Equational Problems and Disunification

Hubert COMON and Pierre LESCANNE[*]

Laboratoire d'Informatique Fondamentale et d'Intelligence Artificielle
46 avenue Félix Viallet 38031 Grenoble Cedex, France

[*] *Centre de Recherche en Informatique de Nancy. CNRS and INRIA-Lorraine*
Campus Scientifique, BP 239 54506 Vandœuvre-les-Nancy, France

Roughly speaking, an equational problem is a first order formula whose only predicate symbol is =. We propose some rules for the transformation of equational problems and study their correctness in various models. Then, we give completeness results with respect to some "simple" problems called solved forms. Such completeness results still hold when adding some control which moreover ensures termination. The termination proofs are given for a "weak" control and thus hold for the (large) class of algorithms obtained by restricting the scope of the rules. Finally, it must be noted that a by-product of our method is a decision procedure for the validity in the Herbrand Universe of any first order formula with the only predicate symbol =.

1 Introduction

1.1 Contents of the Paper

It is well known that (first order) unification can be expressed as a transformation of equations systems (e.g. Kirchner 1985, Kirchner 1986, Colmerauer 1984, Lassez, Maher & Marriott 1986, Gallier & Snyder 1987). This presentation indeed clearly separates inference and control. Then, depending on the problems we are faced to, it is possible to choose the most efficient control.

A first extension of equations systems has been investigated for the semantic definition of PROLOG II (Colmerauer 1984). Indeed, A. Colmerauer introduced "disequations" which are expressions $t \neq t'$. He shows in (Colmerauer 1984) that some transformations may be performed on equations and disequations systems in such a way that "irreducible" systems (called *solved forms* in this paper) have at least one solution in the algebra of rational trees. Such an approach is also developed in (Lassez, Maher & Marriott 1986) where the fundamental mechanisms are demonstrated.

On the other hand, some systems of disequations, called *complement problems*, are used (often in an implicit way) in many situations. For instance in learning from counter-examples (Lassez & Marriott 1987), in pattern matching for functional languages (Laville 1987, Schnoebelen 1988). Finally, such problems are used for solving a classical problem of term rewriting system theory, namely the sufficient completeness (Guttag & Horning 1978) (or, more generally, the inductive reducibility property (Jouannaud & Kounalis 1986)), a natural statement of which is by a set of disequations (Lazrek, Lescanne & Thiel 1986, Comon 1986, Thiel 1984, Kucherov 1988). Complement problems are systems of disequations, but they have the particularity that some of the variables are universally quantified.

The first aim of this paper is to unify all these previous works into a same framework: *equational problems*. Therefore, equational problems will contain equations, disequations, conjunctions and disjunctions, as well as quantified variables[1]. A similar approach was already used in Kirchner & Lescanne (1987). Also, such systems with quantified variables are studied in Lassez, Maher & Marriott (1986) and Maher (1988).

Finally, unification in *equational theories* has been studied in many papers (see Siekmann 1984), and disequations systems in equational theories have been recently studied by Bürckert (1988b). That is the reason why our definitions and rules (section 2 and 3) consider solutions in equational theories or in the algebra of rational trees.

The second aim of this paper is to provide transformation rules which preserve the set of solutions of an equational problem. Therefore, we propose in section 3 a set of rules and study their correctness in the general framework of equational theories. This set of rules (which completes the set given in Kirchner & Lescanne (1987)) is the basis of all further transformations. It will be used together with different controls, depending on the solved forms we are interested in. In this section, we don't care about termination issues since this will be done separately in section 5.

In section 4, the notion of *solved form* is introduced. Such a concept was already used by C. Kirchner in the framework of unification problems (Kirchner 1985). In the unification case, for example, solved forms may either define a most general unifier (replacements have been performed) or insure the existence of a solution without giving it explicitly. This distinction is very important, for example in logic programming, since effective full replacements may be very expensive w.r.t. both space and time while there is generally no need to provide the explicit solutions until the stack of goals is empty. Completeness results of the set of rules given in section 3 are then provided, with respect to various solved forms (the case of equational theories is no longer considered).

[1] Of course, unification problems are equational problems (no quantified variables, no disequations and no disjunctions) but, similarly, *word problems* are equational problems (all the variables are universally quantified).

In section 5, we are interested in termination issues. Thus, we study the rules of section 3, with some additional control and prove termination results, for solutions in the "Herbrand Universe" $T(F)$. This is the main (new) result of the paper. The control we give is "as free as possible": it is possible to obtain, for example, Robinson's unification algorithm (Robinson 1965), Martelli and Montanari's algorithm (Martelli & Montanari 1982), Colmerauer's algorithm (Colmerauer 1984) by refining our control. In this sense, our termination proof is "generic". Indeed, any specialization of the control cannot fail to terminate. A given class of solved forms, called *definitions with constraints* insure the existence of a solution in the Herbrand Universe. Similar results are given in the case of infinite trees. As a corollary, it can be derived that the validity in the Herbrand Universe, of a first order formula containing only the "=" predicate symbol is decidable. This result has been proved (independently) by M.J. Maher (Maher 1988).

1.2 Comparison with Related Works

Problems containing equalities and disequalities with parameters have already been studied. Kirchner and Lescanne (1987) were already cited. They introduced equational problems but no completeness result was given. Also, a particular case of equational problem was studied in Lassez, Maher & Marriott (1986). In the latter paper, it is shown that some equational problems cannot be reduced to unification problems. This is given as a consequence of the results in Lassez & Marriott (1987). Actually, two very recent works tackle similar problems (systems of equations and disequations), but use a quite different approach. H.J. Bürckert (Bürckert 1988b) addresses the problem of solving equations and disequations in equational theories. Roughly speaking, he shows that in any theory in which it is possible to represent the solutions of a set of equations by a finite set of substitutions, it is also possible to represent the set of solutions of a system containing equations and disequations by a finite set of "substitutions with exceptions". This approach is very different from our, since, although the more general framework of equational theories is considered, systems of equations and disequations are not as general as equational problems. Moreover, the method is quite different. Indeed, turning a unification algorithm into a "disunification" algorithm leads to other solved forms. These solved forms are used by H.J. Bürckert for improving AC-unification, but probably cannot be used for solving the above mentioned sufficient completeness problem.

Finally, MJ. Maher in his paper (Maher 1988) studies first order formulae containing the only predicate symbol =. Essentially, he solves the same problem we study in this paper. His motivations (and therefore his point of view) are different. Indeed, we try in this paper to give a generalization of well-known unification algorithms. Thus, we don't use unification algorithms but rather generalize them. Also, as discussed above, we try to minimize the control in order to cover most of the known (and future) algorithms. On the other hand, many similarities can be found. The reader is referred to Maher (1988) for more details.

As outlined above, section 2 will be devoted to the framework, section 3 to the transformation rules, section 4 to solved forms and section 5 contains the main results of the paper: the termination of the transformations.

2 Equational Problems: Syntax and Semantics

In this section, we describe what an equational problem is. Roughly speaking, this is a problem that contains equations and disequations between two terms. Let us recall first some basic notations.

2.1 Basic Notations

$S = \{\underline{s}_1, \ldots\}$ will denote a finite set of symbols called *sorts*. $F = \{f, g \ldots\}$ is a set[2] of functions symbols together with an arity function τ which maps F into S^+, the non empty words constructed on the vocabulary S. We write $f : \underline{s}_1 \times \ldots \times \underline{s}_n \to \underline{s}$ instead of $\tau(f) = \underline{s}_1 \ldots \underline{s}_n \underline{s}$.

An (S,F)*-algebra* (or simply an F-algebra) $\mathcal{A}$ is a familly of sets $\mathcal{A}_{\underline{s}}$, for each $\underline{s} \in S$ and a familly of mappings $f_{\mathcal{A}}$ for each $f \in F$ such that, if $f : \underline{s}_1 \times \ldots \underline{s}_n \to \underline{s}$, then $f_{\mathcal{A}}$ is a mapping from $\mathcal{A}_{\underline{s}_1} \times \ldots \times \mathcal{A}_{\underline{s}_n}$ into $\mathcal{A}_{\underline{s}}$. The set $\mathcal{A}_{\underline{s}}$ is called the *carrier* of $\underline{s}$ in $\mathcal{A}$.

Given a set of sorted variables X, $T(F,X)$ is the free F-algebra over the sets of generators X. (See e.g. Huet & Oppen 1980 for more details). The elements of $T(F,X)$ are called *terms*. $T(F,\emptyset)$ is also denoted $T(F)$. We assume in the following that $T(F)$ contains at least one term of each sort. Its elements are called *ground terms*.

The sort of a term t is noted $sort(t)$. When the carrier of $sort(t)$ in $T(F)$ is infinite (resp. finite) we say that t is *sort-unrestricted* (resp. *sort-restricted*).

Classically, terms can also be viewed as finite trees. We don't recall the complete definition of a tree. Let us just note that a (finite, labeled) tree t is a (finite) prefix-closed set of sequences of integers called positions (or occurrences) and denoted by $Pos(t)$ together with a mapping from this set of positions into $F \cup X$ (the set of labels). Some more conditions, related to the arity of the symbols in $F \cup X$ are required in order to get a "well formed" tree. In particular, a node labeled by a variable cannot have any sons. The symbol at position p in a tree t is classically denoted by $t(p)$ whereas the subterm at position p is denoted by t/p. If $p \in Pos(t)$ and u is a term $t[p \leftarrow u]$ is the tree obtained by replacing t/p by u in t. The *size* of a position is its length as a sequence. The empty sequence is denoted by ε.

In order to avoid confusions, we use the symbol $\equiv$ to denote the *syntactic* equality between terms. Finally, given any expression e, $Var(e)$ will denote the set of variables occurring in e.

Let $\mathcal{A}$ be an F-algebra. An $\mathcal{A}$-substitution σ is a F-morphism from $T(F,X_0)$ into $\mathcal{A}$, where X_0 is a finite subset of X called the *domain* of σ and denoted by $Dom(\sigma)$[3].

[2] F will be assumed to be finite throughout this paper. The case where F is infinite is studied in Lassez, Maher & Marriott (1986) and Maher (1988), it seems to be simpler.

