1, \ldots, t_n$ do not contain multiple occurrences of the same variable. This is an interesting class of problems, since in particular the problems one has to solve in order to eliminate the ambiguity of functional programs are linear.

For *non-linear* problems the decidability of negation elimination is still an open question, and even the decidability of satisfiability is open. We will use our results about negation elimination to develop a decision algorithm for testing the satisfiability of a restricted class of non-linear complement problems: the class of problems where variables do not occur directly below AC-function symbols.

As a first step towards the definition of a decision algorithm for negation elimination, we develop an algorithm for solving linear AC-complement problems. Two different algorithms have already been proposed for solving this class of formulae. Kounalis *et al.* (1991) presented an algorithm based on *test sets*, the idea is that in order to decide whether a complement problem has solutions it is enough to test some particular terms (a finite set). The other algorithm, presented by Lugiez and Moysset (1993), is based on a completely different concept. The authors show first that the set of ground AC-instances of a linear term t is a regular tree language (this is done by constructing a bottom-up

tree automaton which recognizes the set of ground AC-instances of t). Then they show that there is a tree automaton that recognizes the solutions of the complement problem, and therefore it is sufficient to decide the emptiness of this language to decide the satisfiability of the problem.

Our algorithm is related to the first one, but the approach is different: it is inspired by the ideas presented by Comon (1988) and Comon and Fernández (1992), where algorithms for solving complement problems in $T(\mathcal{F})$ are expressed by means of *transformation rules*. As a first step towards solving AC complement problems (and eliminating negation), we give a set of transformation rules over equational formulae, which reduces any complement problem to a finite disjunction of $\forall$-*simple formulae*. These equational formulae correspond to the constrained unifiers of Buntine and Bürckert (1994). Then, we show how to test whether a $\forall$-simple formula has solutions. The decision algorithm for $\forall$-simple formulae is based on test sets, as the algorithm presented by Kounalis *et al.* (1991) but instead of applying the test to the original complement problem as they do, we apply the test to the $\forall$-simple formulae that result from the transformation, decreasing in this way the size of the set of terms used in the test. On the other side, using our method the test has to be applied to each $\forall$-simple formula in the disjunction, which gives a similar complexity in the end.

The algorithm for negation elimination is developed by stepwise refinement of the decision test for satisfiability, using the properties of the $\forall$-simple formulae obtained after the transformation process. Finally, we show that the same techniques can be applied to decide satisfiability of a restricted case of non-linear AC-complement problems.

The satisfiability test for $\forall$-simple formulae can also be used to extend the E-disunification algorithm of Buntine and Bürckert (1994): This algorithm relies on a consistency test for constrained unifiers, which was given only for $T(\mathcal{F}, \mathcal{X})$ or its quotient modulo E. Since constrained unifiers are the same as $\forall$-simple formulae, and consistency is the same as satisfiability, we have now a consistency test for constrained unifiers in $T(\mathcal{F})/_{=_{AC}}$, hence an AC-disunification algorithm. See Buntine and Bürckert (1994) for more details and applications of disunification algorithms.

The paper is organized as follows†: Section 2 contains a concise description of the syntax and semantics of complement problems; Section 3 deals with satisfiability and negation elimination in AC complement problems: transformation rules and decision tests; in Sections 4 and 5 we generalize the algorithms to a wider class of equational formulae.

2. AC Complement Problems

2.1. SYNTAX

Our notations are consistent with those of Dershowitz and Jouannaud (1990), where the reader can find supplementary definitions and examples. We consider a finite set $\mathcal{F}$ of function symbols with fixed arities. $T(\mathcal{F})$ is the infinite set of *ground terms* over this alphabet. Terms are identified with finite labeled trees as usual. Positions are strings of positive integers. The symbol at position p is denoted $t(p)$, the subterm of t at position p is denoted $t|_p$ and the result of replacing $t|_p$ with u at position p in t is denoted $t[u]_p$. Variables are an infinite distinguished set $\mathcal{X}$ of symbols. A sequence $x_1, \ldots, x_n$ of variables

† A short version of this paper appeared in Fernández (1993).

will be abbreviated as $\vec{x}$. The set of *terms* built on $\mathcal{F}$ and $\mathcal{X}$ is denoted $T(\mathcal{F},\mathcal{X})$. $Var(t)$ denotes the set of variables appearing in t. A term t is *linear* if each variable appears at most once in t. A set S of terms is *linear* if each variable occurs at most once in S. The symbol $\equiv$ denotes identity of objects. If S is a set, $|S|$ will denote its cardinal.

DEFINITION 2.1. *An* equation *is an unordered pair of terms, written $s = t$. A* disequation *is also an unordered pair of terms, written $s \neq t$. An* equational formula *is a first order formula whose atoms are equations or disequations.*

For simplicity, we will always assume that each variable is bound at most once in a formula and cannot occur both free and bound in the same formula.

We are interested in two particular fragments of equational formulae: complement problems and unification problems.

DEFINITION 2.2. *A* complement problem *is either $\top$, $\bot$ or a formula $\forall\vec{y} : t \neq t_1 \wedge \ldots \wedge t \neq t_n$ where $Var(t) = \vec{x}$ is the set of free variables (or unknowns) of the formula and $Var(t_1) \cup \ldots \cup Var(t_n) = \vec{y}$. Without loss of generality we assume that $Var(t_i) \cap Var(t_j) = \emptyset$ for all $i \neq j$, then, a complement problem is* linear *if for $1 \leq i \leq n$, t_i does not contain multiple occurrences of the same variable.*

DEFINITION 2.3. *A* unification problem *is an existential formula which does not contain any disequation.*

2.2. SEMANTICS

Given an $\mathcal{F}$-algebra $\mathcal{A}$, an *$\mathcal{A}$-assignment* is a mapping σ which associates to each variable an element of $\mathcal{A}$. Such mappings are lifted to $\mathcal{F}$-homomorphisms from $T(\mathcal{F},\mathcal{X})$ to $\mathcal{A}$ in the usual way, and used in postfix notation. When $\mathcal{A}$ is $T(\mathcal{F},\mathcal{X})$ itself, and when the *domain* of σ ($Dom(\sigma) = \{x \in \mathcal{X} \mid x\sigma \not\equiv x\}$) is finite, σ is called a *substitution*, and written $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$. When $\mathcal{A} = T(\mathcal{F})$, σ is called a *ground assignment*. By $\sigma|_V$ we denote the restriction of a substitution or a ground assignment σ to the set V of variables. The set of all substitutions (resp. ground assignments) is denoted by Σ (resp. Σ_g). The *image* of a substitution σ is the set $Im(\sigma) = \{x\sigma \mid x \in Dom(\sigma)\}$. A substitution is *idempotent* iff $\sigma\sigma = \sigma$. The relation $\preceq$ on substitutions is defined as usual: $\gamma \preceq \mu$ if there exists ρ such that $\sigma\rho = \mu$.

DEFINITION 2.4. *The set of* solutions *of an equational formula ϕ is the set of ground assignments σ such that $T(\mathcal{F}) \models \phi\sigma$. It is denoted by $[\![\phi]\!]$. A formula is* satisfiable *if it has solutions.*

We consider interpretations in quotient algebras $T(\mathcal{F})/_{=_{AC}}$ where $=_{AC}$ is the smallest congruence on $T(\mathcal{F},\mathcal{X})$ such that it is closed by instantiation and contains the set of equations $\{f(x,y) = f(y,x) \mid f \in \mathcal{F}'\} \cup \{f(x,f(y,z)) = f(f(x,y),z) \mid f \in \mathcal{F}'\}$ for a subset $\mathcal{F}'$ of $\mathcal{F}$ containing only binary symbols. The symbols in $\mathcal{F}'$ are called AC function symbols and the theory of $=_{AC}$ is called AC equational theory.

DEFINITION 2.5. *An AC-*solution *of ϕ is a ground assignment σ such that $T(\mathcal{F})/_{=_{AC}} \models \phi\sigma$. The set of AC-solutions of ϕ is denoted by $[\![\phi]\!]_{AC}$. ϕ is AC-*satisfiable *if it has AC-solutions.*

In the following the prefix AC will be used to indicate that we are considering the algebra $T(\mathcal{F})/_{=_{AC}}$.

DEFINITION 2.6. *Two equational formulae ϕ and ψ are* equivalent *(resp. AC-*equivalent*) if they have the same set of solutions (resp. AC-solutions). This is written*

$$\phi \sim \psi \qquad \textit{iff} \qquad [\![\phi]\!] = [\![\psi]\!] \quad (\textit{resp. } \phi \sim_{AC} \psi \quad \textit{iff} \quad [\![\phi]\!]_{AC} = [\![\psi]\!]_{AC}).$$

For example, every unification problem is equivalent and AC-equivalent either to $\top$ or to $\bot$ or to a finite disjunction of formulae of the form $\exists z_1, \ldots, z_m : x_1 = t_1 \wedge \ldots \wedge x_n = t_n$ where $x_1, \ldots, x_n$ are free variables of the unification problem and occur only once in the conjunction. This latter kind of formula can be identified with $\sigma = \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ which is called a *generator* of the set of solutions:

$$[\![\exists z_1, \ldots, z_m : x_1 = t_1 \wedge \ldots \wedge x_n = t_n]\!] = \{\sigma\theta \mid \theta \in \Sigma_g\}.$$

More generally we have the following:

DEFINITION 2.7. *A* generator *(resp. AC-*generator*) of a set of solutions (resp. AC-solutions) of an equational formula ϕ is a substitution σ such that for all $\theta \in \Sigma_g$, $\sigma\theta$ is a solution (resp. AC-solution) of ϕ. A set $\mathcal{S}$ of generators is* complete *(for ϕ) if every solution of ϕ is an instance of some generator $\sigma \in \mathcal{S}$.*

Since AC-unification (i.e. unification in $T(\mathcal{F}, \mathcal{X})/_{=_{AC}}$) is *decidable and finitary* (Stickel, 1981; Comon, 1991), any unification problem ϕ has a finite complete set of idempotent AC-generators[†]. We will denote by $CSU_{AC}(\phi)$ the complete set of idempotent AC-generators obtained by applying to the unification problem ϕ Fages's algorithm (Fages, 1987) described in the appendix.

DEFINITION 2.8. *An equational formula ϕ is E_U (*"equivalent to a unification problem"*) (resp. AC-E_U) if there is a unification problem ψ such that $\phi \sim \psi$ (resp. $\phi \sim_{AC} \psi$).*

EXAMPLE 2.1. *The following formulae are E_U and AC-E_U and will help us to explain some of the transformation rules that follow:*

- *$f(x, y) \neq g(z)$ which is equivalent to $\top$*
- *$x \neq 0$ which is equivalent to $\exists z : x = s(z)$ if $\mathcal{F} = \{0, s\}$*
- *$\forall y : x \neq y + a$ which is equivalent to $(\exists w_1, w_2, \forall y : x = w_1 + w_2 \wedge w_1 + w_2 \neq y + a) \vee x = a$ if $\mathcal{F} = \{a, +\}$, and this is equivalent to $x = a$ if $+$ is AC.*

We are interested in two semantic properties of AC-complement problems: satisfiability, and AC-equivalence to unification problems. Given a complement problem P, if there is a unification problem (a positive formula) ϕ such that $P \sim_{AC} \phi$ we say that *negation can be eliminated from P.*

[†] AC-unifiers are usually defined as solutions of unification problems in $T(\mathcal{F}, \mathcal{X})/_{=_{AC}}$. A complete set of AC-unifiers is then a complete set of AC-generators as well.

3. Satisfiability and Negation Elimination

In order to decide if a complement problem is satisfiable in $T(\mathcal{F})/_{=_{AC}}$, or, in general, if a complement problem has the AC-E_U property, first we use a set R_{EAC} of rules which transforms any complement problem into an equivalent finite set of $\forall$-*simple formulae*, and then we apply a decision test over $\forall$-simple formulae.

Although negation elimination is more general than satisfiability (in that the answer to the latter problem can be deduced from the answer to the former) we present first a decision test for satisfiability alone. This will facilitate the presentation and proofs concerning elimination of negation. By refining the satisfiability test we will obtain a test for AC-E_U.

Hereafter we assume that the complement problems we want to decide are linear. However R_{EAC} can also be applied to non-linear problems. The hypothesis of linearity is needed only for the tests.

DEFINITION 3.1. *A* $\forall$-simple formula[†] *is either* $\top$, $\bot$ *or*

$$\exists \vec{w}, \forall \vec{y} : \Big(\bigwedge_{k \in K} \Big(\bigvee_{l \in L_k} w_{kl} \neq u_{kl} \Big) \Big) \wedge x_1 = v_1 \wedge \ldots \wedge x_n = v_n$$

where

1. $x_1, \ldots, x_n$ *are the free variables (or unknowns), they occur only once in the formula,*
2. K, L_k *are finite sets of indexes,* w_{kl} *(for* $k \in K, l \in L_k$*) are existentially quantified variables in* $\vec{w}$*, and* $Var(v_1) \cup \cdots \cup Var(v_n) = \vec{w}$*,*
3. *for all* k, l*,* $Var(u_{kl}) \subseteq \vec{y}$*; and there exist* $k \in K$*,* $l \in L_k$ *such that* u_{kl} *is not ground,*
4. *each variable appears at most once in a disjunction.*

Note that K *and* L *can be empty, and in this case we have* $\exists \vec{w} : x_1 = v_1 \wedge \ldots \wedge x_n = v_n$.

We will also characterize the class of formulae that are obtained in the intermediate steps of the transformation:

DEFINITION 3.2. *An* intermediate formula *is either* $\top$, $\bot$ *or*

$$\exists \vec{w}, \forall \vec{y} : (\bigwedge_{k \in K} (\bigvee_{l \in L_k} t_{kl} \neq u_{kl})) \wedge x_1 = v_1 \wedge \ldots \wedge x_n = v_n$$

where

1. $x_1, \ldots, x_n$ *are the free variables, they occur only once in the formula,*
2. K, L_k *are finite sets of indexes; for all* $k \in K, l \in L_k$*,* $Var(t_{kl}) \cap \vec{y} = \emptyset$*; and* $Var(v_1) \cup \cdots \cup Var(v_n) = \vec{w}$*,*
3. *for all* k, l*,* $Var(u_{kl}) \subseteq \vec{y}$*,*
4. *each variable in* $\vec{y}$ *appears at most once in a disjunction.*

Also in this case K *and* L *can be empty.*

[†] In contrast with the simple formulae defined by Comon and Fernández (1992), note that $\forall$-simple formulae can still contain universally quantified variables.

The only difference between ∀-simple formulae and intermediate formulae is in the form of the disequations. In the case of ∀-simple formulae, the disequations are formed by an existential variable and a term containing only universally quantified variables, whereas in the case of an intermediate formulae, the disequations are formed by a term containing existential variables and a term containing universal variables (i.e. ∀-simple formulae are also intermediate formulae).