[3] This is not the standard definition (as e.g. in Huet & Oppen 1980), but this allows substitutions in any F-algebra $\mathcal{A}$, including the cases where $\mathcal{A}$ does not contain X

The value of σ on t is written $t\sigma$. An $\mathcal{A}$-substitution is uniquely defined by its domain and the values it takes on its domain. When $\mathcal{A} = T(F,X)$, a $T(F,X)$-substitution σ is canonically extended to $T(F,X)$ by adding the relations $x\sigma \equiv x$ for every $x \in X - Dom(\sigma)$. A substitution σ is *away* from $X_0 \subseteq X$ if both $Dom(\sigma)$ and $\sigma(Dom(\sigma))$ do not share any variable with X_0. When $\mathcal{A} = T(F)$, a $T(F)$-substitution is called a *ground substitution* (or, more simply, a substitution, if there is no ambiguity).

In order to meaningfully compose $\mathcal{A}$-substitutions when $\mathcal{A}$ does not contain X, we have to give some more definitions. Let σ and θ be two $\mathcal{A}$-substitutions. Then $\sigma\theta$, which is *not* the composition of substitutions σ and θ, denotes the $\mathcal{A}$-substitution defined by:

- $Dom(\sigma\theta) = Dom(\sigma) \cup Dom(\theta)$
- If $x \in Dom(\sigma)$, then $x(\sigma\theta) =_{\mathcal{A}} x\sigma$
- If $x \in Dom(\theta)$ and $x \notin Dom(\sigma)$, then $x(\sigma\theta) =_{\mathcal{A}} x\theta$

It must be noted that this operation is associative, that is $\sigma(\theta\rho) = (\sigma\theta)\rho$ (which will be also denoted, as usually, by $\sigma\theta\rho$).

When σ and θ are two $T(F,X)$-substitutions and if σ is away from $Dom(\theta)$, then $\sigma\theta = \theta \circ \sigma$ (the usual composition of applications). In practice, we will always make such an assumption. Thus, there will be no confusion and we may use the notations $(x\sigma)\theta$ or even $x\sigma\theta$ instead of $x(\sigma\theta)$[4].

The $\mathcal{A}$-substitution σ whose domain is $Dom(\sigma) = \{x_1,\ldots,x_n\}$ and such that $x_i\sigma =_{\mathcal{A}} t_i$ will be denoted by $(x_1 \leftarrow t_1,\ldots,x_n \leftarrow t_n)$. Substitutions will either be (explicitly) denoted in this way or denoted by Greek letters $\sigma,\theta,\rho,\ldots$.

We shall also make use of *rational trees* (see e.g. Huet 1976). The definition of an infinite tree is the same as for a finite one except that the set of positions may be infinite. Then, a rational tree is an infinite tree whose set of subtrees is finite. (Note that finite trees are rational too). The algebra of rational trees over F will be denoted by $RT(F)$. $\equiv$ will also denote syntactic equality between rational trees. We shall use the well known characteristic property of rational trees (see e.g. Courcelle 1981).

Theorem 1 (Huet 1976)
Given a system $x_1 = t_1 \wedge \ldots \wedge x_n = t_n$, where $x_1,\ldots,x_n \in X$ are distinct variables and $t_1,\ldots,t_n \in T(F,\{x_1,\ldots,x_n\})$ are not variables, there exists a unique n-uple of rational trees $r_1,\ldots,r_n$ such that the $RT(F)$-substitution $\sigma = (x_1 \leftarrow r_1,\ldots,x_n \leftarrow r_n)$ satisfies $\forall i, x_i\sigma \equiv t_i\sigma$.

2.2 Equational Problems

Definition 1
An equation *is an expression of the form $s = t$ where s and t are terms of a same sort, or the symbol $\top$. A* disequation *is an expression of the form $s \neq t$, or the symbol $\perp$.*

[4] Note that $x\sigma\theta$ is then defined even when σ and θ are $\mathcal{A}$-substitutions and $\mathcal{A}$ does not contain X.

A *trivial equation* is either $\top$ or an equation of the form $s = s$. A *trivial disequation* is either $\perp$ or has the form $s \neq s$. s and t are called the *members* of the equation $s = t$ (resp. the members of the disequation $s \neq t$).

Definition 2
A system *is an equation, a disequation or an expression of the form $P_1 \wedge \ldots \wedge P_n$ where $P_1, \ldots, P_n$ are systems or an expression $P_1 \vee \ldots \vee P_n$ where $P_1, \ldots, P_n$ are systems. If the system is reduced to $\perp$, it is said* empty*, if it is reduced to $\top$, it is said* full.

Actually, we are not working on the strict syntactical structure of definitions 1 and 2. Indeed, $\vee$, $\wedge$ are associative, commutative and idempotent, $\top$ is the identity of $\wedge$ and an absorbing element of $\vee$, $\perp$ is is the identity of $\vee$ and an absorbing element of $\wedge$, $\vee$ and $\wedge$ are distributive one with respect to the other and $=$ and $\neq$ are commutative, ... We are working modulo these properties that we often use in what follows without making any mention. However, in section 5 we shall use conjunctive normal forms, which are representatives of these classes such that the only remaining relations between them are the associative and commutative axioms.

Definition 3
An equational problem *is an expression of the form*

$$\exists w_1, \ldots, w_m, \forall y_1, ..., y_n : P$$

where P is a system and $w_1, \ldots, w_m, y_1, \ldots, y_n$ are distinct variables.

An equational problem is given together with a finite set I which contains the **free variables** of the problem.

Thus, the variables occurring in an equational problem may be divided into three (disjoint) sets:

- In the previous definition the variables $y_1, \ldots, y_n$ are called the *parameters* of the problem.
- $w_1, \ldots w_m$ are called the *auxiliary unknowns* of the problem.
- The variables of I are called the *principal unknowns* of the problem They are denoted by $x_1, \ldots, x_k$ and intuitively a solution assigns values to them.

The parameters (resp. the auxiliary unknowns) are grouped in a set, which means we make no difference between $\forall y, y'$ and $\forall y', y$, although sometimes we use $\forall \vec{y}$ instead of $\forall y_1, ..., y_n$.

The parameters range over a domain of terms, which means that the set of equalities and disequalities will be satisfied by a solution whatever values are taken by the parameters. Given this viewpoint, one can see that a problem without parameter has no constraint on the solution. In what follows, we are going to adopt the following conventions, $x, x', x_i, \ldots$ are principal unknowns, $y, y', y_i, \ldots$ are parameters and $z, z', z_i, \ldots$ are any variables.

Let us give three examples. The first one is a natural example arising in sufficient completeness, the second shows a problem in an equational theory and the last one is built in order to exhibit in the following all the possible transformations. In these three examples, there are no auxiliary unknowns since such variables only arise naturally "during" the transformations.

Example 1
There are two sorts: $\underline{bool}$ (booleans) and $\underline{nat}$ (positive integers), F_1 contains the usual boolean operators, the constructors $0, s$ for the sort $\underline{nat}$ and the equality eq which takes two positive integers and returns a boolean. The following problem

$$\forall y : eq(0, s(y)) \neq eq(x_1, x_2) \wedge eq(s(y), 0) \neq eq(x_1, x_2) \wedge eq(y, y) \neq eq(x_1, x_2).$$

has no solution if and only if the axioms $eq(0, s(y)) == false$; $eq(s(y), 0) == false$; $eq(y, y) == true$ completely define[5] eq.

Example 2
Assume that there is only one sort and that F_2 contains a constant 1 and an associative commutative operator $+$ (which is used in infixed notation). The following problem:

$$\forall y : \; x + y \neq x' + 1$$

is an equational problem with parameter y and (principal) unknowns x, x'.

Example 3
We use this example in the following in order to illustrate the definitions and algorithms given in the paper. We assume that there is only one sort $\underline{s}$ and that

$$F_3 = \{a : \rightarrow \underline{s};\; g : \underline{s} \rightarrow \underline{s};\; f : \underline{s} \times \underline{s} \times \underline{s} \rightarrow \underline{s}\}.$$

Then $\mathcal{P}$ is defined as:

$$\begin{aligned} \mathcal{P} \equiv \forall y_1, y_2, y_3 : \quad & (y_1 = x_1 \vee f(x_1, x_4, x_4) = f(x_1, x_2, g(x_3))) \\ & \wedge f(y_1, y_1, g(y_2)) \neq f(g(y_2), x_1, x_2) \\ & \wedge x_1 \neq f(y_1, y_2, y_3) \end{aligned}$$

In what follows (sections 2 and 3) $\mathcal{A}$ is supposed to be either the algebra of rational trees $RT(F)$[6] or a quotient algebra of $T(F, X)$ by a congruence $=_E$[7]. In all cases, $\mathcal{A}$ is an

[5] "complete definition" and "convertibility" will not be defined here, since this is out of the scope of this paper. The reader is referred to e.g. Comon (1986)

[6] Actually, even $RT(F)$ can be viewed as a quotient algebra of $T(F, RT(F))$. Therefore, $RT(F)$ can be viewed as a quotient algebra of some $T(F, X')$.

[7] We write $=_E$, having in mind that the congruence may be defined by a finite set of equational axioms E, but the finiteness hypothesis is not necessary, until we use explicitly the axioms of E

F-algebra. In order to avoid confusions, $=_{\mathcal{A}}$ denotes the equality in $\mathcal{A}$. In many examples $=_E$ will be either the syntactic equality, denoted $\equiv$, which is the equality based on an empty set of equational axioms or the congruence generated by the equational axioms of associativity and commutativity and denoted $=_{AC}$. In sections 4 and 5, $\mathcal{A}$ is assumed to be $T(F)$.

We are now defining what we mean by a solution of a problem $\mathcal{P}$. First, we have to say when a substitution σ *validates a system P.*

Definition 4
*An $\mathcal{A}$-substitution σ $\mathcal{A}$-*validates *a system P if*

- *P is an equation $t = u$ and $t\sigma =_{\mathcal{A}} u\sigma$*

or • *P is a disequation $t \neq u$ and $t\sigma \neq_{\mathcal{A}} u\sigma$*

or • $P \equiv \top$

or • *P is of the form $P_1 \wedge \ldots \wedge P_n$ and σ validates each P_i*

or • *P is of the form $P_1 \vee \ldots \vee P_n$ and σ validates one of the P_i*

Now we say what we mean by a solution of an equational problem.