Note that a linear complement problem $\forall \vec{y} : t \neq t_1 \wedge \ldots \wedge t \neq t_n$ with unknowns $x_1, \ldots, x_m$ is equivalent to the intermediate formula

$$\exists \vec{w}, \forall \vec{y} : (t \neq t_1 \wedge \ldots \wedge t \neq t_n)\{x_1 \mapsto w_1, \ldots, x_m \mapsto w_m\} \wedge x_1 = w_1 \wedge \ldots \wedge x_m = w_m.$$

In the following *both kinds of formulas will be called complement problems.* It will be clear from the context (when not explicitly written) which is the representation we are considering.

3.1. TRANSFORMATION RULES

Our goal here is to show that every complement problem is equivalent in $T(\mathcal{F})/_{=_{AC}}$ to a finite disjunction of ∀-simple formulae. For this, we present a set R_{EAC} of transformation rules (see Figure 1), such that every intermediate formula (and then every complement problem) which is not a ∀-simple formula can be reduced using these rules. We must prove that each rule transforms a formula into an equivalent one and that the set of rules is terminating, i.e. all reduction sequences are finite and end in a ∀-simple formula.

Almost all the rules in R_{EAC} are conditional. They can be applied only if the condition is true. Some of the conditions are necessary for the system to be correct while others (in particular those involving the applicability of other rules) are only used in the proof of termination.

R_{EAC} contains some classical rules for eliminating universally quantified variables and reorganizing formulae (see e.g. Comon and Lescanne (1989) for more details) and some additional rules which are specific for complement problems modulo AC, such as explosion (which expresses the domain closure axiom) and decomposition rules. The empty disjunction is denoted by $\bot$ and the empty conjunction by $\top$.

There are two decomposition rules:

- D_1 is based on the fact that if $t = u$ has an AC-unifier σ which does not instantiate the variables in t then there is no solution for the complement problem $\forall \vec{y} : t \neq u$ where $Var(u) = \vec{y}$, because for all ground assignment ρ, $\sigma\rho$ unifies $t\rho$ and u).
- D_2 decomposes a disequation $t \neq u$ according to the set of AC-unifiers of $t = u$. Note that, unlike for complement problems in $T(\mathcal{F})$, in $T(\mathcal{F})/_{=_{AC}}$ new universally quantified variables may be introduced. To avoid non-termination problems caused by successive applications of explosion and decomposition rules, we add a label 0 to each existentially quantified variable in the starting problem, and we require that the existentially quantified variables introduced in the application of an explosion rule to a variable $x^{(i)}$ have a label $i+1$. Moreover, we do not allow the application of an explosion rule to a variable labeled by i if $i \geq K$ for a given K. K can be arbitrarily chosen since it is used only to ensure the termination of the process. If the starting problem is $\forall \vec{y} : t \neq t_1 \wedge \ldots \wedge t \neq t_n$ we will choose $K = \max_i\{\text{depth}(t_i)\}+1$, where $\text{depth}(t_i)$ is the depth of the tree corresponding to the term t_i. Note that D_2

Universal Quantifier Elimination (QE)

$$(QE_1) \quad \forall y : P \to P.$$

If $y \notin Var(P)$.

$$(QE_2) \quad \forall \vec{y} : P \wedge (y \neq t \vee d) \to \forall \vec{y} : P \wedge d\{y \mapsto t\}.$$

If d is a disjunction of disequations, $y \in \vec{y}$ and $y \notin Var(t)$.

Reorganization (RO)

$$(RO_1) \quad P \wedge \bot \to \bot$$
$$(RO_2) \quad P \wedge \top \to P$$
$$(RO_3) \quad P \vee \top \to \top$$
$$(RO_4) \quad P \vee \bot \to P.$$

Decomposition (D)

$$(D_1) \quad \exists \vec{w}, \forall \vec{y} : P \wedge (t \neq u \vee d) \to \exists \vec{w}, \forall \vec{y} : P \wedge d.$$

If $t, u \notin \mathcal{X}$ and there exists $\theta \in CSU_{AC}(t = u)$ such that θ does not instantiate existentially quantified variables.

$$(D_2) \quad \exists \vec{w}, \forall \vec{y} : P \wedge (t \neq u \vee d) \to \exists \vec{w}, \forall \vec{y}, \forall \vec{z} : P \wedge \bigwedge_{\theta \in CSU_{AC}(t=u \wedge \neg d)} \Big(\bigvee_{w_j \in Dom(\theta)} w_j \neq w_j\theta \Big).$$

If $t, u \notin \mathcal{X}$ and D_1 does not apply. $\vec{z}$ are the variables introduced by the $\theta \in CSU_{AC}(t = u \wedge \neg d)$.

Explosion (E)

$$(Ex_\forall) \quad \exists \vec{w}, \exists w^{(i)}, \forall \vec{y} : P \to \bigvee_{f \in \mathcal{F}} \exists \vec{w}, \exists \overrightarrow{w'^{(i+1)}}, \forall \vec{y} : P\{w^{(i)} \mapsto f(\overrightarrow{w'^{(i+1)}})\}.$$

If the formula is not $\forall$-simple, there is a disequation $w^{(i)} \neq u$ in P such that $Var(u) \cap \vec{y} \neq \emptyset$, $u \notin \mathcal{X}$ and $i < K$; no other rule can be applied and $\overrightarrow{w'^{(i+1)}}$ are fresh variables (different for each function symbol).

$$(Ex_{gr}) \quad \exists \vec{w}, \exists w^{(i)}, \forall \vec{y} : P \to \bigvee_{f \in \mathcal{F}} \exists \vec{w}, \exists \overrightarrow{w'^{(i+1)}}, \forall \vec{y} : P\{w^{(i)} \mapsto f(\overrightarrow{w'^{(i+1)}})\}.$$

If the formula is not $\forall$-simple, there is a disequation $w^{(i)} \neq u$ in P such that u is a ground term, no other rule except $Ex_\forall$ can be applied, $\overrightarrow{w'^{(i+1)}}$ are fresh variables (different for each function symbol), and if $i \geq K$ there is no disequation $w^{(i)} \neq v$ where $w^{(i)}$ is a variable, v is not a variable and $Var(v) \cap \vec{y} \neq \emptyset$.

Figure 1. Transformation rules for AC complement problems.

applies also when the set of AC-unifiers is empty, and in this case its effect is the elimination of the disjunction containing $t \neq u$.

There are two explosion rules in R_{EAC}. $Ex_\forall$ generalizes the explosion rule of Comon and Lescanne (1989). Besides, now we have to use the domain closure axiom not only for quantifier elimination but also for eliminating disequations whose one member is a ground term, as it is done in Comon and Fernández (1992). For example, assuming $\mathcal{F} = \{0, s\}$, the formula $x \neq 0$ is equivalent to $\exists w : x = s(w)$. This is expressed by the rule Ex_{gr}. Since complement problems are represented by formulas where the free variables appear only in equations, we do not need to consider explosions on free variables.

We will prove in the following section that R_{EAC} is terminating and transforms any complement problem into an equivalent disjunction of $\forall$-simple formulae.

3.2. correctness, termination and irreducible forms of R_{EAC}

Proposition 3.1. (Correctness) *Let ϕ be an intermediate formula. If $\phi \rightarrow^*_{R_{EAC}} \psi$ then $[\![\phi]\!]_{AC} = [\![\psi]\!]_{AC}$, i.e. R_{EAC} preserves solutions.*

Proof. In order to prove that R_{EAC} preserves solutions (hence satisfiability) we only have to show that each rule preserves solutions. This is obvious for reorganization and universal quantifier elimination and also for the decomposition rule D_1. For D_2 the result is a consequence of the logical identity

$$\exists \vec{w}, \forall \vec{y} : P \wedge \bigwedge_{\theta \in CSU_{AC}(t=u \wedge \neg d)} \forall \vec{z} : \Big(\bigvee_{w_j \in Dom(\theta)} w_j \neq w_j\theta \Big)$$
$$\equiv \exists \vec{w}, \forall \vec{y}, \forall \vec{z} : P \wedge \bigwedge_{\theta \in CSU_{AC}(t=u \wedge \neg d)} \Big(\bigvee_{w_j \in Dom(\theta)} w_j \neq w_j\theta \Big)$$

where $\vec{z} \notin Var(P)$. For the explosion rules, it is a consequence of the domain closure axiom. □

As a first step towards a proof of termination and a characterization of irreducible forms of R_{EAC}, let us show some properties of the formulae obtained along the transformation process. Property 3.1 below stresses the fact that the set of disjunctions of intermediate formulae is closed by R_{EAC}, that is, each rule in R_{EAC} transforms a disjunction of intermediate formulae into another disjunction of intermediate formulae. Property 3.2 gives some characteristics of the formulae obtained after applications of **RO**, **QE** and **D** rules. In order to prove these properties we will use the following lemma:

Lemma 3.1. *Let $P_0 \equiv s_1 = t_1 \wedge \ldots \wedge s_n = t_n$ where $t_1, \ldots, t_n$ are linear terms, $Var(s_i) \cap Var(t_1, \ldots, t_n) = \emptyset$ for all i, and $Var(t_i) \cap Var(t_j) = \emptyset$ for all $i \not\equiv j$. Then, there exists a $CSU_{AC}(P_0)$, $\mathcal{S}$, such that for all $\sigma \in \mathcal{S}$:*

1. *for all $x \in Var(s_1, \ldots, s_n)$, if $x\sigma \neq x$ then $Var(x\sigma) \cap Var(s_1, \ldots, s_n) = \emptyset$*
2. *$Im(\sigma|_{Var(s_1,\ldots,s_n)})$ is linear.*

Proof. We use Stickel's AC-unification algorithm as it is presented by Fages (1987), where it is proved to be correct, complete and terminating. A description of the algorithm

is given in Appendix A. We will prove that by using this algorithm we obtain a set $\mathcal{S}$ of AC-unifiers satisfying the required conditions (the proof cannot be easily adapted to other presentations of the algorithm, but to show that the lemma holds it is enough to consider this particular algorithm).

Let $\mathcal{S}$ be $\texttt{unicompound}([s_1, \ldots, s_n], [t_1, \ldots, t_n], \epsilon, Var(P_0))$ where `unicompound` is defined in Fages's algorithm and ϵ is the identity substitution.

Since this algorithm terminates, we can proceed by induction. The base case is trivial. We consider the cases where recursive calls are made:

There are two recursive calls in $\texttt{unicompound}([s_1, \ldots, s_n], [t_1, \ldots, t_n], \epsilon, Var(P_0))$ (marked by (*) and (**) in the appendix). We must show that in both cases the hypothesis of the lemma holds, and then by induction we get the thesis. For this, we will prove that `uniAC` is only applied to terms satisfying the hypothesis and that in this case the recursive calls of `unicompound` satisfy the hypothesis.

Assume `uniAC` is applied to terms s, t satisfying the hypothesis. It is obvious that in (*) the hypothesis of the lemma hold. For (**) this is a consequence of the linearity of t: when t is linear the solutions over N of the Diophantine equation corresponding to $s = t$ are linear in $Var(s)$, then all the pairs (A, B) in the set returned by `dio` satisfy the hypothesis. Hence, by induction, the recursive calls of `unicompound` return a set of AC-unifiers satisfying the required properties. Moreover, if s_i, t_i satisfy the hypothesis, then $s_i\sigma_i$ and $t_i\sigma_i$ (which is the same as t_i because $Var(t_i) \cap Var(t_j) = \emptyset$ for all $i \not\equiv j$) satisfy the hypothesis too. Hence `uniAC` is always applied to terms satisfying the hypothesis. □

PROPERTY 3.1. *If ϕ is a complement problem and $\phi \rightarrow^*_{R_{EAC}} \psi$ then ψ is a finite disjunction of intermediate formulae, and the free variables of ψ are included in those of ϕ.*

PROOF. Since $\mathcal{F}$ is finite and AC-unification is finitary, ψ is finite. We proceed by induction on the length of the derivation $\phi \rightarrow^*_{R_{EAC}} \psi$. It is sufficient to prove that the property is kept by any transformation rule (because if $\phi \equiv \psi$ the property holds trivially). This is straightforward for the rules that remove a part of the formula, such as D_1, QE_1 and reorganization rules. QE_2 preserves the form because it eliminates a disequation from a disjunction and applies a substitution whose domain is a variable not appearing in the disjunction. Since intermediate formulae satisfy the hypothesis of Lemma 3.1 it is easy to see that D_2 preserves the form of the formula too. The same for the explosion rules, which add a disjunction of problems of the same shape where all new variables are existentially quantified. □

PROPERTY 3.2. *Let ϕ be a complement problem and*

$$\phi \rightarrow_{R_{EAC}} \phi_1 \rightarrow_{R_{EAC}} \cdots \rightarrow_{R_{EAC}} \phi_n \rightarrow_{R_{EAC}} \cdots$$

be a finite or infinite sequence of transformations using R_{EAC}. If ϕ_i is irreducible by the rules in **QE**, **D** *and* **RO** *then all the disequations in ϕ_i have the form $w \neq u$ where*

1. *w is an existentially quantified variable,*
2. *u does not contain either free or existentially quantified variables,*
3. *if $w \neq u$ appears in a disjunction d then*
   (a) *each variable of u appears only once in d,*
   (b) *w appears only once in d.*

PROOF. If ϕ_i is irreducible by the rules in **QE**, **D** and **RO** then in all the disequations one of the members is an existentially quantified variable, because otherwise either decomposition or universal quantifier elimination rules would apply. This proves Part 1.

Parts 2 and 3.a are direct consequences of Property 3.1 and the definition of intermediate formulae, while Part 3.b is a consequence of the fact that disjunctions of disequations are created only by decomposition rules, and are associated with AC-unifiers (which are idempotent substitutions). □

PROPOSITION 3.2. (TERMINATION) *Let ϕ be a complement problem. There is no infinite sequence*

$$\phi \to_{R_{EAC}} \phi_1 \to_{R_{EAC}} \cdots \to_{R_{EAC}} \phi_n \to_{R_{EAC}} \cdots.$$

PROOF. First we are going to prove that $R_{EAC} - \{Ex_\forall\}$ terminates. For this, we give an interpretation $\Im$ of the formulae in a well-founded ordered set, such that for each application of a rule the interpretation is strictly decreasing.

Let $\Im(\phi)$ be the 4-tuple $\langle \Im_1(\phi), \Im_2(\phi), \Im_3(\phi), \Im_4(\phi)\rangle$ where $\Im_1(\phi)$ is the number of disequations $t \neq u$ in ϕ such that $u, t \notin \mathcal{X}$, $\Im_2(\phi)$ is the pair $\langle \Im_{21}(\phi), \Im_{22}(\phi)\rangle$ where $\Im_{21}(\phi)$ is the multiset of numbers $K - \text{label}(w_i)$ for each disequation $w_i \neq u_i$ in ϕ such that w_i is an existentially quantified variable and u_i is not ground, and $\Im_{22}(\phi)$ is the multiset of depth(u_i) for each disequation $w_i \neq u_i$ in ϕ such that w_i is existentially quantified and u_i is ground, $\Im_3(\phi)$ is the number of disequations in ϕ and $\Im_4(\phi)$ is the size of ϕ (number of symbols appearing in ϕ). The lexicographic extension of the ordering $<$ on natural numbers is well-founded on this set of 4-tuples. Moreover, this interpretation is strictly decreasing for the rules in $R_{AC} - \{Ex_\forall\}$.