Definition 5
*Let $\mathcal{P} \equiv \exists w_1,\ldots,w_k,\forall y_1,\ldots,y_n : P$ be an equational problem and I be a finite set of principal unknowns which contains the free variables of $\mathcal{P}$. We say that a substitution σ is a $\mathcal{A}$-*solution of the problem *$\mathcal{P}$ if*

1. *σ is an $\mathcal{A}$-substitution away from $\{w_1,\ldots,w_k,y_1,\ldots,y_n\}$ whose domain is I*
2. *there exists an $\mathcal{A}$-substitution ρ away from $I \cup \{y_1,\ldots,y_n\}$ whose domain is $\{w_1,\ldots,w_k\}$ such that, for all $\mathcal{A}$-substitution θ away from $I \cup \{w_1,\ldots,w_k\}$ whose domain is $\{y_1,\ldots,y_n\}$, $\theta\rho\sigma$ $\mathcal{A}$-validates P.*

this corresponds to the intuitive notion: a solution assigns values to free variables of the problem in such a way that there exists an assignment to existentially quantified variables such that the system is validated whatever values are taken by the parameters. The "away conditions" on the substitutions in this definition are obviously not necessary when $\mathcal{A}$ does not contain variables of X.

If, in addition, the co-domain of σ is required to be included in a set of terms $\mathcal{B} \subseteq \mathcal{A}$, σ is called a *solution in $\mathcal{B}$*. In the following, $\mathcal{B}$ is always assumed to contain at least one element of each sort. Notice that conditions on σ, ρ and θ make $\sigma\rho\theta = \theta\rho\sigma = \ldots$. When $\mathcal{A}$ can be easily inferred from the context, it will be omitted.

Example 4

Example 1 continued.
$\mathcal{A}$ is $T(F)$ and I is the set of free variables $\{x_1, x_2\}$.
$\sigma \equiv (x_1 \leftarrow 0, x_2 \leftarrow 0)$ is not a solution of the equational problem since, for $\theta \equiv (y \leftarrow 0)$, $\sigma\theta$ does not validate P. The substitution $\sigma_1 \equiv (x_1 \leftarrow s(s(0)),\ x_2 \leftarrow s(0))$ is a solution of the problem since, for every substitution θ on y, each disequality is validated by $\sigma\theta$.

Example 5
Example 2 continued.
$\mathcal{A}$ is the quotient algebra $T(F)/=_{AC}$. I is the set of free variables $\{x, x'\}$.
$(x \leftarrow 1+1+1,\ x' \leftarrow 1)$ is a solution of the problem, but $(x \leftarrow 1,\ x' \leftarrow 1)$ is not a solution since, for $(y \leftarrow 1)$ the two members become equal (modulo AC).

Example 6
Example 3 continued.
$(x_1 \leftarrow a,\ x_2 \leftarrow g(a),\ x_3 \leftarrow a,\ x_4 \leftarrow g(a))$ is a solution of $\mathcal{P}$.

In the following, we don't make any mention of the set I which is always assumed to contain the free variables of the problem at hand. I is indeed not relevant in the results given in this paper.

3 Transformations of Equational Problems

Once equational problems have been introduced and once we have given a precise definition of what is a solution of such problems, a natural question that arises is how to compute these solutions. In this paper, we propose a method based on rules that transform a problem in another one with the same set of solutions, with the intention that the transformed problem be simpler. One may expect that the last problem eventually provides a straightforward expression of the solution.

3.1 A set of Rules

As equational problems form a (quotient) algebra, transformation rules may be viewed as (schemata of) sets of rewrite rules in this algebra. It would be boring to give more details. Simply note that the rules can be applied *at an occurrence* in a problem and that the rewriting is done *modulo the boolean properties*. In practice, we will use boolean normal forms, but it is not yet necessary to precise them.

When an equational problem $\mathcal{P}$ can be transformed into a problem $\mathcal{P}'$ using the rule $\mathcal{R}$, we write $\mathcal{P} \to_{\mathcal{R}} \mathcal{P}'$.

In a first presentation, one is not concerned about termination issues. In other words, the set of rules which is provided may lead to infinite computations in some cases. To prevent such non termination some kind of control is usually required, which may make the rules harder to read and which is sometimes difficult to express. In this section, we only keep the control which is necessary for the soundness and the completeness of the rules. Thus a rule will have three parts, a left-hand side, a right-hand side and a control part. In addition, to avoid complexity, we will use abbreviations expecting that the reader will easily understand them. For instance, z stands for any variable, i.e., parameter or unknown and s,t,u,v, stand for any term. There are two sorts of rules, those that fully preserve the set of solutions, we use the symbol $\mapsto$ for them, and call them *preserving* and those that return an equational problem whose solutions are only a subset of the given problem, we use the symbol $\rightharpoonup$ for them and call them *globally preserving* when instances of the same rule can be combined to preserve the set of solutions. For instance, a rule of the form

$$\exists \vec{w},\ \forall \vec{y} : P \quad \rightharpoonup \quad \exists \vec{w},\ \forall \vec{y} : P \wedge Q$$

is trivially sound, but should have a specific form, for instance this presented in figure 4, to be globally preserving. The rules are divided in three classes. In figure 1, we put rules that are sound and preserving for any $\mathcal{A}$. They are called "non $\mathcal{A}$-sensitive rules". figure 2 contains "$\mathcal{A}$-sensitive" rules that are not sound in all $\mathcal{A}$. figure 4 contains rules that are only globally preserving. Merging rules could be avoided, in general, by using replacement rules. But as in Martelli-Montanari algorithm (Martelli & Montanari 1982) they decrease considerably the complexity of the algorithms. Similarly, trivial equations or disequations that are not reduced to variables are not absolutely necessary, since they could be implied by decompositions.

The rules **U** and **U**' as well as **E** are not standard in the unification community. (U_1), (U_2) allow to eliminate the parameters in disequations whereas (U_3), (U_4) eliminate the parameters in equations. In addition, the rule (U_5) is devoted to the finite-sort case: when a parameter ranges over a finite domain, it is sufficient to replace it by all the possible values it can take. The rule (RT) is only available in rational trees; it is required since, in this case, it would be impossible to eliminate disequations such as $x \neq f(x)$ using the rule (U_2). Finally, the explosion rule is, roughly speaking, a "decomposition by case", where we make an assumption on the top symbol of s. In practice, it will only be used when s is a variable. Such a rule is also given in Maher (1988).

3.2 Soundness

The rules are sound, which means they do not introduce unexpected solutions. The preserving property means that no solution are lost by application of the rule.

In these definitions, a set I of principal unknowns is assumed. Then the free variables of the problems which are considered are supposed to belong to I.

For every problem $\mathcal{P}$ and every F-algebra $\mathcal{A}$, $\mathcal{S}(\mathcal{P},\mathcal{A})$ is the set of $\mathcal{A}$-solutions of $\mathcal{P}$.

Definition 6
Let $\mathcal{A}$ be an F-algebra (either $RT(F)$ or a quotient algebra $T(F,X)/=_E$).
*A rule $\mathcal{R}$ is $\mathcal{A}$-*sound *if,*

$$\mathcal{P} \rightarrow_{\mathcal{R}} \mathcal{P}' \;\; \Rightarrow \;\; \mathcal{S}(\mathcal{P}',\mathcal{A}) \subseteq \mathcal{S}(\mathcal{P},\mathcal{A})$$

*A rule $\mathcal{R}$ is $\mathcal{A}$-*preserving *if, for every problems $\mathcal{P}$ and $\mathcal{P}'$,*

$$\mathcal{P} \rightarrow_{\mathcal{R}} \mathcal{P}' \;\; \Rightarrow \;\; \mathcal{S}(\mathcal{P},\mathcal{A}) \subseteq \mathcal{S}(\mathcal{P}',\mathcal{A})$$

Definition 7
*A rule $\mathcal{R}$ is $\mathcal{A}$-*globally preserving *if given any problem $\mathcal{P}$,*

$$\mathcal{S}(\mathcal{P},\mathcal{A}) \subseteq \bigcup_{\mathcal{P}_i, \mathcal{P} \rightarrow_{\mathcal{R}} \mathcal{P}_i} \mathcal{S}(\mathcal{P}_i,\mathcal{A})$$

There are actually three kinds of results related to the rules and the definitions.

Proposition 1
The rules of fi gure 1 are $\mathcal{A}$-sound and $\mathcal{A}$-preserving for any algebra $\mathcal{A}$.

Elimination of trivial equations and disequations: T

$$\begin{array}{llll} (T_1) & t = t & \mapsto & \top \\ (T_2) & t \neq t & \mapsto & \bot \end{array}$$

Replacement: R

$$\begin{array}{llll} (R_1) & z = t \wedge P & \mapsto & z = t \wedge P(z \leftarrow t) \\ (R_2) & z \neq t \vee P & \mapsto & z \neq t \vee P(z \leftarrow t) \end{array}$$

Elimination of Parameters: EP

$$(EP) \quad \forall \vec{y}, y : P \quad \mapsto \quad \forall \vec{y} : P \quad \text{if } y \notin Var(P)$$

Merging: M

$$\begin{array}{llll} (M_1) & s = t \wedge s = u & \mapsto & s = t \wedge t = u \\ (M_2) & s \neq t \vee s \neq u & \mapsto & s \neq t \vee t \neq u \\ (M_3) & s = t \wedge s \neq u & \mapsto & s = t \wedge t \neq u \\ (M_4) & s = t \vee s \neq u & \mapsto & t = u \vee s \neq u \end{array}$$

Universality of Parameters: U

$$\begin{array}{llcl} (U_1) & \forall \vec{y} : P \wedge y \neq t & \underset{\text{if } y \in \vec{y}}{\mapsto} & \bot \\ (U_2) & \forall \vec{y} : P \wedge (y \neq t \vee R) & \underset{\text{if } y \in \vec{y}}{\mapsto} & \forall \vec{y} : P \wedge R(y \leftarrow t) \end{array}$$

Cleaning : **CR**

$$\begin{array}{llcl} (CR_1) & \exists w : P & \underset{\text{If } w \notin Var(P)}{\mapsto} & P \\ (CR_2) & \exists \vec{w}, w : w = t \wedge P & \underset{\text{If } w \notin Var(P,t)}{\mapsto} & \exists \vec{w} : P \end{array}$$

Figure 1: Preserving and non $\mathcal{A}$-sensitive rules for solving equational problems

Clash: C

$$\begin{array}{llcll} (C_1) & f(t_1,\dots,t_m) = g(u_1,\dots,u_n) & \mapsto & \bot & \text{if } f \neq g \\ (C_2) & f(t_1,\dots,t_m) \neq g(u_1,\dots,u_n) & \mapsto & \top & \text{if } f \neq g \end{array}$$