- Rules D_1 and D_2 decrease the first component of the 4-tuple.
- QE_2 does not change the first and second components (this is because by Property 3.1, $\phi, \phi_1, \ldots, \phi_n$ are disjunctions of intermediate formulae, therefore each universally quantified variable appears at most once in a disjunction, which means that QE_2 simply eliminates one disequation), and it decreases the third one.
- QE_1 and Reorganization rules do not increase the first, second and third components but decrease the fourth one.
- Since an application of rule Ex_{gr} is always followed by a finite number of applications of **D** and **QE** rules: $\phi_i \to_{Ex_{gr}} \phi_{i+1} \to^+_{\mathbf{D},\mathbf{QE}} \phi_j$, we do not consider the intermediate steps between ϕ_{i+1} and ϕ_j in the sequence of transformations. Then, an application of $Ex_{gr} \circ \{\mathbf{D}, \mathbf{QE}\}^+$ does not change the first component. Let us prove that it decreases the second one.
  If after replacing w by $f(\vec{w'})$ and decomposing, a new non-ground u_i appears it is because we had $w \neq t$ where t is non-ground. But since by Property 3.2 neither free nor existentially quantified variable is contained in t, the new disequations must have the form $w'_i \neq u_i$ where w'_i is existentially quantified and its label is greater than the one of w. Then $\Im_{21}(\phi)$ decreases.
  Otherwise, $\Im_{21}(\phi)$ does not change but $\Im_{22}(\phi)$ decreases, because if we replace w by $f(\vec{w'})$ in a disequation $w \neq u_i$ and the root of u_i is not f then the terms become not AC-unifiable and the disequation disappears, and if the root is f then the disequation is replaced by a conjunction of disjunctions where the right member is less deep (because they come from the AC-unifiers of $f(\vec{w'})$ and the ground term u_i).

	$\Im_1$	$\langle \Im_{21}, \Im_{22} \rangle$	$\Im_3$	$\Im_4$
D	$<$			
QE_1	$=$	$=$	$=$	$<$
QE_2	$=$	$=$	$<$	
RO	$\leq$	$\leq$	$\leq$	$<$
$Ex_{gr} \circ \{\mathbf{D}, \mathbf{QE}\}^+$	$=$	$<$		

Figure 2. Summary of the variations of $\Im$ w.r.t. the rules of R_{EAC}.

A summary of the variations of $\Im$ is given in Figure 2.

Hence, $R_{AC} - \{Ex_\forall\}$ is terminating.

Now, we will prove that in every finite or infinite sequence of transformations of problems

$$\phi \rightarrow_{R_{EAC}} \phi_1 \rightarrow_{R_{EAC}} \cdots \rightarrow_{R_{EAC}} \phi_n \rightarrow_{R_{EAC}} \cdots$$

the rule $Ex_\forall$ is applied only a finite number of times. We consider the interpretation $\Im'$ such that $\Im'(\phi)$ is the multiset of $K - \text{label}(w)$ for each occurrence of an existentially quantified variable w in ϕ such that $\text{label}(w) \leq K$. It is easy to see that $Ex_\forall$ strictly decreases $\Im'$. Since the other rules do not increase it (the only difficult case is rule D_2, but by Part 2 of Property 3.2 we know that D_2 cannot introduce occurrences of existentially quantified variables having a smaller label), $Ex_\forall$ applies only a finite number of times.

This fact and the termination of $R_{AC} - \{Ex_\forall\}$ prove that R_{EAC} terminates[†]. □

It remains to prove that the irreducible formulae of R_{EAC} are finite disjunctions of $\forall$-simple formulae.

Proposition 3.3. (Irreducible formulae) *Let ϕ be a linear complement problem. Then any irreducible form of ϕ is a finite disjunction of $\forall$-simple formulae.*

Proof. It is a consequence of Properties 3.1 and 3.2:

By Property 3.1, the irreducible forms of ϕ are either $\top$, $\bot$ or else have the form $\bigvee_{j \in J} \phi_j$ where ϕ_j is $\exists \vec{w_j} \forall \vec{y_j} : \bigwedge_{k \in K_j} (\bigvee_{l \in L_{kj}} t_{kjl} \neq u_{kjl}) \wedge x_1 = v_{j1} \wedge \ldots \wedge x_n = v_{jn}$ (the free variables are included in those of ϕ).

Moreover, in all the disequations one member (e.g. t_{kjl}) is an existentially quantified variable (otherwise the formula would be reducible) and by Property 3.2 the other member does not contain existentially quantified variables nor free variables, and each variable occurs only once in a disjunction. Besides, there is at least one non-ground u_{kjl}, otherwise Ex_{gr} is applicable. □

Now, we can state the main result of this section:

Theorem 3.1. *R_{EAC} transforms any linear complement problem into an equivalent finite disjunction of $\forall$-simple formulae.*

Proof. Consequence of Propositions 3.1, 3.2 and 3.3. □

[†] Although the hypothesis of linearity is used in this proof, we conjecture that R_{EAC} can be shown terminating also for non-linear complement problems.

3.3. SOLVING ∀-SIMPLE FORMULAE

By Theorem 3.1, satisfiability of linear AC complement problems reduces to satisfiability of ∀-simple formulae. The decision test we are going to present is inspired by Kounalis *et al.* (1991). We will prove that in order to decide satisfiability of a ∀-simple formula it is sufficient to consider a finite set of substitutions instead of the set of all ground assignments.

Terms involving AC function symbols will be represented in *flattened form* (Jouannaud and Kounalis, 1989): $\mathrm{flat}(t)$, for a term t, is the normal form of (the tree) t for the convergent tree rewriting system that contains the rules:

$$f(x_1, \ldots, x_{p-1}, f(y_1, \ldots, y_q), x_{p+1}, \ldots, x_n) \to f(x_1, \ldots, x_{p-1}, y_1, \ldots, y_q, x_{p+1}, \ldots, x_n)$$

for any integers p, q and n such that $1 \leq p \leq n$, and for any AC operator f.

In other words, AC-symbols are treated as varyadic symbols in flattened forms. For example, if $\mathcal{F} = \{0, succ, +\}$ and $+$ is AC, $\mathrm{flat}(+(0, +(x, +(y, x)))) = +(0, x, y, x)$.

The notion of flattening extends to substitutions and ground assignments in the natural way: $\mathrm{flat}(\sigma)$ assigns $\mathrm{flat}(x\sigma)$ to each variable x.

In the following we will use two well-known properties of flattened terms [see Jouannaud and Kounalis (1989)]:

PROPERTY 3.3. *Let t, t' be terms and σ a substitution.*

1. $t =_{AC} t'$ *if and only if* $\mathrm{flat}(t) =_P \mathrm{flat}(t')$
2. $t =_{AC} t'\sigma$ *if and only if* $\mathrm{flat}(t) =_P \mathrm{flat}(\mathrm{flat}(t')\,\mathrm{flat}(\sigma))$

where $=_P$ is the permutative congruence on subterms of AC symbols.

DEFINITION 3.3. *Let ϕ be a (non-trivial) ∀-simple formula*

$$\phi \equiv \exists \vec{w}, \forall \vec{y} : \bigwedge_{k \in K} \Big(\bigvee_{l \in L_k} w_{kl} \neq u_{kl} \Big) \wedge x_1 = v_1 \wedge \ldots \wedge x_n = v_n.$$

The formula

$$\psi \equiv \forall \vec{y} : \bigwedge_{k \in K} \Big(\bigvee_{l \in L_k} w_{kl} \neq u_{kl} \Big)$$

will be called the kernel *of ϕ. Note that the existentially quantified variables of ϕ become the unknowns of ψ.*

We will restrict our attention to kernels of ∀-simple formulae, since a ∀-simple formula is satisfiable if and only if its kernel is.

Now we are going to define the sets of substitutions that will be used in the satisfiability test.

DEFINITION 3.4. *Let $\psi \equiv \forall \vec{y} : \bigwedge_{k \in K} (\bigvee_{l \in L_k} w_{kl} \neq u_{kl})$. Let $d = \max_{k,l}\{\mathrm{depth}(\mathrm{flat}(u_{kl}))\} + 1$, and let κ be the maximal number of arguments of an AC symbol in $\mathrm{flat}(u_{kl})$ for u_{kl}*

in ψ. We define the finite sets $B(\psi)$, $A(\psi)$, and $S_{\vec{w},A(\psi)}$ as follows:

$$B(\psi) = \{\mathrm{flat}(r) \mid r \in T(\mathcal{F},\mathcal{X}), r \textit{ is linear}, \mathrm{depth}(\mathrm{flat}(r)) \leq d, \textit{AC-symbols have at most } \kappa+1 \textit{ arguments, and variables can occur only at depth } d\}.$$

where the variables occurring in $B(\psi)$ are fresh (i.e. the terms in $B(\psi)$ do not contain variables in $\vec{w},\vec{y}$).

$A(\psi)$ is obtained from $B(\psi)$ by taking out the terms that are AC-instances of other terms in the set.

$$S_{\vec{w},A(\psi)} = \{\sigma \in \Sigma \mid Dom(\sigma) = \vec{w}, Im(\sigma) \subseteq A(\psi) \textit{ and } Im(\sigma) \textit{ is linear}\}$$

$S_{\vec{w},A(\psi)}$ is the test-set *for ψ.*

EXAMPLE 3.1. *Let $\mathcal{F} = \{a, f, +\}$ where $+$ is AC, f is a unary symbol and a is a constant, and let $\psi \equiv \exists w, \forall y : w \neq y + a + a$. Here $\kappa + 1 = 4$ and $d = 2$.*

$$\begin{aligned} A(\psi) = \{\ & a, \\ & f(a), f(f(z)), \\ & a+a, a+a+a, a+a+a+a, \\ & f(z)+a, f(z)+a+a, f(z)+a+a+a, \\ & f(z_1)+f(z_2), f(z_1)+f(z_2)+a, f(z_1)+f(z_2)+a+a, \\ & f(z_1)+f(z_2)+f(z_3), f(z_1)+f(z_2)+f(z_3)+a, \\ & f(z_1)+f(z_2)+f(z_3)+f(z_4) \\ \}\ . & \end{aligned}$$

Note that in $A(\psi)$ no term is an AC-instance of other term.

$$\begin{aligned} S_{w,A(\psi)} = \{\ & \sigma_1 = \{w \mapsto a\} \\ & \sigma_2 = \{w \mapsto f(a)\} \\ & \sigma_3 = \{w \mapsto f(f(z))\} \\ & \sigma_4 = \{w \mapsto a+a\} \\ & \sigma_5 = \{w \mapsto a+a+a\} \\ & \sigma_6 = \{w \mapsto a+a+a+a\} \\ & \sigma_7 = \{w \mapsto f(z)+a\} \\ & \textit{etc.} \\ \}\ . & \end{aligned}$$

Lemma 3.2 below shows that for deciding satisfiability of ψ it is sufficient to consider only the substitutions in the test set $S_{\vec{w},A(\psi)}$.

LEMMA 3.2. *Let ϕ and ψ be formulas as above. $T(\mathcal{F})/_{=_{AC}} \models \exists \vec{y} : \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl} = u_{kl})$ if and only if for all $\lambda \in S_{\vec{w},A(\psi)}$, $T(\mathcal{F})/_{=_{AC}} \models \exists \vec{y} : \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\lambda = u_{kl})$.*

PROOF. The "only if" part is trivial. Let us prove the "if" part. For this, we must show that for any ground assignment μ, $T(\mathcal{F})/_{=_{AC}} \models \exists \vec{y} : \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\mu = u_{kl})$.

Let $w_{kl}\mu = v_{kl}$, for $k \in K, l \in L_k$.

- If $\max_{kl}\{\text{depth}(\text{flat}(v_{kl}))\} \leq d$ and for any occurrence of an AC symbol in $Im(\text{flat}(\mu|_{\vec{w}}))$ the number of arguments is less than or equal to $\kappa{+}1$ then $\text{flat}(\mu|_{\vec{w}})$ is an instance of some $\lambda \in S_{\vec{w},A(\psi)}$. Then $T(\mathcal{F})/_{=_{AC}} \models \exists \vec{y} : \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\mu = u_{kl})$ by hypothesis.
- Otherwise, we define the nth projection of the flattened term t [written $\text{proj}_n(t)$] as the term obtained from t by erasing all $(n+i)$th arguments of AC symbols ($i > 0$). For example, $\text{proj}_2(+(0, x, y, x)) = +(0, x)$. The definition extends to substitutions as usual: for any substitution σ, $\text{proj}_n(\sigma) = \{x \mapsto \text{proj}_n(x\sigma) \mid x \in Dom(\sigma)\}$.
  If there is an AC symbol occurring in $Im(\mu|_{\vec{w}})$ with more than $\kappa + 1$ arguments, there exists $\lambda \in S_{\vec{w},A(\psi)}$ such that $\lambda \preceq \mu' \equiv \text{proj}_{\kappa+1}(\text{flat}(\mu|_{\vec{w}}))$ and by hypothesis $T(\mathcal{F})/_{=_{AC}} \models \exists \vec{y} : \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\lambda = u_{kl})$. Then, for all ground assignment ρ there exists a ground assignment σ such that $T(\mathcal{F})/_{=_{AC}} \models \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\lambda\rho = u_{kl}\sigma)$. In particular there exists a ground assignment σ such that $T(\mathcal{F})/_{=_{AC}} \models \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\mu' = u_{kl}\sigma)$ since μ' is a ground instance of λ.
  Then, $\bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\mu' =_{AC} u_{kl}\sigma)$, since the terms are ground.
  Then, for some $j \in K$, $\bigwedge_{l\in L_j} w_{jl}\mu' =_{AC} u_{jl}\sigma$.
  Then, by Property 3.3, $\bigwedge_{l\in L_j} \text{flat}(w_{jl}\mu') =_P \text{flat}(\text{flat}(u_{jl})\,\text{flat}(\sigma))$.
  Since in $\text{flat}(u_{jl})$ the maximal number of arguments of an AC symbol is κ while in $w_{jl}\mu'$ it is $\kappa + 1$, $\text{flat}(\sigma)$ must introduce the remaining arguments. But since u_{jl} is linear and each variable appears only once in the conjunction, we can modify σ in order to add the arguments that remained in μ. Let σ' be the substitution obtained from σ by adding to the AC-symbols with $\kappa{+}1$ arguments those arguments that were left out to construct $\mu' = \text{proj}_{\kappa+1}(\text{flat}(\mu|_{\vec{w}}))$. Then, $\bigwedge_{l\in L_j} \text{flat}(w_{jl}\mu) =_P \text{flat}(\text{flat}(u_{jl})\,\text{flat}(\sigma'))$[†].
  Then $\bigwedge_{l\in L_j} w_{jl}\mu =_{AC} u_{jl}\sigma'$.
  Then $\bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\mu =_{AC} u_{kl}\sigma')$.
  Then $T(\mathcal{F})/_{=_{AC}} \models \exists \vec{y} : \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\mu = u_{kl})$.