Decomposition: D

$$\begin{array}{llcl} (D_1) & f(t_1,\dots,t_m) = f(u_1,\dots,u_m) & \mapsto & t_1 = u_1 \wedge \dots \wedge t_m = u_m \\ (D_2) & f(t_1,\dots,t_m) \neq f(u_1,\dots,u_m) & \mapsto & t_1 \neq u_1 \vee \dots \vee t_m \neq u_m \end{array}$$

Occur Check: O

$$\begin{array}{llcll} (O_1) & z = t & \mapsto & \bot & \text{if } z \in Var(t) \text{ and } z \not\equiv t \\ (O_2) & z \neq t & \mapsto & \top & \text{if } z \in Var(t) \text{ and } z \not\equiv t \end{array}$$

Universality of Parameters: U'

$$(U_3) \quad \forall \vec{y} : P \wedge z = t \quad \mapsto \quad \bot$$

If $z \not\equiv t$ and there exists $y \in Var(z = t) \cap \vec{y}$ such that $|T(F)_{sort(y)}| \geq 2$.

$$(U_4) \quad \forall \vec{y},\, P \wedge (z_1 = u_1 \vee \dots \vee z_n = u_n \vee R) \quad \mapsto \quad \forall \vec{y},\, P \wedge R$$

If 1. for each index i, z_i is a variable and $z_i \not\equiv u_i$,
2. for each index i, $z_i = u_i$ contains at least one occurrence of a parameter,
3. for each index i and any parameter $y \in Var(z_i, u_i)$, y is sort-unrestricted,
4. R does not contain any parameter.

$$(U_5) \quad \forall \vec{y} : P \wedge Q \quad \mapsto \quad \forall \vec{y} : P \wedge Q(y \leftarrow t_1) \wedge \dots \wedge Q(y \leftarrow t_n)$$

If y is a parameter, of sort $\underline{s}$, $y \in Var(Q)$ and $\mathcal{A}_{\underline{s}} = \{t_1,\dots,t_n\}$

Cleaning Rules : CR'

$$(CR_3) \quad \exists \vec{w} :\ (d_1 \vee z_1 \neq u_1) \wedge \dots \wedge (d_n \vee z_n \neq u_n) \wedge P \quad \mapsto \quad \exists \vec{w} :\ P$$

If each d_i is a disjunction of equations and disequations, each z_i is a variable, each $z_i \neq u_i$ is a non trivial disequation and there exists a variable $w \in \vec{w} \cap Var(z_1, u_1) \cap \dots \cap Var(z_n, u_n)$ which does not occur in P and which is sort-unrestricted.

Figure 2: Preserving and $\mathcal{A}$-sensitive rules for solving equational problems

Elimination of parameters in the algebra of rational trees : RT

$$\begin{aligned}(RT)\quad \forall\vec{y}:\ P \wedge (y_1 \neq t_1 \vee \ldots \vee y_n \neq t_n \\ \vee\, y_{n+1} = t_{n+1} \vee \ldots \vee y_{n+m} = t_{n+m} \vee d) \quad \mapsto \quad \forall\vec{y}:\ P \wedge d\end{aligned}$$

If

1. d is a disjunction of equations and disequations and d does not contain any parameter,
2. $y_1,\ldots,y_n$ are distinct parameters,
3. $y_{n+1},\ldots y_{n+m}$ are parameters,
4. every (y_i), $n+1 \leq i \leq n+m$, in $RT(F)$ is sort-unrestricted,
5. The sets $\{y_1 \ldots y_n\}$, $\{y_{n+1},\ldots,y_{n+m},t_{n+1},\ldots,t_{n+m}\}$, $\{t_1,\ldots t_m\}$ are disjoint.

Figure 3: Parameter elimination in Rational Trees

Explosion: E

$$(E)\quad \forall\vec{y}:P \quad \rightharpoonup \quad \exists w_1,\ldots w_p, \forall\vec{y}:P \wedge s = f(w_1,\ldots,w_p)$$

if $\{w_1,\ldots,w_p\} \cap (Var(P) \cup \vec{y} \cup I) = \emptyset$, $f \in F$ and s is a member of an equation or a disequation of P and no parameter occurs in s.

Elimination of disjunctions: ED

$$(ED)\quad \forall\vec{y}:P \wedge (P_1 \vee P_2) \quad \rightharpoonup \quad \forall\vec{y}:P \wedge P_1$$

if $Var(P_1) \cap \vec{y} = \emptyset$ or $Var(P_2) \cap \vec{y} = \emptyset$

Figure 4: Globally preserving rules for solving equational problems

The proof of this proposition can be found in appendix A.

Soundness and preservation of the rules in figure 2 depend on which algebra $\mathcal{A}$ is considered.

Proposition 2
The rules of figure 2 (except for the **RT** *rule) are $\mathcal{A}$-sound and $\mathcal{A}$-preserving in any subalgebra $\mathcal{A}$ of $T(F,X)$.*
The rules of figure 2 (except for the **O** *rules) are $RT(F)$-sound and $RT(F)$-preserving.*

Note that equational theories are not in the scope of proposition 2.

The proof of the preserving property of the rule (U_4) uses an interesting lemma that we state here:

Lemma 1
Let P be a conjunction of non trivial disequations. Let A be a subset of $T(F)$ (resp. $A = RT(F)$) that contains infinitely many ground terms (resp. infinitely many rational trees) for each sort of a variable of P. Then P has at least a solution in A.

Both proofs of the lemma and the proposition are given in appendix B.

Proposition 3
The explosion rule **E** *is $\mathcal{A}$-sound and $\mathcal{A}$-globally preserving when $\mathcal{A}$ is either $T(F)$, $RT(F)$ or any quotient of $T(F)$.*
ED *is $\mathcal{A}$-sound and $\mathcal{A}$-globally preserving for any $\mathcal{A}$.*

The proof of this proposition is given in appendix C.

3.3 Working on Boolean Normal Forms

The previous rules do not make assumptions on the form of the boolean expressions one works on. However, in actual situations, one does not apply the rules modulo the boolean relations but rather uses boolean normal forms, applying a boolean normalization step before any other rule. In this paper, we choose to take conjunctive normal forms, in other words, each equational problem is reduced to a conjunction of disjunctions of normal forms before applying a rule. Our aim indeed is to get rid of disequations first and rules with disjunctions are better suited for this purpose, and among them the elimination of parameters rules **U** play a central role, especially because universal quantifiers that are implicitly associated with parameters go better through conjunctions. On another hand, the **ED** rule eliminates internal disjunctions and so eventually the problems boil down to a disjunction of equations and disequations. The normalization of an expression can disable a rule that was applicable before normalization. This problem is well known from people working on rewriting systems and the purpose of completion procedures is precisely to add new rules in order to avoid this. For instance, consider the rule

$$(M_3) \quad s = t \wedge s \neq u \quad \mapsto \quad s = t \wedge t \neq u.$$

It can be applied on $(s = t \wedge v \neq u) \vee (s = t \wedge Q)$ but not on its conjunctive normal form $s = t \wedge (s \neq u \vee Q)$. This naturally suggests to introduce two new rules of the form

$$\begin{array}{llll} (M_1') & s = t \wedge (s = u \vee Q) & \mapsto & s = t \wedge (t = u \vee Q) \\ (M_3') & s = t \wedge (s \neq u \vee Q) & \mapsto & s = t \wedge (t \neq u \vee Q) \end{array}$$

Similarly, one may introduce a rule

$$(R_1') \quad (s = t \vee Q) \wedge (P \vee Q) \quad \mapsto \quad (s = t \vee Q) \wedge (P(s \leftarrow t) \vee Q).$$

In some sense, the relations between the rules (U_1) and (U_2) on one hand and the rules (U_3) and (U_4) on the other hand are of the same vein. The difference is that they apply on the full equational problem. (U_1) and (U_3) apply on a problem without disjunction.

We don't try to give a complete set of rules obtained in this way. And we don't need such a complete set of rules. Only some of them will be added and given in the figures 6,7,8, 9 and 10. They will be sufficient for proving the results of section 5.

Of course, boolean rules are sound and preserving. Therefore, soundness and preservation of the rules obtained by interaction with boolean rules follow from propositions 1,2 and 3.

4 Completeness and Solved Forms

A "good" set of rules is supposed to transform a problem in a new equivalent presentation, called a *solved form* because it is such that the solutions may be straightforwardly extracted from it. An interesting case, for instance, is when the problem is equivalent to a unification problem: in this case a good solved form, called here *unification solved form* is $x_1 = t_1 \wedge \ldots \wedge x_m = t_m$ where all the unknowns $x_1, \ldots, x_m$ are distinct and do not occur in the t_i's. Obviously, completeness results that prove that a solution is always reachable by the rules will depend on the kind of solved forms one considers. In this paper, we consider in addition to the unification solved forms, the *parameterless solved forms* and the *definitions with constraints*. In the following definition, an algebra $\mathcal{A}$ must be understood: we avoid the prefix $\mathcal{A}$ every time it does not matter.

Definition 8
A set **S** *of rules for solving equational problems is* complete *w.r.t. a kind of solved forms* Σ *(which may be seen as a (syntactically) given subset of equational problems) if for each equational problem* $\mathcal{P}$ *there exists a family of problems* $\mathcal{Q}_i$ *in* Σ*-solved forms such that the* $\mathcal{Q}_i$*'s are obtained from* $\mathcal{P}$ *by applications of the rules in* **S** *and the union of the solutions of the* $\mathcal{Q}_i$*'s is the set of the solutions of* $\mathcal{P}$*.*

One problem with the rules we have presented is that some of them can loop. This is the case for rules (R_1), (R_2) and (E), for example, that increase the size of the expressions. It is then necessary to restrict the application of the rules to prevent such bad situations. Actually we will see later that we can get more than completeness. Indeed it is possible, by adding control, to produce an algorithm which actually stops in any situation and associates with any equational problem a family of solved forms with the same set of solutions. So, in this section, we give no proof since stronger results will be given later on.

4.1 Parameterless Solved Forms

Definition 9
A problem $\mathcal{P}$ is in parameterless solved form *if it contains no* $\forall$.

One can show that some of the rules can be used to transform any equational problem in an equivalent family of problems that have globally the same set of solutions and that do not contain parameters. Before stating the theorem, let us look at an easy example.

Example 7
Consider the equational problem in $T(F_1)$,

$$\forall y : s(x) \neq s(s(y))$$

By decomposition (D_2) one gets

$$\forall y : x \neq s(y).$$

Then by explosion (E), one gets two problems

$$\forall y : x \neq s(y) \wedge x = 0 \tag{1}$$

$$\exists z, \forall y : x \neq s(y) \wedge x = s(z) \tag{2}$$

(1) is equivalent to $\forall y : 0 \neq s(y) \wedge x = 0$ by (R_1) and to $x = 0$ by (C_2) and (EP). Similarly (2) reduces to the empty problem $\perp$. In both cases, the quantifier $\forall$ has been eliminated.

Proposition 4
When $\mathcal{A} = T(F)$, the rules **T**, **M**, **EP**, **C**, **D**, **O**, **U**, **U'** , **E** *are complete for parameterless solved forms.*

The proof of this proposition will follow from theorem 3. It can be extended to the case $\mathcal{A} = RT(F)$. However, some more rules are then needed (for example **RT**) and the occur check has to be removed. See appendix G for more details.

Example 8
We show in this example how the problem $\mathcal{P}$ of Example 3 can be reduced using the rules quoted in proposition 4. (It is assumed that $\mathcal{A} = T(F)$). Actually, we use the algorithm given in section and produce a finite set of parameterless solved forms which is equivalent to $\mathcal{P}$. Figure 5 gives a sequence of reduction of the problems.

Since **E** is only globally preserving we have to look at the two other ways for transforming **(1)** by **E**. This gives the two solved forms:

$$
\begin{aligned}
\forall y_1,y_2,y_3:\quad & (y_1 = x_1 \vee f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)))\\
& \wedge f(y_1,y_1,g(y_2)) \neq f(g(y_2),x_1,x_2) \wedge x_1 \neq f(y_1,y_2,y_3)\\[1ex]
\mapsto_{U_4} \forall y_1,y_2,y_3:\quad & f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge f(y_1,y_1,g(y_2)) \neq f(g(y_2),x_1,x_2)\\
& \wedge x_1 \neq f(y_1,y_2,y_3)\\[1ex]
\mapsto_{D_2} \forall y_1,y_2,y_3:\quad & f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge (y_1 \neq g(y_2) \vee y_1 \neq x_1 \vee g(y_2) \neq x_2)\\
& \wedge x_1 \neq f(y_1,y_2,y_3)\\[1ex]
\mapsto_{U_2} \forall y_1,y_2,y_3:\quad & f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge (g(y_2) \neq x_1 \vee g(y_2) \neq x_2)\\
& \wedge x_1 \neq f(y_1,y_2,y_3) \qquad\qquad \textbf{(1)}\\[1ex]
\mapsto_{E} \exists x_5, \forall y_1,y_2,y_3:\quad & f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge (g(y_2) \neq x_1 \vee g(y_2) \neq x_2)\\
& \wedge x_1 \neq f(y_1,y_2,y_3) \wedge x_1 = g(x_5)\\[1ex]
\mapsto_{M_3} \exists w, \forall y_1,y_2,y_3:\quad & f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge (g(y_2) \neq x_1 \vee g(y_2) \neq x_2)\\
& \wedge g(x_5) \neq f(y_1,y_2,y_3) \wedge x_1 = g(x_5)\\[1ex]
\mapsto_{C_2} \exists x_5, \forall y_1,y_2,y_3:\quad & f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge (g(y_2) \neq x_1 \vee g(y_2) \neq x_2) \wedge x_1 = g(x_5)\\[1ex]
\mapsto_{M_3'} \exists x_5, \forall y_1,y_2,y_3:\quad & f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge (g(y_2) \neq g(x_5) \vee g(y_2) \neq x_2)\\
& \wedge x_1 = g(x_5)\\[1ex]
\mapsto_{D_2} \exists x_5, \forall y_1,y_2,y_3:\quad & f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge (y_2 \neq x_5 \vee g(y_2) \neq x_2) \wedge x_1 = g(x_5)\\[1ex]
\mapsto_{U_2} \exists x_5, \forall y_1,y_2,y_3:\quad & f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge g(x_5) \neq x_2 \wedge x_1 = g(x_5)\\[1ex]
\mapsto_{EP} \exists x_5:\quad & f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge g(x_5) \neq x_2 \wedge x_1 = g(x_5)
\end{aligned}
$$

Figure 5: An sequence of reductions for reaching parameterless solved forms

- $f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge x_1 = a$ (corresponding to the case $x_1 = a$)
- $\perp$ (corresponding to the case $x_1 = f(w,x_6,x_7)$)

These three problems have the same set of solutions than $\mathcal{P}$ and are in parameterless solved form.

4.2 Definitions with Constraints

An acceptable solved form for equational problems containing disequations is a presentation of the problem like $x_1 = t_1 \wedge \ldots \wedge x_m = t_m \wedge x'_1 \neq u_1 \wedge \ldots \wedge x'_p \neq u_p$ with the restrictions given in the next definition. We call it a definition with constraints because the first part, made of equalities, describes or defines a generic substitution and the set of disequalities tells the constraints a substitution has to satisfy in order to be accepted as a solution. Of course, the definition of such solved forms depends on whether we are working in a rational trees algebra or not. Indeed an equation $x = f(x)$ is a solved form in $RT(F)$ although it can be reduced to $\perp$ by the rules **O** in the case of finite trees.

4.2.1 Finite Trees

We assume throughout this subsection that $\mathcal{A}$ is contained in $T(F,X)$.

Definition 10
A problem is a definition with constraints *if it is either* $\top$, $\perp$ *or a conjunction of equalities and disequalities* $\exists w_1,\ldots,w_k : x_1 = t_1 \wedge \ldots \wedge x_m = t_m \wedge x'_1 \neq u_1 \wedge \ldots \wedge x'_p \neq u_p$*, where*

1. *all the unknowns* $x_1,\ldots,x_m$ *occur only once in the problem*
2. *for every index* $i = 1,\ldots,p$*,* x'_i *is a sort-unrestricted variable distinct from* u_i*.*

Theorem 2
Let $\mathcal{A} \subseteq T(F,X)$*. Then, the rules* **T**, **M**, **EP**, **C**, **D**, **O**, **U**, **E**, **ED** *are complete for the solved forms* definitions with constraints.

This follows from theorem 4.

Example 9
We start from the solved form of figure 5 and get an equivalent problem which is a definition with constraints.

$$\begin{gathered}
\exists w:\ f(x_1,x_4,x_4) = f(x_1,x_2,g(x_3)) \wedge g(w) \neq x_2 \wedge x_1 = g(w) \\
\downarrow D_1 \\
\exists w:\ x_1 = x_1 \wedge x_4 = x_2 \wedge x_4 = g(x_3) \wedge g(w) \neq x_2 \wedge x_1 = g(w) \\
\downarrow T_1 \\
\exists w:\ x_4 = x_2 \wedge x_4 = g(x_3) \wedge g(w) \neq x_2 \wedge x_1 = g(w) \\
\downarrow R_1 \\
\exists w:\ g(x_3) = x_2 \wedge x_4 = g(x_3) \wedge g(w) \neq x_2 \wedge x_1 = g(w) \\
\downarrow M_3 \\
\exists w:\ g(x_3) = x_2 \wedge x_4 = g(x_3) \wedge g(w) \neq g(x_3) \wedge x_1 = g(w) \\
\downarrow D_2 \\
\exists w:\ x_1 = g(w) \wedge x_2 = g(x_3) \wedge x_4 = g(x_3) \wedge x_3 \neq w
\end{gathered}$$

4.2.2 Rational Trees

In this subsection, we will assume that $\mathcal{A} = RT(F)$. A *cycle of variables* is a system $x_1 = x_2 \wedge \ldots \wedge x_{n-1} = x_n \wedge x_n = x_1$ where $n \geq 1$ and $x_1, \ldots, x_n$ are distinct variables.

Definition 11
When $\mathcal{A} = RT(F)$, an equational problem is a definition with constraints *if it is either $\top$, $\bot$ or has the form $\exists w_1, \ldots, w_k : x_1 = t_1 \wedge \ldots \wedge x_m = t_m \wedge x'_1 \neq u_1 \wedge \ldots \wedge x'_n \neq u_n$ where:*

1. *the unknowns $x_1, \ldots, x_m$ are distinct*
2. *there is no cycle of variables*
3. *for every index $i = 1, \ldots, p$, x'_i is a sort-unrestricted variable distinct from u'_i.*
4. $\{x_1, \ldots, x_m\} \cap \{x'_1, \ldots, x'_n, u_1, \ldots, u_n\} = \emptyset$

In the case of rational trees, a proposition similar to proposition 4 holds. However we only focus our attention on finite trees. See appendix G for an idea of the extension to this case.

4.3 Unification Solved Forms

Definition 12
A Unification solved form *is a definition with constraints which does not contain any disequation.*

Definition 13
An equational problem $\mathcal{P}$ is said to be equivalent to a unification problem *(we write EUP for short) if there is a finite set of equational problems $\mathcal{P}_1, \ldots, \mathcal{P}_n$ whose solutions (restricted to $Var(\mathcal{P})$) are those of $\mathcal{P}$ and which do not contain any disequations nor parameters.*

In order to reach unification problems, the **CR** rules are needed. (Note we have not yet used them). They “clean up” the problems, removing useless equations and disequations.

Definition 14
An ELD-problem *is an equational problem whose conjunctive normal form is $\exists \vec{w}, \forall \vec{y} : d_1 \wedge \ldots \wedge d_n$ where each d_i is either an equation, a disequation or a disjunction of disequations.*

This is of course a restricted class of equational problems but it still contains complement problems as in Lassez & Mariott (1987). The following result is not proved in this paper (it would need a full paper by itself):

Starting from problems having both the properties EUP and ELD, the rules of figures 1,2,4 (except for the **RT** *rule) are complete for unification solved forms in $T(F)$.*

Actually, a more general result holds, since we still have completeness with a restricted control which insures termination. It proves that, when it is possible to turn the disequations into equations, the algorithm does it. This is a generalization of the result of Lassez & Marriott (1986). The proof will be given in a forthcoming paper. A French version can already be found in Comon (1988).

4.4 Other Solved Forms

Other solved forms can be considered, depending on the application at hand. For example, it is possible to impose that "there are no cycles in the disequations". This means that $x \neq f(y) \wedge y \neq f(x)$ would not be in solved form. Also, the rules given above are complete for such solved forms. Again, the termination requires a control which is not given in this paper. The reader is referred to Comon (1988) for more details and/or other solved forms.