□

As a consequence, the negation of the condition in Lemma 3.2, that is, $\exists \lambda \in S_{\vec{w},A(\psi)}$ such that $T(\mathcal{F})/_{=_{AC}} \not\models \exists \vec{y} : \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\lambda = u_{kl})$, is a necessary and sufficient condition for the satisfiability of ϕ:

THEOREM 3.2. *Let ϕ and ψ be formulas as above. ϕ has solutions in $T(\mathcal{F})/_{=_{AC}}$ iff $\exists \lambda \in S_{\vec{w},A(\psi)}$ such that $T(\mathcal{F})/_{=_{AC}} \not\models \exists \vec{y} : \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\lambda = u_{kl})$.*

PROOF.

$$\phi \text{ is satisfiable in } T(\mathcal{F})/_{=_{AC}}$$

iff

$$\psi \text{ is satisfiable in } T(\mathcal{F})/_{=_{AC}}$$

iff

$$\neg\Big(T(\mathcal{F})/_{=_{AC}} \models \exists \vec{y} : \bigvee_{k\in K}\Big(\bigwedge_{l\in L_k} w_{kl} = u_{kl}\Big)\Big)$$

[†] Note that linearity is fundamental in this proof. Without linearity we have only a sufficient (but not necessary) condition.

iff (Lemma 3.2)

$$\neg\Big(\textit{for all } \lambda \in S_{\vec{w},A(\psi)}, T(\mathcal{F})/_{=AC} \models \exists \vec{y} : \bigvee_{k\in K}\Big(\bigwedge_{l\in L_k} w_{kl}\lambda = u_{kl}\Big)\Big)$$

iff

$$\text{there exists } \lambda \in S_{\vec{w},A(\psi)} \text{ such that } \neg\Big(T\Big(\mathcal{F}\Big)/_{=AC} \models \exists \vec{y} : \bigvee_{k\in K}\big(\bigwedge_{l\in L_k} w_{kl}\lambda = u_{kl}\Big)\Big)$$

iff

$$\text{there exists } \lambda \in S_{\vec{w},A(\psi)} \text{ such that } T(\mathcal{F})/_{=AC} \not\models \exists \vec{y} : \bigvee_{k\in K}\Big(\bigwedge_{l\in L_k} w_{kl}\lambda = u_{kl}\Big).$$

□

Theorem 3.2 suggests an algorithm for solving ∀-simple formulae: it is sufficient to test for each $\lambda \in S_{\vec{w},A(\psi)}$ whether $T(\mathcal{F})/_{=AC} \models \exists \vec{y} : \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\lambda = u_{kl})$ or, equivalently (since all the variables in $w_{kl}\lambda$ are deeper than those in u_{kl} and the terms u_{kl} are linear), whether there is σ such that $\bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\lambda =_{AC} u_{kl}\sigma)$. A substitution σ satisfying this condition is usually called an AC-matcher. So, Theorem 3.2 can be reformulated as follows:

ϕ is satisfiable if and only if there exists $\lambda \in S_{\vec{w},A(\psi)}$ such that the AC-matching problem $\bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\lambda =_{AC} u_{kl})$ has no solution.

AC-matching is a well-known decidable problem (Hullot, 1979), so this gives a decision test for satisfiability of ∀-simple formulae.

In fact, this decision test can be applied to the irreducible forms of $R_{AC} = R_{EAC} - \{Ex_{gr}\}$, since Ex_{gr} is only needed to eliminate negation. R_{AC} is correct and terminating. Its irreducible forms differ from those of R_{EAC} in that now all the terms u_{kl} can be ground, i.e. condition 3 in the definition of ∀-simple formulae is no longer required. Now, if for all k, l, u_{kl} is ground, ϕ has solutions, since for all terms u, t such that $\neg(u =_{AC} t)$ there is a solution of $u \neq t$ in $T(\mathcal{F})/_{=AC}$. Otherwise we apply the test.

This decision test can be written as a set $\mathbf{T}$ of transformation rules to be added to R_{AC}, see Figure 3.

A careful analysis of the algorithm issued from $R_{AC} \cup \mathbf{T}$ shows that the complexity of the transformation process using R_{AC} is given by the complexity of AC-unification (which is known to be doubly exponential in general, but it is simply exponential for linear unification problems), and the complexity of $\mathbf{T}$ is $O(|\mathcal{F}|^{a^d})$, where $a = \max\{\max_{f\in\mathcal{F}} \{\text{arity}(f)\}, \kappa\}$. It is then clear that only by decreasing the values of d and κ one can improve significantly the efficiency of the algorithm.

When the input is a linear AC-complement problem $\forall \vec{y} : t \neq t_1 \wedge \ldots \wedge t \neq t_n$ where t is also a linear term (this is the class of problems we have to solve to eliminate the ambiguity of functional programs), the AC-unification problems that have to be solved during the transformation process are always linear, and this ensures that the values of κ and d do not increase during the transformation process. In this case the bounds d and κ we use for constructing the test set are smaller than those used by Kounalis *et al.* (1991), since we do not consider the terms in the original problem but the smaller terms in the formulae obtained after application of R_{AC}. However, if t is not linear, a non-linear AC-unification problem can appear during the transformation, and AC-unification can increase the values of κ and d, in which case it is better to stop the transformation process

Test (T)

$$(T_0)\quad P \equiv \exists \vec{w} : \bigwedge_{k\in K}\Big(\bigvee_{l\in L_k} w_{kl} \neq u_{kl}\Big) \wedge x_1 = v_1 \wedge \ldots \wedge x_n = v_n \rightarrow \top.$$

If P is irreducible by R_{AC} and all u_{kl} are ground terms.

$$(T_1)\quad P \equiv \exists \vec{w}, \forall \vec{y} : \bigwedge_{k\in K}\Big(\bigvee_{l\in L_k} w_{kl} \neq u_{kl}\Big) \wedge x_1 = v_1 \wedge \ldots \wedge x_n = v_n \rightarrow \bot\ .$$

If P is irreducible by R_{AC}, there exists k, l such that u_{kl} is not ground, and for all $\lambda \in S_{\vec{w},A(\psi)}$, $T(\mathcal{F})/_{=_{AC}} \models \exists \vec{y} : \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\lambda = u_{kl})$.

$$(T_2)\quad P \equiv \exists \vec{w}, \forall \vec{y} : \bigwedge_{k\in K}\Big(\bigvee_{l\in L_k} w_{kl} \neq u_{kl}\Big) \wedge x_1 = v_1 \wedge \ldots \wedge x_n = v_n \rightarrow \top.$$

If P is irreducible by R_{AC}, there exists k, l such that u_{kl} is not ground, and there exists $\lambda \in S_{\vec{w},A(\psi)}$ such that $T(\mathcal{F})/_{=_{AC}} \not\models \exists \vec{y} : \bigvee_{k\in K}(\bigwedge_{l\in L_k} w_{kl}\lambda = u_{kl})$.

Figure 3. Decision test for satisfiability of ∀-simple formulae.

and apply the test directly, as it is done by Kounalis *et al.* (1991). The complexities of both algorithms are equal in the worst case, since using our algorithm the (smaller) test has to be applied to each ∀-simple formula in the disjunction obtained after the transformation process, which is equivalent to applying the bigger test to the original problem.

3.4. NEGATION ELIMINATION IN ∀-SIMPLE FORMULAE

Negation elimination in *AC* complement problems reduces to the problem of negation elimination in finite disjunctions of ∀-simple formulae (by Theorem 3.1). Moreover, as a consequence of the following lemma, each disjunct can be treated separately.

LEMMA 3.3. *Let P be a linear AC-complement problem and $\bigvee_{j\in J} P_j$ be an irreducible form by R_{EAC} of P. P is AC-E_U if and only if every P_j is so.*

PROOF. The "if" part is trivial. Let us prove the "only if": Correctness of R_{EAC} implies $[\![P]\!]_{AC} = [\![\bigvee_{j\in J} P_j]\!]_{AC}$, which implies that P is AC-E_U if and only if $\bigvee_{j\in J} P_j$ is. We have to show that if P is AC-E_U then each P_j is. Since the only rules that introduce disjunctions are the explosions, we consider only this case. If $P \rightarrow_{\mathbf{E}} \bigvee_{f\in\mathcal{F}} P_f$ then each P_f is obtained from P by instantiating a variable, in a way that is essentially equivalent to adding an equation. This means that if P is AC-E_U then each P_f is. The thesis follows by a simple induction on the number of applications of explosion rules. □

Now we are faced with the problem of negation elimination in *one* ∀-simple formula. The following is a very important property:

LEMMA 3.4. *Let ϕ be a $\forall$-simple formula*

$$\phi \equiv \exists \vec{w}, \forall \vec{y} : \bigwedge_{k \in K} \Big(\bigvee_{l \in L_k} w_{kl} \neq u_{kl} \Big) \wedge x_1 = v_1 \wedge \ldots \wedge x_n = v_n$$

and ψ its kernel

$$\psi \equiv \forall \vec{y} : \bigwedge_{k \in K} \Big(\bigvee_{l \in L_k} w_{kl} \neq u_{kl} \Big).$$

ϕ is AC-E_U if and only if ψ is.

PROOF. The "if" part is trivial. Let us prove the "only if". Assume ϕ is AC-E_U, then there exists a finite complete set of AC-generators for ϕ:

$$G = \{\sigma'_1, \ldots, \sigma'_m\}$$

where for $1 \leq j \leq m$, $Dom(\sigma'_j) = \{x_1, \ldots, x_n\}$.

Let $\{\theta_{i1}, \ldots, \theta_{im_i}\}$ be a complete set of AC-unifiers of $x_1\sigma'_i = v_1 \wedge \ldots \wedge x_n\sigma'_i = v_n$, $1 \leq i \leq m$ (recall that $Var(v_1) \cup \cdots \cup Var(v_n) = \vec{w}$) and let

$$\Theta_i = \{\theta_{i1}|_{\vec{w}}, \ldots, \theta_{im_i}|_{\vec{w}}\}.$$

We will show that $\Theta = \bigcup_{i=1,\ldots,m} \Theta_i = \{\sigma_1, \ldots, \sigma_q\}$ is a complete set of AC-generators for ψ, that is, we will prove that a ground assignment σ is a solution of ψ if and only if there exists $\sigma_h \in \Theta$ such that σ is an AC-instance of σ_h.

Let us prove first the "only if". Assume σ is a solution of ψ. Let τ be a ground assignment such that $x_j\tau = v_j\sigma$ for $1 \leq j \leq n$. As τ is a solution of ϕ, then $\tau =_{AC} \sigma'_h\mu$ for some $\sigma'_h \in G$ and ground assignment μ. Besides, since $x_j\tau = v_j\sigma$ $(1 \leq j \leq n)$, $x_j\sigma'_h\mu =_{AC} v_j\sigma$. Now, let ρ be the ground assignment such that $\rho|_{Var(Im(\sigma'_h))} = \mu|_{Var(Im(\sigma'_h))}$. Then $x_j\sigma'_h\rho =_{AC} v_j\rho$. Note that ρ is well-defined since $Var(v_j) \subseteq \vec{w}$ and $Var(Im(\sigma'_h)) \cap \vec{w} = \emptyset$. Then ρ is an AC-instance of $\sigma_h \in \Theta$. Then σ is an AC-instance of σ_h since $\rho|_{\vec{w}} = \sigma|_{\vec{w}}$.

Let us prove now the "if". Assume the ground assignment σ is an AC-instance of $\sigma_h \in \Theta$ (recall that $Dom(\sigma_h) = \vec{w}$). We have to prove that σ is a solution of ψ. This is a consequence of the following property of R_{EAC}-derivations:

Let P be a linear complement problem such that

$$P \equiv \bigvee_{j \in J_0} \phi_{0j} \rightarrow_{R_{EAC}} \bigvee_{j \in J_1} \phi_{1j} \rightarrow_{R_{EAC}} \cdots \rightarrow_{R_{EAC}} \bigvee_{j \in J_p} \phi_{pj}$$

where for $1 \leq i \leq p$ and $j \in J_i$, $\phi_{ij} \equiv \exists \vec{w} : \psi_{ij} \wedge x_1 = v_{ij1} \wedge \ldots \wedge x_n = v_{ijn}$. Let $\{\sigma'_1, \ldots, \sigma'_m\}$ be a complete set of AC-generators of ϕ_{ij} and $\Theta = \{\sigma_1, \ldots, \sigma_q\} = \bigcup_{k=1,\ldots,m} \Theta_k$ where Θ_k is a complete set of AC-unifiers of $x_1\sigma'_k = v_{ij1} \wedge \ldots \wedge x_n\sigma'_k = v_{ijn}$. If a ground assignment σ is an AC-instance of $\sigma_h \in \Theta$ then σ is an AC-solution of ψ_{ij}.

We prove this property by induction on the length of the derivation.

[Base Case ($p = 0$):] It is trivial since $P \equiv \exists \vec{w} : t \neq t_1 \wedge \ldots \wedge t \neq t_{n'} \wedge x_1 = w_1 \wedge \ldots \wedge x_n = w_n$.

[Induction hypothesis:] Let $\phi_{pj} \equiv \exists \vec{w} : \psi_{pj} \wedge x_1 = v_{pj1} \wedge \ldots \wedge x_n = v_{pjn}$ and let σ be an AC-instance of σ_h. Then σ is a solution of ψ_{pj}.

We have to prove the property for each $\phi_{p+1,j}$ such that $\bigvee_{j \in J_p} \phi_{pj} \rightarrow_{R_{EAC}} \bigvee_{j \in J_{p+1}} \phi_{p+1,j}$. Assume R_{EAC} was applied to ϕ_{pk}, that is, $\phi_{pk} \rightarrow_{R_{EAC}} \bigvee_i \phi_{p+1,k_i}$. Then $\bigvee_{j \in J_{p+1}} \phi_{p+1,j} \equiv \phi_{p1} \vee \ldots \vee \phi_{p,k-1} \vee \phi_{p+1,k_1} \vee \ldots \vee \phi_{p+1,k_q} \vee \phi_{p,k+1} \vee \ldots \vee \phi_{pj_p}$. The thesis follows trivially

from the induction hypothesis for $\phi_{p1}, \ldots, \phi_{p,k-1}, \phi_{p,k+1}, \ldots, \phi_{pj_p}$. It follows also trivially for $\bigvee_i \phi_{p+1,k_i}$ if the rule applied in the last step is in **QE**, **RO** or **D**. If it is an explosion step, then $\phi_{p+1,k_i} \equiv \exists \vec{w'} : \phi_{pk}\{w \mapsto f(\vec{w'})\}$. If σ is an AC-instance of σ_h for σ_h in the set Θ of some ϕ_{p+1,k_i} then there exists σ' generator of solutions of ϕ_{p+1,k_i} and there exists an AC-unifier θ of $x_1\sigma' = v_{p+1,k_i,1} \wedge \ldots \wedge x_n\sigma' = v_{p+1,k_i,n}$ such that $\sigma_h = \theta|_{\vec{w}}$, where $\vec{w} = Var(\overrightarrow{v_{p+1,k_i}})$.

But σ' is also a generator of solutions of ϕ_{pk}, and since $v_{p+1,k_i,j} = v_{pkj}\{w \mapsto f(\vec{w'})\}$ for $1 \leq j \leq n$, the ground assignment ρ that coincides with σ everywhere except on w where $w\rho = f(\vec{w'})\sigma$, is an AC-instance of some σ_h of ϕ_{pk} and by the induction hypothesis it is a solution of ψ_{pk}. Then σ is a solution of $\psi_{pk}\{w \mapsto f(\vec{w'})\} \equiv \psi_{p+1,k_i}$. □

Note that the previous proof not only shows that ϕ is AC-E_U if and only if ψ is, but also shows that in case ϕ is AC-E_U, the first step towards obtaining a positive formula equivalent to ϕ is eliminating negation from its kernel ψ.

Another useful observation towards obtaining an algorithm for negation elimination is the following: If $\phi \to_{\mathbf{T}} \bot$ then $\phi \sim_{AC} \bot$ and we are finished, but if $\phi \to_{\mathbf{T}} \top$, i.e. if ϕ has solutions, then it must be rule T_2 which applies (not T_0 because at least one of the u_{kl} is not ground). Then, since the condition of rule T_2 holds, there exists $L = \{\lambda_1, \ldots, \lambda_h\} \subseteq S_{\vec{w},A(\psi)}$ such that for $1 \leq i \leq h$, $T(\mathcal{F})/_{=_{AC}} \not\models \exists \vec{y} : \bigvee_{k \in K}(\bigwedge_{l \in L_k} w_{kl}\lambda_i = u_{kl})$. We will use L to eliminate negation from ψ (note that the substitutions in L are generators of solutions of ψ).

Before presenting the formal description of the negation elimination algorithm, let us illustrate the ideas behind it with an example.

EXAMPLE 3.2. *Let $\mathcal{F} = \{+, a, b, c, d, e\}$ where $+$ is AC and the other symbols are constants, and let us consider a formula having the AC-E_U property:*

$$\psi \equiv \forall y : w \neq a + a + y \wedge w \neq b + b + y \wedge w \neq c + c + y \wedge w \neq d + d + y \wedge w \neq e + e + y.$$

In this example $d = 2$ and $\kappa + 1 = 4$.

$$\begin{aligned} A(\psi) = \{\ & a, b, c, d, e, \\ & a+a, a+b, a+c, a+d, a+e, b+b, b+c, b+d, b+e, \ldots \\ & a+a+a, a+a+b, a+a+c, a+a+d, a+a+e, a+b+b, a+b+c, a+b+d, \ldots \\ & a+a+a+a, a+a+a+b, \ldots \\ \}\ . \end{aligned}$$

$S_{w,A(\psi)}$ is the set of all the substitutions with domain $\{w\}$ and image in $A(\psi)$. Note that all the terms in $A(\psi)$ are ground since there are no terms of depth 2 in $T(\mathcal{F}, \mathcal{X})$; then all the substitutions in $S_{w,A(\psi)}$ are ground. The substitutions of $S_{w,A(\psi)}$ for which

$$T(\mathcal{F})/_{=_{AC}} \models \not\exists y : w \neq a+a+y \wedge w \neq b+b+y \wedge w \neq c+c+y \wedge w \neq d+d+y \wedge w \neq e+e+y$$

are:

$$\begin{aligned} L = \{\ & \{w \mapsto P\} \mid P \in \{a, b, c, d, e\}\} \cup \\ & \{\{w \mapsto P + Q\} \mid P, Q \in \{a, b, c, d, e\}\} \cup \\ & \{\{w \mapsto P + Q + R\} \mid P, Q, R \in \{a, b, c, d, e\},\ \textit{and } P, Q, R \textit{ are pairwise distinct}\} \cup \end{aligned}$$

$\{\{w \mapsto P + Q + R + S\} \mid P, Q, R, S \in \{a, b, c, d, e\},$
and P, Q, R, S *are pairwise distinct*$\}$.

Since $L \neq \emptyset$, ψ *has solutions in* $T(\mathcal{F})/_{=_{AC}}$.

We are faced now with the problem of deciding whether ψ is *AC*-E_U or not, and in case it is, finding a finite complete set of generators for ψ. We will use L to this purpose.

Although the substitutions in L are generators of ψ, L will not be a *complete* set of generators of ψ in general. It is clear in the previous example that the substitutions in L cannot generate a solution with more than four arguments, as, for instance $\{w \mapsto a + b + c + d + e\}$ (which *is* a solution of ψ). At first glance one could think that to overcome this incompleteness problem it is sufficient to add new variables as arguments in all the *AC* symbols with $\kappa + 1$ arguments ($\kappa + 1$ is the bound used in the construction of $S_{\vec{w},A(\psi)}$). We would obtain in this way a set of substitutions which is complete, but nothing guarantees that those substitutions will generate *only* solutions of ψ. In case they do, i.e. in case they are generators, the problem is solved, but this will not be the case in general. What we will do to solve the problem is to consider a bigger test set (i.e. with $\kappa' > \kappa$). For the formula in the example 3.2, it is easy to see that the solutions that are missing in L are recovered if we take $\kappa' = 4$ instead of $\kappa = 3$: we obtain a new set L' of substitutions which is a complete set of generators of ψ by repeating the construction above until $\kappa' + 1$ (i.e. 5) arguments of *AC* symbols.

Let us formalize these ideas. Let $\psi \equiv \forall \vec{y} : \bigwedge_{k \in K}(\bigvee_{l \in L_k} w_{kl} \neq u_{kl})$ be the kernel of a $\forall$-simple formula ϕ, and assume there exists $L = \{\lambda_1, \ldots, \lambda_h\} \subseteq S_{\vec{w},A(\psi)}$ such that for $1 \leq i \leq h$, $T(\mathcal{F})/_{=_{AC}} \not\models \exists \vec{y} : \bigvee_{k \in K}(\bigwedge_{l \in L_k} w_{kl}\lambda_i = u_{kl})$. Let $\vec{z_i} = Var(Im(\lambda_i))$. Since the variables in $Im(\lambda_i)$ appear at positions strictly deeper than those in u_{kl}, and each variable in u_{kl} appears at most once in a conjunction, for $1 \leq i \leq h$:

$$(1) \qquad T(\mathcal{F})/_{=_{AC}} \not\models \exists \vec{z_i}, \exists \vec{y} : \bigvee_{k \in K} \Big(\bigwedge_{l \in L_k} w_{kl}\lambda_i = u_{kl} \Big).$$

Let us add a different sub-index to each occurrence of an *AC* symbol with $\kappa + 1$ arguments in $Im(\lambda_1), \ldots, Im(\lambda_h)$. If $+_{i_1}, \ldots, +_{i_m}$ occur in $Im(\lambda_i)$, we define λ_i^I, for all $I \subseteq \{i_1, \ldots, i_m\}$, to be the same as λ_i but for the subterms whose root is $+_{i_j}$ with $i_j \in I$, where there is one more argument: a new variable x_{i_j}. Let us call L^* the set $\{\lambda_i^I \mid \lambda_i \in L, I \subseteq \{i_1, \ldots, i_m\}\}$ of (flattened) substitutions, and let $\vec{z_i^I} = Var(Im(\lambda_i^I))$. As we said before, two cases are possible: Either there is no $\lambda_i^I \in L^*$ such that

$$(2) \qquad T(\mathcal{F})/_{=_{AC}} \models \exists \vec{z_i^I}, \exists \vec{y} : \bigvee_{k \in K} \Big(\bigwedge_{l \in L_k} w_{kl}\lambda_i^I = u_{kl} \Big)$$

and then $L \cup L^*$ is a complete set of generators for ψ (its correctness and completeness will be proved in Theorem 3.3), or some λ_i^I satisfies (2). In the latter case, we will prove, also in Theorem 3.3 below, that there exists $\kappa' > \kappa$ such that if we construct $A(\psi)$ using κ' instead of κ and then construct $S_{\vec{w},A(\psi)}$, L and L^* in the same way as before but using the new $A(\psi)$ (we will call NL, NL^* the new L, L^* thus obtained), then

- ψ is *AC*-E_U iff there is no ${\nu_i}^I \in NL^*$ such that (2) holds,
- if ψ is *AC*-E_U then $NL \cup NL^*$ is a complete set of generators for ψ.

Let us explain now how this κ' will be computed. First of all we have to define the notion of "correspondence" between arguments of AC symbols.

DEFINITION 3.5. *Let $+$ be an AC symbol and t, u be flattened terms $t \equiv +(t_1, \ldots, t_n)$ and $u \equiv +(u_1, \ldots, u_m)$. An AC-*correspondence $\varsigma : [1, \ldots, n] \to [1, \ldots, m]$ between t and u *is a partial injective function such that*

- *for all $i \in Dom(\varsigma)$, t_i and $u_{\varsigma(i)}$ are AC-unifiable and*
- *ς is maximal, i.e., there is no $\varsigma' : [1, \ldots, n] \to [1, \ldots, m]$ such that, for all $i \in Dom(\varsigma')$, t_i and $u_{\varsigma'(i)}$ are AC-unifiable and $|\, Dom(\varsigma')| > |\, Dom(\varsigma)|$.*

EXAMPLE 3.3. *Let $\mathcal{F} = \{f, a, +\}$, where $+$ is AC, f is a unary function and a is a constant. Let $t = y{+}a{+}a$, $u = f(z_1){+}f(z_2){+}f(z_3){+}a$. There are six AC-correspondences between t and u: three of them assign 2 to 4, and are not defined on 3 (they differ only in the image of 1, which can be 1, 2, or 3); the other three assign 3 to 4, and are not defined on 2 (again they differ in the image of 1, which can be 1, 2, or 3).*

Note that all the AC-correspondences in the previous example have the same cardinality. This is a general property:

PROPERTY 3.4. *Given t, u, there exists n such that for all AC-correspondences ς between t and u, $|\, Dom(\varsigma)| = n$.*

PROOF. By maximality of AC-correspondences. □

We will first give the intuitive ideas that suggest how to compute κ', and then the formal definitions. The following observation is essential: if in the disequation $t \neq u$ where $t = +(t_1, \ldots, t_n)$, $u = +(u_1, \ldots, u_m)$ and $+$ is an AC symbol, there is a t_i without a correspondent in u (for any correspondence between t and u), then there is a solution of $t \neq u$. Moreover, also the disequations $+(t_1, \ldots, t_n, t_i) \neq u$, $+(t_1, \ldots, t_n, t_i, t_i) \neq u$, etc, have solutions. In other words, t_i can be added an arbitrary number of times without affecting the solvability of the disequation. This can be seen in the Example 3.3: a can be added an arbitrary number of times in t.

Intuitively, we want κ' to be big enough so that if for a substitution $\nu \in NL$ (recall that NL contains only generators of ψ) some $w_{kl}\nu$ has an AC-symbol $+_{ij}$ with $\kappa' + 1$ arguments, then one of these arguments does not correspond with any of the arguments of $+_{ij}$ in u_{kl}. If this happens, the argument can be added an arbitrary number of times in $w_{kl}\nu$ and the resulting substitutions are still generators of ψ. Now the idea is that in this case the only possible finite representation of these solutions of ψ is ν^*, which has a variable as an argument of $+_{ij}$. Let us develop this idea.

We know that (2) is equivalent to

$$\exists \vec{z_i^l}, \exists \vec{y} : \bigvee_{k \in K} \Big(\bigwedge_{l \in L_k} \text{flat}(w_{kl}\lambda_i^l) \Big) =_P \text{flat}(u_{kl}))$$

and (1) is equivalent to

$$\neg(\exists \vec{z_i}, \exists \vec{y} : \bigvee_{k \in K} \Big(\bigwedge_{l \in L_k} \text{flat}(w_{kl}\lambda_i) =_P \text{flat}(u_{kl}))).$$

Let us compare these formulae. If we have an equality with λ_i^I but not with λ_i it is because some arguments of u_{kl} which do not have a correspondent in $w_{kl}\lambda_i$ are covered by a variable in λ_i^I. In other words, there exists $k \in K$ such that for all $l \in L_k$ there are permutations $\overline{\text{flat}(w_{kl}\lambda_i)}$, $\overline{\text{flat}(u_{kl})}$ of $\text{flat}(w_{kl}\lambda_i)$, $\text{flat}(u_{kl})$ respectively, such that all non variable positions of $\overline{\text{flat}(w_{kl}\lambda_i)}$ coincide with those of $\overline{\text{flat}(u_{kl})}$ but there are some remaining arguments under some AC symbols $+_{i_j}$ ($i_j \in I$) without a correspondent. The number n_{ij} of such arguments is computed as follows: if $+_{i_j}$ is an AC symbol with $\kappa+1$ arguments that occurs at position q in $\overline{\text{flat}(w\lambda_i)}$ and $\overline{\text{flat}(u_{kl})}$ (for all k, l such that $w_{kl} \equiv w$), that is, $\overline{\text{flat}(w\lambda_i)}|_q \equiv +_{i_j}(t_1 \ldots, t_{\kappa+1})$ and $\overline{\text{flat}(u_{kl})}|_q \equiv +_{i_j}(s_1, \ldots, s_{r_{kl}})$, and for any AC-correspondence ς between $\overline{\text{flat}(u_{kl})}|_q$ and $\overline{\text{flat}(w_{kl}\lambda_i)}|_q$, $|\,Dom(\varsigma)| = c_{kl}$, then

$$n_{ij} = \sum_{k,l \text{ s.t. } w_{kl} \equiv w} r_{kl} - c_{kl}.$$

Let $N_{ij} = n_{ij} - |\{w_{kl} \mid w_{kl} \equiv w\}| + 1$ and let $N = \max_{i,j}\{N_{ij}\}$. We take $\kappa' = \kappa + N$.

Example 3.4. *Let us calculate κ' for the problem in Example* 3.2. *We have to consider the set of substitutions:*

$\{\{w \mapsto P + Q + R + S\} \mid P, Q, R, S \in \{a, b, c, d, e\}$ *and* P, Q, R, S *are pairwise different*$\} = \{\lambda_1 : \{w \mapsto a+b+c+d\}, \lambda_2 : \{w \mapsto a+b+c+e\}, \lambda_3 : \{w \mapsto a+c+d+e\}, \lambda_4 : \{w \mapsto a+b+d+e\}, \lambda_5 : \{w \mapsto b+c+d+e\}\}$.

Each one of these substitutions has one AC symbol with $\kappa+1$ arguments (recall that in this example, $w_k \equiv w$ and $r_k = 3$ for all k). Now, for λ_1, $c_1 = c_2 = c_3 = c_4 = 2$ and $c_5 = 1$ then $n_{11} = 6$ and $N_{11} = 6 - 5 + 1 = 2$. For λ_2, $c_1 = c_2 = c_3 = c_5 = 2$ and $c_4 = 1$ then $n_{21} = 6$ and $N_{21} = 2$. In the same way, $N_{31} = N_{41} = N_{51} = 2$. Then $N = 2$ and $\kappa' = 5$.