Finally, note that, for other purposes (improving AC-unification), HJ. Bürckert in Bürckert (1988b) uses another kind of solved forms called "substitutions with exceptions". We don't study such solved forms in this paper.

5 Algorithms for Solving Equational Problems

According to a certain usage wedistinguish between an algorithm and a procedure. An algorithm is a procedure which always terminates and returns a result. In this section, we prove that there exists an algorithm that returns a set of solutions for any equational problem. This algorithm is described by adding more control to the rules, trying to keep as liberal as possible. Actually the control can be either strengthened to improve the efficiency or weakened to allow more freedom. This has to be done carefully to avoid loosing completeness on one side and termination on the other side. In order to be clearer, we first eliminate the parameters and then try to reach definitions with constraints. Actually, such a strict control is not necessary (Comon 1988). However, mixing the two steps would lead to some confusion, without giving much more results. Moreover, we only consider the case $\mathcal{A} = T(F)$ in this section.

Before starting to give the termination results, we need to recall some basic definitions on multisets which are used in the termination's proofs. Such results can be found e.g. in Dershowitz & Manna (1979).

5.1 Multiset Orderings

We assume that $\mathcal{E}$ is a set, together with an ordering $\geq$. A (finite) *multiset M* of elements of $\mathcal{E}$ is an application from $\mathcal{E}$ in $\mathbf{N}$, the set of non-negative integers such that there is only a finite number of elements x in $\mathcal{E}$ satisfying $M(x) \neq 0$. Usually, a multiset is denoted by repeating x n times when $M(x) = n$. In this way, $\{a,a,a,b,a,b\}$ denotes the multiset M such that $M(a) = 4$, $M(b) = 2$ and $M(x) = 0$ for every x distinct from a and b.

The ordering on $\mathcal{E}$ is extended to the multisets of elements in $\mathcal{E}$ by the following (recursive) definition:

$$X = \{x_1,\ldots,x_n\} \succeq \{y_1,\ldots,y_n\} = Y$$

iff one of the following holds

1 $X = Y$
2 $\exists i \in \{1,\ldots,m\},\ \exists j \in \{1,\ldots,n\},\ x_i = y_j \ \ and \ \ X - \{x_i\} \succeq Y - \{y_j\}$
3 $\exists Z \subseteq Y, \exists x \in X, \forall y \in Z,\ \ x > y \ \ and \ \ X - \{x\} \succeq Y - Z$

An ordering is *well founded* if there exists no infinite decreasing sequence. The following result is well-known (see e.g Dershowitz & Manna (1979) where a more general version is given).

The ordering $\geq$ on $\mathcal{E}$ is well-founded iff its multiset extension $\succeq$ is well founded.

5.2 Elimination of Parameters from Equational Problems

In figures 6, 7 and 8 we give the rules used in the algorithm, together with a control which insures termination. Some of the rules given there are obtained from interaction with boolean rules. The rule D_3 is nothing but the combination of D_1 and the boolean normalization. Thus, it could be avoided. However it is given here in order to simplify the expression of the control.

In order to express this control, we use the notion of *solved parameter*. A parameter y is a solved parameter in a disjunction of equations and disequations d if there exists a disequation $y \neq u$ in d and y occurs only once in d.

Moreover we use the function size-parameter(t) which denotes the sum of the sizes of the parameter's positions in t. For example,

$$\text{size-parameter}(f(y_1, g(y_1), g(g(y_2)))) = 6$$

if both y_1 and y_2 are parameters.

Theorem 3

Let $\mathcal{A} = T(F)$. The non deterministic application of the rules given in figure 6, 7 and 8 always terminates. Moreover, irreducible problems for these rules are in parameterless solved form.

Proof: We only sketch the proof of termination. The full proofs of both termination and completeness are given in appendix D.

We construct some "interpretation" functions which are intended to decrease by applications of the rules:

- Given a disjunction of equations and disequations d, $\phi_1(d)$ is the number of distinct parameters in d.

Elimination of Parameters: EP

$$(EP) \quad \forall \vec{y}, y : P \quad \mapsto \quad \forall \vec{y} : P \quad \text{if } y \notin Var(P)$$

Universality of Parameters: U

$$\begin{array}{lccc}
(U_1) & \forall \vec{y} :\ P \wedge y \neq t & \mapsto & \perp \\
 & & \text{if } y \notin Var(t) \ \&\ y \in \vec{y} & \\
(U_2) & \forall \vec{y} :\ P \wedge (y \neq t \vee R) & \mapsto & \forall \vec{y} :\ P \wedge R(y \leftarrow t) \\
 & & \text{if } y \notin Var(t) \ \&\ y \in \vec{y} & \\
(U_3) & \forall \vec{y} : P \wedge z = t & \mapsto & \perp
\end{array}$$

The rule (U_3) is only used if $z \not\equiv t$ and there exists $y \in Var(z = t) \cap \vec{y}$ such that $T(F)_{sort(y)})$ contains at least two terms.

$$(U_4) \quad \forall \vec{y} :\ P \wedge (z_1 = u_1 \vee \ldots \vee z_n = u_n \vee R) \quad \mapsto \quad \forall \vec{y},\ P \wedge R$$

If
1. for each index i, z_i is a variable and $z_i \not\equiv u_i$,
2. for each index i, $z_i = u_i$ contains at least one occurrence of a parameter,
3. for each index i and any parameter $y \in Var(z_i, u_i)$, y is sort unrestricted,
4. R does not contain any parameter.

$$(U_5) \quad \forall \vec{y} : P \wedge Q \quad \mapsto \quad \forall \vec{y} : P \wedge Q(y \leftarrow t_1) \wedge \ldots \wedge Q(y \leftarrow t_n)$$

If y is a parameter, of sort $\underline{s}$, $y \in Var(Q)$ and $\{t_1, \ldots, t_n\} = T(F)_{\underline{s}}$.

Figure 6: Elimination of parameters in free theories

Merging: M

$$\begin{array}{lrcl}
(M_1) & z = t \wedge z = u & \mapsto & z = t \wedge t = u \\
(M_3) & z = t \wedge z \neq u & \mapsto & z = t \wedge t \neq u \\
(M_1') & z = t \wedge (z = u \vee Q) & \mapsto & z = t \wedge (t = u \vee Q) \\
(M_3') & z = t \wedge (z \neq u \vee Q) & \mapsto & z = t \wedge (t \neq u \vee Q)
\end{array}$$

For theserules one supposes that

1. z is an unknown and t is not a variable,
2. t does not contain any parameter,
3. u does contain parameters and is not a variable.

$$\begin{array}{lrcl}
(M_2) & z \neq t \vee z \neq u & \mapsto & z \neq t \vee t \neq u \\
(M_4) & z \neq t \vee z = u & \mapsto & z \neq t \vee t = u
\end{array}$$

For these rules, one supposes that

1. z is a variable and t is not a variable,
2. u does contain a parameter
3. Either size-param$(t) \leq$ size-param(u) or u is a solved parameter.

Elimination of trivial equations and disequations: T

$$\begin{array}{lrcl}
(T_1) & t = t & \mapsto & \top \\
(T_2) & t \neq t & \mapsto & \bot
\end{array}$$

Figure 7: Elimination of parameters (continued)

Clash: C

$$
\begin{array}{llcll}
(C_1) & f(t_1,\ldots,t_m) = g(u_1,\ldots,u_n) & \mapsto & \bot & \text{if } f \neq g \\
(C_2) & f(t_1,\ldots,t_m) \neq g(u_1,\ldots,u_n) & \mapsto & \top & \text{if } f \neq g
\end{array}
$$

Decomposition: D

$$
\begin{array}{llcl}
(D_1) & f(t_1,\ldots,t_m) = f(u_1,\ldots,u_m) & \mapsto & t_1 = u_1 \wedge \ldots \wedge t_m = u_m \\
(D_2) & f(t_1,\ldots,t_m) \neq f(u_1,\ldots,u_m) & \mapsto & t_1 \neq u_1 \vee \ldots \vee t_m \neq u_m
\end{array}
$$

$$
\begin{array}{ll}
(D_3) & P \wedge (f(t_1,\ldots,t_n) = f(u_1,\ldots,u_n) \vee Q) \\
& \qquad\qquad \mapsto\ P \wedge (t_1 = u_1 \vee Q) \wedge \ldots \wedge (t_n = u_n \vee Q)
\end{array}
$$

The decomposition rules are only used when $f(t_1,\ldots t_n)$ or $f(u_1,\ldots u_n)$ contains at least one occurrence of a parameter.

Occur Check: O

$$
\begin{array}{llcll}
(O_1) & z = t & \mapsto & \bot & \text{if } z \in Var(t)\ \&\ z \not\equiv t \\
(O_2) & z \neq t & \mapsto & \top & \text{if } z \in Var(t)\ \&\ z \not\equiv t
\end{array}
$$

Explosion: E

$$
(E) \quad \forall \vec{y} : P \quad \rightharpoonup \quad \exists w_1,\ldots,w_p, \forall \vec{y} : P \wedge x = f(z_1,\ldots,z_p)
$$

If 1. x is an unknown and $\vec{w} \cap (Var(P) \cup \vec{y} \cup I) = \emptyset$ and $f \in F$,
2. there exist an equation $x = u$ (or a disequation $x \neq u$) in P
such that u is not a variable and contains at least one parameter,
3. **U**, **M**, **D**, **C**, **O**, **EP**, **T** do not apply.

Figure 8: Elimination of parameters (end)

- Given a disjunction of equations and disequations $d = e_1 \vee \ldots \vee e_n$, $\phi_2(d)$ is the multiset $\{MSP(e_1), \ldots, MSP(e_n)\}$ where $MSP(e)$ is defined by:

  - $MSP(e) = 0$ if a member of e is a solved parameter
  - Otherwise, $MSP(s = t) = MSP(s \neq t) = \max(\text{size-param}(s),\text{size-param}(t))$.

  For example,

  $$\phi_2(y_1 \neq f(g(g(g(y_3))), a) \vee y_3 \neq g(y_2) \vee g(y_4) = g(g(y_5))) = \{0, 2, 3\}$$

  if the y_i's are assumed to be parameters.

- Let d be again a disjunction of equations and disequations, $\phi_3(d)$ is the number of equations and disequations in d having (at least) one variable as a member.

- If $\mathcal{P} \equiv \exists \vec{w}, \forall \vec{y} :\ d_1 \wedge \ldots \wedge d_n$ is a problem in conjunctive normal form, $\psi_1(\mathcal{P})$ is the multiset $\{(\phi_1(d_1), \phi_2(d_1), \phi_3(d_1)), \ldots, (\phi_1(d_n), \phi_2(d_n), \phi_3(d_n))\}$.