A detailed example of application of the algorithm is given in Section 3.5.

Let NL be the set of substitutions computed for ψ by using κ' instead of κ. By definition, $NL = \{\nu_1, \ldots, \nu_{h'}\}$ such that for $1 \le i \le h'$,

$$(3) \qquad T(\mathcal{F})/_{=_{AC}} \not\models \exists \vec{z_i}, \exists \vec{y} : \bigvee_{k \in K} \Big(\bigwedge_{l \in L_k} w_{kl}\nu_i = u_{kl} \Big)$$

where $\vec{z_i} = Var(Im(\nu_i))$.

Still we have to prove that $NL \cup NL^*$ is a complete set of generators for ψ when such a set exists. The following properties of NL will be used in the proof.

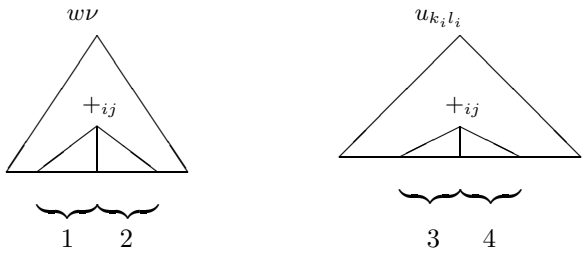
Lemma 3.5. *If $\nu \in NL$ then* $\text{proj}_{\kappa+1}(\nu) \in L$.

Proof. By construction. □

Lemma 3.6. *If there is $\nu \in NL$ such that an AC symbol occurs in $Im(\nu)$ with $\kappa'+1$ arguments then there is a solution of ψ with $\kappa'+1+i$ arguments for any $i > 0$.*

Proof. By Lemma 3.5, if $\nu \in NL$ then $\text{proj}_{\kappa+1}(\nu) \in L$. Let us define $\lambda \equiv \text{proj}_{\kappa+1}(\nu)$. Since $\nu \in NL$, by (3) there exists a variable w and an AC-symbol $+_{ij}$ with $\kappa'+1$ arguments in $w\nu$ such that some arguments of $u_{k_1 l_1}, \ldots, u_{k_J l_J}$ (where $w_{k_i l_i} \equiv w$, $J = |\{w_{kl} \mid$

$w_{kl} \equiv w\}|$) which did not have a correspondent in $w\lambda$ still don't have a correspondent in $w\nu$. This situation is depicted in the following picture.



1. Arguments in $w\lambda$.
2. N more arguments.
3. Arguments with a correspondent in $w\lambda$.
4. Arguments without a correspondent in $w\lambda$ (at least one).

Assume each $u_{k_i l_i}$ has m_i more arguments with a correspondent in $w\nu$ than in $w\lambda$. By definition of n_{ij}, $m_1 + \cdots + m_J \leq n_{ij} - J$ since in each $u_{k_i l_i}$ there is still an argument without a correspondent [by (3)]. Also by definition, $n_{ij} - J \leq N - 1$. Hence, in the worst case there is still an argument of $+_{ij}$ in $w\nu$ which does not unify with any of the arguments that were without a correspondent in the u_{kl}'s. Then we can add this argument an arbitrary number of times. □

Now the idea is that either $NL \cup NL^*$ is a complete set of generators of ψ, or there is no finite complete set of generators for ψ. In other words, the only finite representation of the infinite solutions that are shown in the previous lemma is $NL \cup NL^*$. Formally, we will show that if this set is not correct, i.e. if some instances of substitutions in this set are not solutions of ψ, then there is no finite set of generators for ψ (ψ is not AC-E_U).

First we define a set of substitutions that will be empty when ψ is not AC-E_U and will be shown to be a complete set of generators in case such a set exists.

Definition 3.6. *The set G of substitutions is defined by cases:*

- $G = L \cup L^*$ *if there is no* $\lambda \in L^*$ *such that* (2) *holds.*
- *Otherwise,* $G = NL \cup NL^*$ *if there is no* $\nu \in NL^*$ *such that* (2) *holds.*
- *Otherwise,* $G = \emptyset$.

For instance, for the problem in Example 3.2, $G = NL \cup NL^*$. In Section 3.5 we show an example where $G = \emptyset$.

Now we are ready to prove the main result of this section.

Theorem 3.3. *ψ is AC-E_U if and only if $G \neq \emptyset$. If*

$$G = \{\lambda_1, \ldots, \lambda_n\} \neq \emptyset \text{ and } \bigcup_i Var(Im(\lambda_i)) = \vec{z} \text{ then } \psi \sim_{AC} \bigvee_{\lambda_i \in G} \exists \vec{z} : \bigwedge_{w_j \in Dom(\lambda_i)} w_j = w_j \lambda_i.$$

PROOF. First we will prove that if $G \neq \emptyset$ then ψ is AC-E_U, and more precisely,

$$\psi \sim_{AC} \bigvee_{\lambda_i \in G} \exists \vec{z} : \bigwedge_{w_j \in Dom(\lambda_i)} w_j = w_j \lambda_i$$

i.e. G is a complete set of AC-generators of ψ when $G \neq \emptyset$.

Correctness: We must show that for all $\lambda \in G$ and for all ground assignment μ, $T(\mathcal{F})/_{=_{AC}} \not\models \exists \vec{y} : \bigvee_{k \in K}(\bigwedge_{l \in L_k} w_{kl}\lambda\mu = u_{kl})$. By contradiction:

If

$$(*) \qquad T(\mathcal{F})/_{=_{AC}} \models \exists \vec{y} : \bigvee_{k \in K} \Big(\bigwedge_{l \in L_k} w_{kl}\lambda\mu = u_{kl} \Big)$$

then, $T(\mathcal{F})/_{=_{AC}} \models \exists\, Var(Im(\lambda)), \exists \vec{y} : \bigvee_{k \in K}(\bigwedge_{l \in L_k} w_{kl}\lambda = u_{kl})$, then, by (1) and (3), $\lambda \notin L$ and $\lambda \notin NL$. Thus, $\lambda \in L^*$, but $(*)$ implies that (2) holds, which contradicts the fact that $G \neq \emptyset$.

Completeness: We must show that for any AC solution μ of ψ there exists $\lambda \in G$ such that μ is an AC-instance of λ. Let μ be a solution of ψ: $T(\mathcal{F})/_{=_{AC}} \not\models \exists \vec{y} : \bigvee_{k \in K}(\bigwedge_{l \in L_k} w_{kl}\mu = u_{kl})$. Let us define $\mu' \equiv \mathrm{proj}_{\kappa+1}(\mu)$ in case $G = L \cup L^*$, and $\mu' \equiv \mathrm{proj}_{\kappa'+1}(\mu)$ in case $G = NL \cup NL^*$.

1. If $\mu' \equiv \mu$ then there exists $\lambda \in L$ or $\lambda \in NL$ such that $\lambda \preceq_{AC} \mu$, because if $\max\{\mathrm{depth}(t) \mid t \in Im(\mu')\} \leq d$ then $\mu \in L$ or $\mu \in NL$, and if $\max\{\mathrm{depth}(t) \mid t \in Im(\mu')\} > d$ then, by definition of $S_{\vec{w},A(\psi)}, S_{\vec{w},A'(\psi)}$, there exists $\lambda \in S_{\vec{w},A(\psi)}$ or $\lambda \in S_{\vec{w},A'(\psi)}$ which coincides with μ till depth $d-1$ and has variables at depth d (i.e., $\lambda \preceq_{AC} \mu$), and $T(\mathcal{F})/_{=_{AC}} \not\models \exists \vec{y} : \bigvee_{k \in K}(\bigwedge_{l \in L_k} w_{kl}\lambda = u_{kl})$ (i.e., $\lambda \in G$) because if there exists θ such that $\bigvee_{k \in K}(\bigwedge_{l \in L_k} w_{kl}\lambda =_{AC} u_{kl}\theta)$, then, since $\mu = \lambda\alpha$, $\bigvee_{k \in K}(\bigwedge_{l \in L_k} w_{kl}\mu = u_{kl}\theta\alpha)$.
2. If $\mu \not\equiv \mu'$, as a consequence of Lemma 3.5, μ' is also a solution, then, using 1., there exists $\lambda \in L$ if $G = L \cup L^*$ or $\lambda \in NL$ if $G = NL \cup NL^*$, such that $\lambda \preceq_{AC} \mu'$. Then, there is $\lambda^* \in L^*$ or $\lambda^* \in NL^*$ such that $\lambda^* \preceq_{AC} \mu$.

Now it remains to prove that if $G = \emptyset$ then ψ is not AC-E_U. We will prove this by contradiction. Assume $G = \emptyset$ and ψ is AC-E_U. Then, since AC-unification is finitary, there exists a complete set $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$ of generators of ψ. Since $G = \emptyset$, there is $\lambda^I \in L^*$ and $\nu^{I'} \in NL^*$ such that (2) holds. This implies that there exists a ground assignment μ' such that

$$(4) \qquad \exists \vec{y} : \bigvee_{k \in K} \Big(\bigwedge_{l \in L_k} w_{kl}\nu^{I'}\mu' =_{AC} u_{kl} \Big).$$

But if $\nu^{I'} \in NL^*$ then $\nu \in NL$, that is, all the instances of ν are solutions of ψ, in particular $\nu\mu'$. Besides, if $\nu^{I'} \in NL^*$, $Im(\nu)$ has an occurrence of an AC symbol with $\kappa'+1$ arguments and by Lemma 3.6, there is an argument of an AC symbol $+_j$ that we can add n times for any $n > 0$ yielding always a solution, which we denote by $(\nu\mu')^n$.

Since Γ is finite, there exists γ_i such that infinitely many $(\nu\mu')^n$ are AC instances of γ_i, that is, there are infinitely many ρ_n such that $\gamma_i\rho_n =_{AC} (\nu\mu')^n$.

If $+_j$ occurs in $w_{kl}\nu$,

$$w_{kl}\gamma_i\rho_n =_{AC} w_{kl}(\nu\mu')^n$$

for infinitely many n.

We assume that all these substitutions are flattened, then for infinitely many n:

$$\text{flat}(w_{kl}\gamma_i\rho_n) =_P \text{flat}(w_{kl}(\nu\mu')^n).$$

Then, they can be made identical by permutations.

$$\overline{\text{flat}(w_{kl}\gamma_i\rho_n)} \equiv \overline{\text{flat}(w_{kl}(\nu\mu')^n)}.$$

Assume $+_j$ occurs at position p in $\overline{\text{flat}(w_{kl}(\nu\mu')^n)}$. Then either $\exists q \leq p$ such that $\overline{w_{kl}\gamma_i}|_q \equiv x \in \mathcal{X}$ or $\overline{w_{kl}\gamma_i}(p) = +_j$ and $\exists q$ such that $|q| = 1$ and $\overline{w_{kl}\gamma_i}(p.q) = x \in \mathcal{X}$, and x occurs only once in $Im(\gamma_i)$ since we are adding arguments only to the $+_j$ occurring at position p. Then, in both cases $\gamma_i \preceq_{AC} \nu^{I'}\mu'$. But $\nu^{I'}\mu'$ is not a solution of ψ by (4). This contradicts the fact that Γ is a set of generators of ψ. □

As a consequence of Theorem 3.3 *negation elimination is decidable for linear AC complement problems*. We can write the test stated in Theorem 3.3 as a transformation rule to be added to R_{EAC}, see Figure 4. The system $R_{EAC} \cup \{T_{NE}\}$ is obviously terminating. When applied to a linear AC complement problem P, it yields a positive formula equivalent to P whenever such a formula exists, and a finite disjunction of $\forall$-simple formulae equivalent to P in case P is not AC-E_U.

Test for Negation Elimination (TNE)

$$(T_{NE}) \quad P \equiv (\exists \vec{w} \forall \vec{y} : \bigwedge_{k \in K} \Big(\bigvee_{l \in L_k} w_{kl} \neq u_{kl} \Big) \wedge x_1 = v_1 \wedge \ldots \wedge x_n = v_n) \vee d$$
$$\rightarrow \Big(\exists \vec{w} : \Big(\bigvee_{\lambda_i \in G} \exists \vec{z} : \bigwedge_{w_j \in Dom(\lambda_i)} w_j = w_j\lambda_i \Big) \wedge x_1 = v_1 \wedge \ldots \wedge x_n = v_n \Big) \vee d.$$

If P is irreducible by R_{EAC} and $G \neq \emptyset$.

Figure 4. Negation elimination in $\forall$-simple formulae.

3.5. A DETAILED EXAMPLE

In Section 3.4 we showed an example of a formula where negation can be eliminated (Example 3.2). Let us show now a simple example of an AC complement problem where negation cannot be eliminated.

Let $\mathcal{F} = \{a, f, +\}$ where $+$ is AC and $P \equiv \forall y : x \neq y + a + a$. This problem is already irreducible by R_{EAC}. It is a $\forall$-simple formula (in fact it is the kernel of the $\forall$-simple formula $\exists w, \forall y : w \neq y + a + a \wedge x = w$), and it has solutions. The values of κ and d are 3 and 2 respectively.

We now construct the test set:

$$\begin{aligned} A(P) = \{\ & a, \\ & f(a), f(f(z)), \\ & a + a, a + a + a, a + a + a + a, \\ & f(z) + a, f(z) + a + a, f(z) + a + a + a, \end{aligned}$$

$$
\begin{array}{l}
f(z_1)+f(z_2), f(z_1)+f(z_2)+a, f(z_1)+f(z_2)+a+a,\\
f(z_1)+f(z_2)+f(z_3), f(z_1)+f(z_2)+f(z_3)+a,\\
f(z_1)+f(z_2)+f(z_3)+f(z_4)
\end{array}
\}\,.
$$

Note that in $A(P)$ we have not written the terms that are AC-instances of terms in $A(P)$.

We do not write explicitly all the substitutions in $S_{x,A(P)}$, but only those in L:

$$
\begin{aligned}
L = \{\ & \lambda_1 = \{x \to a\}\\
& \lambda_2 = \{x \to f(a)\}\\
& \lambda_3 = \{x \to f(f(z))\}\\
& \lambda_4 = \{x \to a+a\}\\
& \lambda_5 = \{x \to f(z)+a\}\\
& \lambda_6 = \{x \to f(z_1)+f(z_2)\}\\
& \lambda_7 = \{x \to f(z_1)+f(z_2)+a\}\\
& \lambda_8 = \{x \to f(z_1)+f(z_2)+f(z_3)\}\\
& \lambda_9 = \{x \to f(z_1)+f(z_2)+f(z_3)+a\}\\
& \lambda_{10} = \{x \to f(z_1)+f(z_2)+f(z_3)+f(z_4)\}\\
\}\,.&
\end{aligned}
$$

In $Im(\lambda_9)$ and $Im(\lambda_{10})$ there are AC symbols with $\kappa+1$ arguments. Then we must build L^*:

$$
\begin{aligned}
L^* = \{\ & \lambda_9^1 = \{x \to f(z_1)+f(z_2)+f(z_3)+a+z_5\}\\
& \lambda_{10}^2 = \{x \to f(z_1)+f(z_2)+f(z_3)+f(z_4)+z_6\}\\
\}\,.&
\end{aligned}
$$

Here, $f(z_1)+f(z_2)+f(z_3)+a+z_5$ and $y+a+a$ are AC-unifiable, and so are $f(z_1)+f(z_2)+f(z_3)+f(z_4)+z_6$ and $y+a+a$, but neither $f(z_1)+f(z_2)+f(z_3)+a$ and $y+a+a$ nor $f(z_1)+f(z_2)+f(z_3)+f(z_4)$ and $y+a+a$ are. In the first case the number n_{11} of arguments of $y+a+a$ without a correspondent in $f(z_1)+f(z_2)+f(z_3)+a$ is 1, and in the second case, the number n_{21} of arguments of $y+a+a$ without a correspondent in $f(z_1)+f(z_2)+f(z_3)+f(z_4)$ is 2. Then $N_{11}=1-1+1=1$ and $N_{21}=2-1+1=2$, hence $N=2$.

After constructing $A'(P)$ (with $\kappa'=\kappa+N=5$) and $S_{x,A'(P)}$ the substitutions in NL are:

$$
\begin{aligned}
NL = \{\ & \nu_1 = \{x \to a\}\\
& \nu_2 = \{x \to f(a)\}\\
& \nu_3 = \{x \to f(f(z))\}\\
& \nu_4 = \{x \to a+a\}\\
& \nu_5 = \{x \to f(z)+a\}\\
& \nu_6 = \{x \to f(z_1)+f(z_2)\}\\
& \nu_7 = \{x \to f(z_1)+f(z_2)+a\}\\
& \nu_8 = \{x \to f(z_1)+f(z_2)+f(z_3)\}
\end{aligned}
$$

$$\begin{aligned}
&\nu_9 = \{x \to f(z_1) + f(z_2) + f(z_3) + a\}\\
&\nu_{10} = \{x \to f(z_1) + f(z_2) + f(z_3) + f(z_4)\}\\
&\nu_{11} = \{x \to f(z_1) + f(z_2) + f(z_3) + f(z_4) + a\}\\
&\nu_{12} = \{x \to f(z_1) + f(z_2) + f(z_3) + f(z_4) + f(z_5)\}\\
&\nu_{13} = \{x \to f(z_1) + f(z_2) + f(z_3) + f(z_4) + f(z_5) + a\}\\
&\nu_{14} = \{x \to f(z_1) + f(z_2) + f(z_3) + f(z_4) + f(z_5) + f(z_6)\}\\
\}\,.
\end{aligned}$$

In $Im(\nu_{13})$ and $Im(\nu_{14})$ there are AC symbols with $\kappa' + 1$ arguments. Then we must build NL^*. But when we add a new variable to $x\nu_{13}$ and to $x\nu_{14}$, they become AC-unifiable with $y + a + a$. Then $G = \emptyset$, which means that P is not AC-E_U: negation cannot be eliminated from P.

4. Extensions to Restricted Cases of Non-Linearity

The algorithms presented in the previous sections were proved correct, complete and terminating for linear AC-complement problems. It is easy to see that the decision test for satisfiability fails to be complete if the problem is not linear. The aim of this section is to advance towards a solution of the satisfiability problem for AC-complement problems in general. For this, we will express a non-linear AC-complement problems as a combination of a linear problem with other equational formulas. Then, we will use the previous results to deduce the restrictions we have to impose to the original non-linear problem in order to obtain a combination whose solutions we can compute.

Let

$$\mathcal{C} \equiv \forall \vec{y} : t \neq t_1 \wedge \ldots \wedge t \neq t_n$$

be a non-linear AC-complement problem (without loss of generality we assume that for all $i \neq j$, $Var(t_i) \cap Var(t_j) = \emptyset$), and let

$$\mathcal{L} \equiv \forall \vec{y'} : t \neq t'_1 \wedge \ldots \wedge t \neq t'_n$$

be the linearized version of $\mathcal{C}$, that is, $\mathcal{L}$ is obtained from $\mathcal{C}$ by renaming its non-linear universally quantified variables in the following way: if $y_i \in \vec{y}$ occurs m times in $t_i \in \mathcal{C}$ then we will rename the occurrences of y_i by fresh variables $y'_{i1}, \ldots, y'_{im}$.

Example 4.1. *Let $\mathcal{F} = \{a, f, +\}$ where a is a constant, f a unary function symbol, and $+$ an AC symbol.*

Let $\mathcal{C} = \forall y : x_1 + x_1 + x_2 \neq y + y + a$.

$\mathcal{C}$ has solutions: for instance, $\sigma = \{x_1 \mapsto a, x_2 \mapsto f(z)\}$ is a generator. The linearized version of $\mathcal{C}$ is

$$\mathcal{L} = \forall y_1, y_2 : x_1 + x_1 + x_2 \neq y_1 + y_2 + a$$

$\mathcal{L}$ is a linear complement problem (recall that free variables can be non-linear). Note that σ is not a generator for $\mathcal{L}$.

A first step towards our goal is to describe precisely the relation between the solutions of $\mathcal{L}$ and the solutions of $\mathcal{C}$. In fact, it is easy to see that the solutions of $\mathcal{L}$ are also solutions of $\mathcal{C}$, but $\mathcal{C}$ can have more solutions. We will have to "complete" $\mathcal{L}$ in order

to obtain a formula with the same set of solutions as $\mathcal{C}$. We will show this first for the previous example: In order to recover the solutions of $\mathcal{C}$ in Example 4.1, it is enough to consider

$$\mathcal{L}' = \forall y_1, y_2 : x_1 + x_1 + x_2 \neq y_1 + y_2 + a \vee y_1 \neq y_2$$

then σ is a generator for $\mathcal{L}'$.

In general, what we have to do is to replace each disequation

$$t \neq t'_j$$

in $\mathcal{L}$ by the disjunction

$$t \neq t'_j \vee \bigvee_{1 \leq i \leq m} Y_{ji}$$

where $y_1, \ldots, y_m$ are the universally quantified variables of t_j and y'_{ik} are the variables introduced by the linearization of y_i, and

$$Y_{ji} = \bigvee_{y'_{ih} \not\equiv y'_{il} \in \vec{y'_i}} y'_{ih} \neq y'_{il}.$$

This can be done in a very simple way if $n = 1$: in this case

$$\mathcal{C} \sim \forall \vec{y'} : t \neq t' \vee \bigvee_{1 \leq i \leq m} Y_i.$$

Now we will apply this technique to the general case. In the following, to abbreviate the formulae, we will assume that the variables of t'_j are $y_1, \ldots, y_{m_j}$, and we will use the notation

$$\bigvee_{1 \leq i \leq |Var(t'_j)|} Y_{ji}$$

to represent the disjunction of disequations of the form $y'_{ih} \neq y'_{il}$ for all the pairs y'_{ih}, y'_{il} of different variables introduced by the linearization of each variable y_i in t_j. In general:

$$\mathcal{C} \sim \forall \vec{y'} : (t \neq t'_1 \vee \bigvee_{1 \leq i \leq |Var(t'_1)|} Y_{1i}) \wedge \ldots \wedge (t \neq t'_n \vee \bigvee_{1 \leq i \leq |Var(t'_n)|} Y_{ni})$$

and using the distributivity laws of $\wedge$,$\vee$ we obtain

$$\mathcal{C} \sim \mathcal{D} \equiv \forall \vec{y'} : (t \neq t'_1 \wedge \ldots \wedge t \neq t'_n) \vee \Big(\Big(\bigvee_{1 \leq i \leq |Var(t'_1)|} Y_{1i} \Big) \wedge t \neq t'_2 \wedge \ldots \wedge t \neq t'_n \Big) \vee \ldots$$
$$\vee \Big(\Big(\bigvee_{1 \leq i \leq |Var(t'_1)|} Y_{1i} \Big) \wedge \ldots \wedge \Big(\bigvee_{1 \leq i \leq |Var(t'_n)|} Y_{ni} \Big) \Big).$$

The relation between the solutions of $\mathcal{L}$ and the solutions of $\mathcal{C}$ becomes explicit in $\mathcal{D}$.

Since $\mathcal{L}$ is a linear AC-complement problem, its satisfiability is decidable. Let $S_{\vec{w},A(\psi)}$ be the test set for $\mathcal{L}$ (without loss of generality we can assume that $\mathcal{L}$ is a $\forall$-simple formula, because any linear complement problem is equivalent to a finite disjunction of $\forall$-simple formulas by Theorem 3.1). If $\mathcal{L}$ has solutions then so has $\mathcal{C}$, but the interesting case is when $\mathcal{C}$ has solutions but $\mathcal{L}$ does not. In this case, by Theorem 3.2, for all $\lambda \in S_{\vec{w},A(\psi)}$, there exists an AC-matcher σ of $t\lambda = t'_1 \vee \ldots \vee t\lambda = t'_n$, that is, $t\lambda =_{AC} t'_1\sigma \vee \ldots \vee t\lambda =_{AC} t'_n\sigma$. Moreover, if we consider $S^*_{\vec{w},A(\psi)}$ defined as $S_{\vec{w},A(\psi)}$ but including also substitutions

with a variable as (κ+2)th argument of AC symbols, it is still true that for all $\lambda \in S^*_{\vec{w},A(\psi)}$, there exists an AC-matcher σ of $t\lambda = t'_1 \vee \ldots \vee t\lambda = t'_n$, that is,

$$(1) \qquad t\lambda =_{AC} t'_1\sigma \vee \ldots \vee t\lambda =_{AC} t'_n\sigma$$

(this is because each variable appears at most once in the problem).

For $1 \leq j \leq n$ and $\lambda \in S^*_{\vec{w},A(\psi)}$, let $M_{j\lambda} = \{\sigma_i \mid i \in I_{j\lambda}\}$ be a minimal complete set of AC-matchers of $t\lambda = t'_j$. It is easy to see that if $\mathcal{L}$ does not have solutions but $\mathcal{C}$ does, then there is some $\lambda \in S^*_{\vec{w},A(\psi)}$ and a ground assignment μ, such that for $1 \leq j \leq n$, $\bigwedge_{\sigma\in M_{j\lambda}}(\bigvee_{1\leq i\leq|Var(t'_j)|}(\bigvee_{y'_{ih}\not\equiv y'_{il}\in\vec{y'_i}} y'_{ih}\sigma\mu \neq_{AC} y'_{il}\sigma\mu))$. Formally:

THEOREM 4.1. *If $\mathcal{C}$ has a solution in $T(\mathcal{F})/_{=_{AC}}$ then either*

- *$\mathcal{L}$ has a solution, or*
- *there exists $\lambda \in S^*_{\vec{w},A(\psi)}$, such that*

$$T(\mathcal{F})/_{=_{AC}} \models \exists\vec{z}\colon \bigwedge_{1\leq j\leq n}\bigwedge_{\sigma\in M_{j\lambda}}\Big(\bigvee_{1\leq i\leq|Var(t'_j)|}\Big(\bigvee_{y'_{ih}\not\equiv y'_{il}\in\vec{y'_i}} y'_{ih}\sigma \neq y'_{il}\sigma\Big)\Big)$$

where $\vec{z}$ are the variables in $Im(\lambda)$.

PROOF. Assume that ν is a solution of $\mathcal{C}$. By construction, there exists $\lambda \in S^*_{\vec{w},A(\psi)}$ such that $\nu =_{AC} \lambda\mu$, where μ is a ground assignment. Then, either ν is a solution of $\mathcal{L}$ and we are finished, or ν is not a solution of $\mathcal{L}$ and then by (1), there exist some j's ($1 \leq j \leq n$) such that $T(\mathcal{F})/_{=_{AC}} \models \exists\vec{y'} : t\lambda\mu = t'_j$, and since $\lambda\mu$ is a solution of $\mathcal{C}$, for all such j, $\bigwedge_{\sigma\in M_{j\lambda}}(\bigvee_{1\leq i\leq|Var(t'_j)|}(\bigvee_{y'_{ih}\not\equiv y'_{il}\in\vec{y'_i}} y'_{ih}\sigma\mu \neq_{AC} y'_{il}\sigma\mu))$. This proves that for all $1 \leq j \leq n$, $t\lambda\mu =_{AC} t'_j\sigma\mu$ implies $\bigwedge_{\sigma\in M_{j\lambda}}(\bigvee_{1\leq i\leq|Var(t'_j)|}(\bigvee_{y'_{ih}\not\equiv y'_{il}\in\vec{y'_i}} y'_{ih}\sigma\mu \neq_{AC} y'_{il}\sigma\mu))$, and this proves the thesis. □

But the converse is not true: if there exist $\lambda \in S^*_{\vec{w},A(\psi)}$ and a ground assignment μ such that for $1 \leq j \leq n$, $\bigwedge_{\sigma\in M_{j\lambda}}(\bigvee_{1\leq i\leq|Var(t'_j)|}(\bigvee_{y'_{ih}\not\equiv y'_{il}\in\vec{y'_i}} y'_{ih}\sigma\mu \neq_{AC} y'_{il}\sigma\mu))$, it may be the case that $\lambda\mu$ is not a solution of $\mathcal{C}$. For instance, consider the problem $\mathcal{C} \equiv \forall\vec{y} : x_1 + x_2 \neq y + y$, in the signature $\mathcal{F} = \{a, +\}$ where a is a constant and $+$ is an AC function symbol. In this case λ can only have a variable as fourth argument of $+$. Let us denote by $n.a$ the term $a + \cdots + a$ where a appears n times. Let $\lambda = \{x_1 \mapsto 3.a + x_3, x_2 \mapsto 3.a + x_4\}$ and μ any ground assignment such that $x_3\mu = a$, and $x_4\mu = 9.a$. An analysis of the AC-matchers of the problem $6.a + x_3 + x_4 = y'_1 + y'_2$ shows that μ satisfies the condition of Theorem 4.1, however μ is not a solution of $\mathcal{C}$.

The problem is that there are, in general, other ground assignments μ' such that $t\lambda\mu =_{AC} t\lambda\mu'$ and some of these μ' may not satisfy the required condition (in the previous example, if we take $\mu' = \{x_3 \mapsto 5.a, x_4 \mapsto 5.a\}$ then $(x_1 + x_2)\lambda\mu' =_{AC} (x_1 + x_2)\lambda\mu$, and μ' does not satisfy the condition).

To avoid this problem we will strengthen the hypothesis of Theorem 4.1.

A non-linear AC-complement problem $\mathcal{C} \equiv t \neq t_1 \wedge \ldots \wedge t \neq t_n$ will be called *safe* if no variable occurs *as* an argument of an AC-operator in $t_1, \ldots, t_n$ (i.e. variables can appear *inside* the arguments of AC-operators, but not directly below an AC-operator). We will prove in the following that the converse of Theorem 4.1 holds for safe non-linear

problems. This will provide a necessary and sufficient condition for the satisfiability of this class of non-linear AC-complement problems.