- If $\mathcal{P}$ is again an equational problem, then $\psi_2(\mathcal{P})$ is the total size of $\mathcal{P}$ (i.e. the number of operators and variable symbols in $\mathcal{P}$).

We first prove that the function $\Phi = (\psi_1, \psi_2)$ is strictly decreasing by application of any rule, except for the explosion rule. Since the domain of Φ is obtained by lexicographic and multiset extensions of the set of natural numbers, this proves the termination of the rules when **E** is not considered.

Then, we prove that, whenever $\mathcal{P} \mapsto_{\mathbf{E}} \mathcal{P}'$ using the explosion rule, for every $\mathcal{P}''$ such that $\mathcal{P}' \mapsto \mathcal{P}''$, $\Phi(\mathcal{P}'') < \Phi(\mathcal{P})$. This completes the proof, since, assuming that there is an infinite transformation chain, we could extract an infinite sequence of problems for which Φ is strictly decreasing, which is absurd.

Note that proposition 4 is nothing but a consequence of theorem 3 since theorem 3 proves the completeness for a particular control.

The termination proof holds in other algebras $\mathcal{A}$. In particular, removing the **O** rules, we get a correct and terminating set of rules in $RT(F)$. However, in this case, irreducible problems may still contain parameters. Actually, some more rules are needed for a completeness result in $RT(F)$. (In particular the **RT** rule). The reader is referred to appendix G for more details on rational trees.

5.3 Definitions with Constraints

Now (because of theorem 3), we may assume that the problems we are working on do not contain any parameter. The rules given in figures 9 and 10 provide algorithms for the simplification of parameterless problems into definitions with constraints. We try here to keep as much freedom as possible. In particular, replacements may be postponed, as well as elimination of disjunctions. These two features (among others) allow deriving

Elimination of trivial equations and disequations: T

$$\begin{array}{lccc} (T_1) & t = t & \mapsto & \top \\ (T_2) & t \neq t & \mapsto & \bot \end{array}$$

Replacement: R

$$(R_1) \quad z = t \wedge P \quad \mapsto \quad z = t \wedge P(z \leftarrow t)$$

If $z \notin Var(t)$, z occurs in P, t does not contain any parameter and either t is not a variable or t occurs in P

Merging: M

$$\begin{array}{lrcl} (M_1) & z = t \wedge z = u & \mapsto & z = t \wedge t = u \\ (M_2) & z \neq t \vee z \neq u & \mapsto & z \neq t \vee t \neq u \\ (M_3) & z = t \wedge z \neq u & \mapsto & z = t \wedge t \neq u \\ (M_4) & z = t \vee z \neq u & \mapsto & t = u \vee z \neq u \\ (M_1') & z = t \wedge (z = u \vee Q) & \mapsto & z = t \wedge (t = u \vee Q) \\ (M_3') & z = t \wedge (z \neq u \vee Q) & \mapsto & z = t \wedge (t \neq u \vee Q) \end{array}$$

Where z is a variable, t is not a variable and either $size(t) \leq size(u)$ or u is a solved variable. (Recall that the size of a term is the number of its nodes).

Clash: C

$$\begin{array}{llccl} (C_1) & f(t_1, \ldots, t_m) = g(u_1, \ldots, u_n) & \mapsto & \bot & \text{if } f \not\equiv g \\ (C_2) & f(t_1, \ldots, t_m) \neq g(u_1, \ldots, u_n) & \mapsto & \top & \text{if } f \not\equiv g \end{array}$$

Figure 9: Rules for the transformation into definitions with constraints

Decomposition: D

$$\begin{array}{llll}(D_1) & f(t_1,\ldots,t_m) = f(u_1,\ldots,u_m) & \mapsto & t_1 = u_1 \wedge \ldots \wedge t_m = u_m \\ (D_2) & f(t_1,\ldots,t_m) \neq f(u_1,\ldots,u_m) & \mapsto & t_1 \neq u_1 \vee \ldots \vee t_m \neq u_m\end{array}$$

$$\begin{array}{ll}(D_3) & P \wedge (f(t_1,\ldots,t_n) = f(u_1,\ldots,u_n) \vee Q) \\ & \qquad\qquad \mapsto P \wedge (t_1 = u_1 \vee Q) \wedge \ldots \wedge (t_n = u_n \vee Q)\end{array}$$

Occur Check: O

$$\begin{array}{lllll}(O_1) & z = t & \mapsto & \bot & \text{if } z \in Var(t) \;\&\; z \not\equiv t \\ (O_2) & z \neq t & \mapsto & \top & \text{if } z \in Var(t) \;\&\; z \not\equiv t\end{array}$$

Explosion: E

$$(E) \quad P \quad \rightharpoonup \quad P \wedge x = u$$

If 1. P contains a disequation $x \neq t$ such that $\mathcal{A}_{sort(x)}$ is finite and $u \in \mathcal{A}_{sort(x)}$.
2. **M**, **O**, **R**, **C**, **D** do not apply

Elimination of disjunctions: ED

$$(ED) \quad \forall \vec{y} : P \wedge (P_1 \vee P_2) \quad \rightharpoonup \quad \forall \vec{y} : P \wedge P_1$$

If $Var(P_1) \cap \vec{y} = \emptyset$ or $Var(P_2) \cap \vec{y} = \emptyset$

Figure 10: Rules for the transformation into definitions with constraints (end)

algorithms for infinite trees and for finite trees as well. They also delay splitting the problem which is particularly useful for the unification solved forms.

Theorem 4

Let $\mathcal{A} \subseteq T(F,X)$. The non deterministic application of the rules of figure 9 and 10 to any parameterless problem terminates. Moreover, irreducible problems for these rules are definitions with constraints.

Proof: Like above, we only give the ordering for the proof of termination. The complete proofs of both termination and completeness can be found in appendix E. We give a function Φ which is decreasing by any application of the rules and whose codomain is a well-founded ordered set. Let us introduce concepts which are necessary for the expression of this function. $\mathcal{P} \equiv d_1 \wedge \ldots \wedge d_n$ is a problem in conjunctive normal form: the d_i's are disjunctions of one or more equations and disequations. If $d_i \equiv z = t$ where z is a variable and $z \not\equiv t$, then z is called an *almost solved variable of $\mathcal{P}$*. A *solved variable of $\mathcal{P}$* is an almost solved variable of $\mathcal{P}$ which occurs only once in $\mathcal{P}$.

Let $\Phi = (\phi_1, \phi_2, \phi_3, \phi_4)$ where:

$\phi_1(\mathcal{P})$ is the number of variables of $\mathcal{P}$ which are not almost solved,

$\phi_2(\mathcal{P})$ is the number of unsolved variables of P,

$\phi_3(d_1 \wedge \ldots \wedge d_n)$ where $d_1, \ldots, d_n$ are disjunctions of (one or more) equations and disequations, is the multiset $\{M(d_1), \ldots, M(d_n)\}$ where $M(d)$ is the multiset of numbers $MS(e)$, for each equation and each disequation in d. $MS(e)$ is equal to 0 if e contains a solved variable; otherwise it is equal to the maximal size of its two members.

$\phi_4(\mathcal{P}$**)** is the total number of variable occurrences as a member of an equation or a disequation.

It must be noted that, since we allow as much freedom as possible, it is possible to deduce easily theorem 2 from theorem 4. Also, it is possible to delay the application of **ED**. This is necessary if we want to reach unification solved forms (when they do exist).

Now, we have to show that definitions with constraints are suitable solved forms. In other words, we show that every problem which is in "definition with constraints" solved form has at least one ground solution.

5.4 Solvable Problems

The following result is similar to those given in Colmerauer (1984) and Lassez, Maher & Mariott (1986). Indeed, it shows that, provided they are not empty, problems we have obtained always have a solution.

Proposition 5 *Let $\mathcal{A}$ be either $T(F)$ or $RT(F)$. A problem in* definition with constraints solved form *has at least one solution if and only if it is syntactically different from $\perp$.*

This result is a consequence of lemma 1. Its complete proof is given in appendix F.

6 Extensions

6.1 Extension to an Arbitrary Number of Quantifiers

Let $\mathcal{A} = T(F)$. Because of the results of theorem 3, for any equational problem $\mathcal{P}$, it is possible to find a finite set of parameterless problems whose set of solutions is equal to the set of solutions of $\mathcal{P}$. This may be viewed as a transformation of $\mathcal{P}$ into a formula $\mathcal{P}_1 \vee \ldots \vee \mathcal{P}_n$ where the $\mathcal{P}_i$'s are parameterless problems. This transformation "eliminates the innermost quantifier" in the formula $\mathcal{P}$, when it is a universal one. This transformation is still available if $\mathcal{P}$ is surrounded by other quantifiers. Moreover, assuming that a problem $\mathcal{P}$ has the form $Q\forall\ldots\exists\ldots P$ where Q is a sequence of quantifiers, then its set of solutions is equal to the complement of the set of solutions of $not(\mathcal{P}) \equiv not(Q)\exists\ldots\forall\ldots not(P)$. Now, if we forget about $not(Q)$, we get an equational problem which can be turned into a disjunction of parameterless problems. Taking again the complement, we obtain a problem which is equivalent to $\mathcal{P}$ and where the innermost quantifier is eliminated. By repeating such a transformation, a problem with any number of quantifier is turned into a problem with at most one quantifier. Finally, since the prenex normal forms of a first order formula (with the only predicate symbol =) are precisely equational problems (or their negation) surrounded by a sequence of quantifiers, it is possible to transform any such formula into a formula with only one quantifier and which has the same set of solutions. Now, if this quantifier is an existential one, theorems 4 and 5 provide a decision procedure for the existence of a solution (in $T(F)$). If this is a universal one, applying again the transformation of section 5.1, we get a formula containing only existential quantifiers, and we are back to the previous case.

Essentially, the same method is used by M.J. Maher in Maher (1988). He eliminates the existential quantifiers whereas we eliminate the universal ones.

6.2 Extensions to Equational Theories

Let us recall that the results of sections 4 and 5 do not hold in equational theories (i.e. when $\mathcal{A} = T(F)/=_E$ where $=_E$ is a non trivial congruence). Actually, some extensions to equational theories are investigated in Comon (1988). It is shown that the method presented in this paper can be extended in the case of *quasi-free* theories, which include the commutativity case.