LEMMA 4.1. *Let $\mathcal{C}$ and $\mathcal{L}$ be as above, and assume $\mathcal{C}$ is safe and $\mathcal{L}$ does not have solutions. If there exist $\lambda \in S^*_{\vec{w},A(\psi)}$ and a ground assignment μ such that for $1 \leq j \leq n$, $\bigwedge_{\sigma \in M_{j\lambda}}(\bigvee_{1 \leq i \leq |Var(t'_j)|}(\bigvee_{y'_{ih} \not\equiv y'_{il} \in \vec{y'_i}} y'_{ih}\sigma\mu \neq_{AC} y'_{il}\sigma\mu))$, then for all ground assignment μ' such that $t\lambda\mu =_{AC} t\lambda\mu'$, and for $1 \leq j \leq n$, $\bigwedge_{\sigma \in M_{j\lambda}}(\bigvee_{1 \leq i \leq |Var(t'_j)|}(\bigvee_{y'_{ih} \not\equiv y'_{il} \in \vec{y'_i}} y'_{ih}\sigma\mu' \neq_{AC} y'_{il}\sigma\mu'))$.*

PROOF. By hypothesis, there exist t_j and σ such that $t\lambda =_{AC} t'_j\sigma$ (because $\mathcal{L}$ does not have solutions). Then,

$$(2) \qquad t'_j\sigma\mu =_{AC} t\lambda\mu =_{AC} t\lambda\mu' =_{AC} t'_j\sigma\mu'.$$

Now, by Property 3.3, $\mathrm{flat}(t'_j\sigma) =_P \mathrm{flat}(\mathrm{flat}(t'_j)\,\mathrm{flat}(\sigma))$, but since $\mathcal{C}$ is safe, $\mathrm{flat}(t'_j\sigma) =_P \mathrm{flat}(t'_j)\,\mathrm{flat}(\sigma)$. Then, by (2), $\mathrm{flat}(t'_j)\,\mathrm{flat}(\sigma\mu') =_P \mathrm{flat}(t'_j)\,\mathrm{flat}(\sigma\mu)$.

Hence, since t'_j does not have variables as arguments of AC symbols, the latter equality implies that for all y', $\mathrm{flat}(y'\sigma\mu') =_P \mathrm{flat}(y'\sigma\mu)$ and this implies that $\sigma\mu =_{AC} \sigma\mu'$ on the variables $\vec{y'}$. □

THEOREM 4.2. *Let $\mathcal{C}$ be a safe AC-complement problem. $\mathcal{C}$ has a solution in $T(\mathcal{F})/_{=_{AC}}$ if and only if*

- *$\mathcal{L}$ has a solution, or*
- *there exists $\lambda \in S^*_{\vec{w},A(\psi)}$, such that*

$$T(\mathcal{F})/_{=_{AC}} \models \exists \vec{z} : \bigwedge_{1 \leq j \leq n} \bigwedge_{\sigma \in M_{j\lambda}} \Big(\bigvee_{1 \leq i \leq |Var(t'_j)|} \Big(\bigvee_{y'_{ih} \not\equiv y'_{il} \in \vec{y'_i}} y'_{ih}\sigma \neq y'_{il}\sigma \Big) \Big)$$

where $\vec{z}$ are the variables in $Im(\lambda)$.

PROOF. Consequence of Theorem 4.1 and Lemma 4.1. □

COROLLARY 4.1. *Satisfiability of non-linear safe AC-complement problems is decidable.*

Other classes of non-linear AC-complement problems whose satisfiability is decidable are described by Kounalis *et al.* (1991) and Lugiez and Moysset (1993): the class of problems where the signature contains only one AC-operator and constants, and the class of problems where all the occurrences of a non-linear variable are under the same node, respectively.

Unfortunately, the three classes of non-linear problems for which satisfiability is known to be decidable do not cover all possible non-linear AC-complement problems. The decidability of AC-complement problems in general still remains an open problem.

However, the case of safe complement problems is equivalent to the general case if the domain is defined by constructors: with this extra-hypothesis, if a variable x occurs as an argument of an AC symbol in a t_i, we can replace $t \neq t_i$ with $t \neq t_i\{x \mapsto c_1(\vec{z}_1)\} \wedge \ldots \wedge t \neq t_i\{x \mapsto c_n(\vec{z}_n)\}$ in $\mathcal{C}$ (assuming that $c_1, \ldots, c_n$ are the constructors and $\vec{z}_1, \ldots, \vec{z}_n$ are fresh variables that will be universally quantified). In this way we obtain an equivalent safe problem.

5. Generalization

The most interesting generalization of the previous results, the non-linear case, is still open. But our decision algorithms for validity and negation elimination in linear AC complement problems generalize easily to a wider class of formulae in Σ_2: the $\forall$-*linear* formulae. These are formulae

$$\exists \vec{w}, \forall \vec{y} : t_1 \neq t'_1 \wedge \ldots \wedge t_n \neq t'_n$$

where $t'_1, \ldots, t'_n$ do not contain existentially quantified variables and each universally quantified variable occurs only once.

R_{EAC} is correct and terminating for $\forall$-linear formulae. Moreover, the irreducible forms are again $\forall$-simple formulae. This means that we can use for $\forall$-linear formulae in $T(\mathcal{F})/_{=_{AC}}$ the same decision tests presented for AC complement problems.

THEOREM 5.1. *Validity and negation elimination are decidable for* $\forall$*-linear formulae in* $T(\mathcal{F})/_{=_{AC}}$.

The decomposition of the problem into a simplification phase (using transformation rules) followed by a decision test over irreducible forms, allows us to generalize easily the results to a wider class of formulae that have the same irreducible forms. This is another advantage of the formalism we have chosen.

Acknowledgements

I would like to thank Hubert Comon for many fruitful discussions about complement problems and for reading the first drafts of this article, and Alexandre Boudet for answering my questions about AC-unification algorithms. Thanks also to Ian Mackie and the referees for their comments and suggestions.

References

Buntine, W., Bürckert., H.-J. (1994). On solving equations and disequations. *J. ACM*, **41**(4):591–629. Also available as Seki Report SR89-03.

Comon, H. (1988). Unification et disunification: Théorie et applications. Thèse de Doctorat, Institut National Polytechnique de Grenoble, France.

Comon, H. (1991). Complete axiomatizations of some quotient term algebras. In *Proc.* 18*th Int. Coll. on Automata, Languages and Programming, Madrid,* LNCS 510.

Comon, H., Fernández, M. (1992). Negation elimination in equational formulae. In *Proc.* 17*th Mathematical Foundations of Computer Science, Praha*, LNCS 629.

Comon, H., Lescanne, P. (1989). Equational problems and disunification. *J. Symbolic Computation*, **7**:371–425.

Dershowitz, N., Jouannaud, J.-P. (1990). Rewrite Systems. In van Leeuwen, J. (ed), *Handbook of Theoretical Computer Science*, volume B, chapter 6, pp. 245–320. North-Holland.

Fages, F. (1987). Associative-commutative unification. *J. Symbolic Computation*, **3**:257–275.

Fernández, M. (1992). Narrowing based procedures for equational disunification. *Applicable Algebra in Engineering Communication and Computing*, **3**:1–26.

Fernández, M. (1993). AC-Complement Problems: Satisfiability and Negation Elimination. In *Proc.* 5*th Int. Conf. Rewriting Techniques and Applications, Montréal*, LNCS 690.

Hullot, J.-M. (1979). Associative commutative pattern matching. In *Proc.* 6*th IJCAI (Vol. I), Tokyo*, pp. 406–412.

Jouannaud, J.-P., Kounalis, E. (1989). Automatic proofs by induction in theories without constructors. *Information and Computation*, **82**(1).

Kounalis, E. (1990). Learning from examples and counterexamples with equational background knowledge. In *Proc.* 2*nd IEEE Conference on Tools for Artificial Intelligence.*

Kounalis, E., Lugiez, D., Pottier, L. (1991). A Solution of the Complement Problem in Associative and Commutative Theories. In *Proc.* 16*th Mathematical Foundations of Computer Science, Warsaw*, LNCS 520.

Lassez, J.-L., Maher, M., Marriott, K.G. (1991). Elimination of negation in term algebras. In *Proc.* 16*th Mathematical Foundations of Computer Science, Warsaw*, LNCS 520.

Lassez, J.-L., Marriott, K.G. (1987). Explicit representation of terms defined by counter examples. *J. Automated Reasoning*, **3**(3):1–17.

Lugiez, D., Moysset, J.-L. (1993). Complement problems and tree automata in AC-like theories. In *Proc. Symp. on Theoretical Aspects of Computer Science, Würzburg.* Also available as Technical Report CRIN 92-R-175.

Mal'cev, A.I. (1971). Axiomatizable classes of locally free algebras of various types. In *The Metamathematics of Algebraic Systems. Collected Papers.* 1936–1967, pp. 262–289. North-Holland.

Stickel, M. (1981). A unification algorithm for associative-commutative functions. *J. ACM*, **28**(3):423–434.

Thiel, J.-J. (1984). Stop loosing sleep over incomplete specifications. In *Proc.* 11*th ACM Symp. Principles of Programming Languages, Salt Lake City.* ACM Press.

A. Appendix

We recall here the AC-unification described by Fages (1987). Let $P \equiv s_1 = t_1 \wedge \ldots \wedge s_n = t_n$ be a unification problem. The function call $\texttt{unicompound}([s_1, \ldots, s_n], [t_1, \ldots, t_n], \epsilon, Var(P))$ where ϵ is the identity substitution returns a $CSU_{AC}(P)$.

In order to describe the algorithm we use Fages's notation: $Dom(\sigma) \cup Var(Im(\sigma))$ is denoted by $V(\sigma)$, terms are denoted by capital letters $M, N, \ldots$ and for two terms M and N, $Var(M) \cup Var(N)$ is denoted by $V(M, N)$. For a sequence $\vec{M} = [M_1, \ldots, M_n]$ of terms, the set of variables $\bigcup_{1 \leq i \leq n} Var(M_i)$ is denoted by $V(\vec{M})$. $\mathcal{F}'$ denotes the subset of AC-function symbols in $\mathcal{F}$. The sequence of top-level arguments of M is denoted by $\vec{M}$, and if the root symbol of M is in $\mathcal{F}'$ then $\vec{M}_{AC}$ is the sequence of AC arguments of M, that is the sequence of arguments obtained after elimination of parenthesis on the root symbol.

$\texttt{unicompound}([M_1, \ldots, M_n], [N_1, \ldots, N_n], \sigma_0, W_0) =$

$$\{\sigma_n|_{Var(\vec{M},\vec{N})} \mid \exists \rho_1, \ldots, \rho_n, \sigma_1, \ldots, \sigma_n \in \Sigma, \exists W_1, \ldots, W_n \subset \mathcal{X}, \text{ for } 1 \leq i \leq n,\ \rho_i \in \texttt{uniAC}(M_i\sigma_{i-1}, N_i\sigma_{i-1}, W_i),\ \sigma_i = \sigma_{i-1}\rho_i,\ W_i = (W_0 \cup \bigcup_{1 \leq j \leq i-1} V(\rho_j)) - V(M_i\sigma_{i-1}, N_i\sigma_{i-1})\}$$

$\texttt{uniAC}(M, N, W) =$

$\texttt{if } M \in \mathcal{X} \texttt{ then if } M = N \texttt{ then return } \{\epsilon\}$
$\texttt{else if } M \textit{ occurs in } N \texttt{ then return } \emptyset$
$\texttt{else return } \{\{M \mapsto N\}\}$
$\texttt{else if } N \in \mathcal{X} \texttt{ then if } N \textit{ occurs in } M \texttt{ then return } \emptyset$
$\texttt{else return } \{\{N \mapsto M\}\}$
$\texttt{else let } f(\vec{M}) \equiv M \texttt{ and } g(\vec{N}) \equiv N,$
$\texttt{if } f \not\equiv g \texttt{ then return } \emptyset$
$\texttt{else if } f \notin \mathcal{F}' \texttt{ then return unicompound}(\vec{M}, \vec{N}, \epsilon, W \cup V(M, N)) \quad (*)$
$\texttt{else return } \bigcup_{(A,B) \in \texttt{dio}(M,N,W)} \texttt{unicompound}(\vec{A'}, \vec{B'}, \sigma, W \cup V(M, N) \cup V(\sigma)) \quad (**)$
$\texttt{where } (\sigma, \vec{A'}, \vec{B'}) = \texttt{elimvar}(\vec{A}, \vec{B})$

$\texttt{elimvar}([A_1, \ldots, A_p], [B_1, \ldots, B_p]) =$

$\texttt{let } \sigma = \{A_i \mapsto B_i \mid 1 \leq i \leq p \text{ and } A_i \in \mathcal{X}\}$,
$\texttt{let } \vec{A'} = [A_i \mid 1 \leq i \leq p \text{ and } A_i \notin \mathcal{X}]$
$\texttt{let } \vec{B'} = [B_i \mid 1 \leq i \leq p \text{ and } A_i \notin \mathcal{X}]$
$\texttt{return}(\sigma, \vec{A'}, \vec{B'})$

$\texttt{dio}(M, N, W) =$

$\texttt{let } (\vec{A}, \vec{c}, \vec{c'})$ be the triple composed of the sequence $\vec{A}$ of distinct arguments in $\vec{M}_{AC}$ and $\vec{N}_{AC}$ after elimination of common arguments pair by pair, and of sequences $\vec{c}$ and $\vec{c'}$ of their coefficients of multiplicity ($\vec{c}$, $\vec{c'}$ determine the associated Diophantine equation to solve in N_+).
$\texttt{return } \{\texttt{trans}(D_{pq}, \vec{A}, W) \mid D_{pq} \in \texttt{partit}(\texttt{dioph}(\vec{c}, \vec{c'}), \vec{A})\}$,

where $\texttt{dioph}(\vec{c}, \vec{c'})$ denotes the matrix of solutions over N of the Diophantine equation with coefficients $\vec{c}, \vec{c'}$ and $\texttt{partit}(\texttt{dioph}(\vec{c}, \vec{c'}), \vec{A})$ denotes the set of all submatrices such that each column has at least one non-zero coefficient and each column corresponding to one non-variable argument has exactly one coefficient equal to 1 and all the others are 0. Let M_X, N_X be the terms M and N where non-variable arguments u_i, v_i of M_{AC}, N_{AC} have been replaced by new different variables x_i, y_i. The function `trans` calculates for each element of D_{pq} a solution μ of the equation $M_X = \mathbf{N}_X$ and then makes all possible replacements of variables by terms in the set of equations $x_i\mu = u_i$ and $y_i\mu = v_i$ obtaining a set E of equations. It returns a pair (A, B) of lists, where the elements of A are the left members in E and the variable arguments of M_{AC}, N_{AC}, and the elements of B are the right members of E and the image in μ of each variable argument of M_{AC}, N_{AC}.