However, we cannot expect to extend our results to any finitary equational theories as in Bürckert (1988b). Indeed, *word problems* are equational problems and there exists equational theories in which the word problems are decidable whereas unification is not (Bürckert 1988a).

Acknowledgments

We would acknowledge JP. Jouannaud, C. Kirchner, Ph. Schnoebelen and the referees for their valuable comments on earlier drafts of this paper.

References

[Bur88a] H. J. Bürckert. 1988. Private communication.

[Bur88b] H. J. Bürckert. Solving disequations in equational theories. In *Proc. 9th Conf. Automated Deduction*, Springer-Verlag, 1988.

[Col84] A. Colmerauer. Equations and inequations on finite and infinite trees. In *FGCS'84 Proceedings*, pages 85–99, November 1984.

[Com86] H. Comon. Sufficient completeness, term rewriting systems and anti-unification. In *Proc. 8th Conf. on Automated Deduction, Oxford, LNCS 230*, pages 128–140, Springer-Verlag, 1986.

[Com88] H. Comon. *Unification et Disunification: Théorie et Applications*. Thèse de Doctorat, I.N.P. de Grenoble, France, 1988.

[DM79] N. Dershowitz and Z. Manna. Proving termination with multisets orderings. *Communications of the ACM*, 22:465–476, 1979.

[GH78] J. V. Guttag and J. J. Horning. The algebraic specification of abstract data types. *Acta Informatica*, 10:27–52, 1978.

[GS87] J. Gallier and W. Snyder. A general complete E-unification procedure. In *Proc. Rewriting Techniques and Applications 87, Bordeaux, LNCS 256*, Springer-Verlag, May 1987.

[HO80] G. Huet and D. Oppen. Equations and rewrite rules: a survey. In R. Book, editor, *Formal Language Theory: Perspectives and Open Problems*, pages 349–405, Academic Press, 1980.

[Hue76] G. Huet. *Résolution d'équations dans les langages d'ordre* $1,2,\ldots\omega$. Thèse d'Etat, Univ. Paris 7, 1976.

[JK86] J.-P. Jouannaud and E. Kounalis. Automatic proofs by induction in equational theories without constructors. In *Proc. 1st IEEE Symp. Logic in Computer Science, Cambridge, Mass.*, June 1986.

[Kir85] C. Kirchner. *Méthodes et Outils de Conception Systématique d'Algorithmes d'Unification dans les Théories equationnelles*. Thèse d'Etat, Univ. Nancy, France, 1985.

[Kir86] C. Kirchner. Computing unification algorithms. In *Proc. 1st IEEE Symp. Logic in Computer Science, Cambridge, Mass.*, pages 206–216, 1986.

[KL87] C. Kirchner and P. Lescanne. Solving disequations. In *Proc. 2nd IEEE Symp. Logic in Computer Science, Ithaca, NY*, pages 347–352, 1987.

[Kuc88] G. A. Kucherov. A new quasi-reducibility testing algorithm and its applications to proofs by induction. In *Proc. Workshop on Algebraic and Logic Programming*, 1988. To appear.

[Lav87] A. Laville. Lazy pattern matching in the ML language. In *Proc. 7th Conf. Found. of Software Technology and Theoretical Computer Science, Pune, INDIA, LNCS 287*, December 1987.

[LLT86] A. Lazrek, P. Lescanne, and J.-J. Thiel. *Proving Inductive Equalities. Algorithms and Implementation*. Research Report, CRIN, Nancy, France, 1986. To appear in Information and Control.

[LM87] J.-L. Lassez and K. Marriott. Explicit representation of terms defined by counter examples. *J. Automated Reasoning*, 3(3), September 1987.

[LMM86] J.-L. Lassez, M. J. Maher, and K. G. Marriot. *Unification Revisited*. Research Report RC 12394, IBM T. J. Watson Research Center, November 1986.

[Mah88] M. J. Maher. Complete axiomatization of the algebra of finite, rational and infinite trees. January 1988. Draft Paper.

[MM82] A. Martelli and U. Montanari. An efficient unification algorithm. *ACM Transactions on Programming Languages and Systems*, 4(2):258–282, 1982.

[Rob65] J. A. Robinson. A machine-oriented logic based on the resolution principle. *Journal of the ACM*, 12(1):23–41, 1965.

[Sch88] Ph. Schnoebelen. *Refined Compilation of Pattern-Matching for Functional Languages*. Research Report Lifia 71 Imag 715, Univ. Grenoble, April 1988. Submitted for publication.

[Sie84] J. Siekmann. Universal unification. In *Proc. 7th Conf. on Automated Deduction, Napa, LNCS 87*, Springer-Verlag, May 1984.

[Thi84] J. J. Thiel. Stop loosing sleep over incomplete specifications. In *Proc. ACM Symp. Principles of Programming Languages*, 1984.

A Proof of Proposition 1

Let us recall the statement of proposition 1:

proposition 1
The rules of figure 1 are $\mathcal{A}$-sound and $\mathcal{A}$-preserving for any algebra $\mathcal{A}$.

The sets of solutions are “monotonic” in the following sense: If $\mathcal{S}(\mathcal{P}_1,\mathcal{A}) \subseteq \mathcal{S}(\mathcal{P}_2,\mathcal{A})$, then, for any equational problem $\mathcal{Q}$ and any finite sets $\vec{y},\vec{w}$ of variables,

$$\begin{array}{rcl}
\mathcal{S}(\mathcal{P}_1 \vee \mathcal{Q},\mathcal{A}) & \subseteq & \mathcal{S}(\mathcal{P}_2 \vee \mathcal{Q},\mathcal{A}) \\
\mathcal{S}(\mathcal{P}_1 \wedge \mathcal{Q},\mathcal{A}) & \subseteq & \mathcal{S}(\mathcal{P}_2 \wedge \mathcal{Q},\mathcal{A}) \\
\mathcal{S}(\forall\vec{y}:\mathcal{P}_1,\mathcal{A}) & \subseteq & \mathcal{S}(\forall\vec{y},\mathcal{P}_2,\mathcal{A}) \\
\mathcal{S}(\exists\vec{w}:\mathcal{P}_1,\mathcal{A}) & \subseteq & \mathcal{S}(\exists\vec{w},\mathcal{P}_2,\mathcal{A})
\end{array}$$

In the case of the soundness + preservation proof of a rule $L \to R$, it is therefore sufficient to show that $\mathcal{S}(L,\mathcal{A}) = \mathcal{S}(R,\mathcal{A})$.

In these proofs, we sometimes omit the $\mathcal{A}$ prefix, which is not relevant. Moreover, we will make use of the “away-properties” of definition 5 without any mention.

We only give the proof for the (U_2) rule (universality of parameters) which is not obvious and for the rule (M_2). (The others are in Comon 1988). Let us recall these rules:

$$\begin{array}{lrcl}
(M_2) & s \neq t \vee s \neq u & \mapsto & s \neq t \vee t \neq u \\
(U_2) & \forall\vec{y}: P \wedge (y \neq t \vee R) & \underset{\text{if } y \in \vec{y}}{\mapsto} & \forall\vec{y}: P \wedge R(y \leftarrow t)
\end{array}$$

The rule (M_2) is sound and preserving. Let P≡ $s \neq t \,\vee\, s \neq u$ and Q≡ $s \neq t \,\vee t \neq u$. We have to show that the set of solutions of P and the set of solutions of Q are equal. Notice that, if $\mathcal{P} \mapsto \mathcal{Q}$ by (M_2), then $\mathcal{Q} \mapsto \mathcal{P}$ by (M_2). Thus, it is sufficient to prove that the solutions of P are solutions of Q. Actually, it is sufficient to prove that the substitutions that validate P validate Q .

Let σ be a substitution that validates P. Then, either $s\sigma \neq_{\mathcal{A}} t\sigma$ and σ obviously validates Q or $s\sigma =_{\mathcal{A}} t\sigma$ and $s\sigma \neq_{\mathcal{A}} u\sigma$. In this last case, we have $t\sigma \neq_{\mathcal{A}} u\sigma$ which means that σ validates Q.

The rule (U_2) is sound. We have to prove that the solutions of the right hand side of the rule are solutions of the left hand side of the rule. Note that the solutions of a problem $\forall\vec{y}: P \wedge Q$ is equal to the intersection of the sets of solutions of $\forall\vec{y}: P$ and $\forall\vec{y}: Q$ respectively. It is thus sufficient to prove that any solution σ of $\forall\vec{y}: R(y \leftarrow t)$ is also a solution of $\forall\vec{y}: y \neq t \vee R$.

Let σ be a $\mathcal{A}$-solution of $\forall\vec{y}: R(y \leftarrow t)$ and ψ any $\mathcal{A}$-substitution with domain $\vec{y}$, we have to prove that $\psi\sigma$ validates $y \neq t \vee R$.

- *First case*: $y\psi =_{\mathcal{A}} t\psi$ then $\psi\sigma \equiv (\psi \circ (y \leftarrow t))\sigma$.
  On the other hand, $\psi\sigma$ validates $R(y \leftarrow t)$ by hypothesis. Therefore $(\psi \circ (y \leftarrow t))\sigma$ validates R.
  From these two statements we deduce that $\psi\sigma$ validates R and thus validates $y \neq t \vee R$.
- *Second case*: $y\psi \neq_{\mathcal{A}} t\psi$, then $\psi\sigma$ validates $y \neq t$.

The rule (U_2) is preserving. Let σ be a solution of $\forall \vec{y} : y \neq t \vee R$. We have to prove that σ is also a solution of $\forall \vec{y} : R(y \leftarrow t)$.
Let ψ be any substitution whose domain is $\vec{y}$. $(y \leftarrow t\sigma)\psi$ is an $\mathcal{A}$-substitution with the same domain. Then, since σ is an $\mathcal{A}$-solution of $\forall \vec{y} : y \neq t \vee R$, $\sigma(y \leftarrow t\sigma)\psi$ validates $y \neq t \vee R$ and since $\sigma(y \leftarrow t\sigma)\psi$ cannot validate $y \neq t$, it validates R. Hence, since $\sigma(y \leftarrow t\sigma)\psi \equiv (y \leftarrow t)\sigma\psi$, $\sigma\psi$ validates $R(y \leftarrow t)$.

B Proof of Proposition 2

Let us recall the statements of proposition 2 and lemma 1:

proposition 2
The rules of figure 2, except the rule RT, are $\mathcal{A}$-sound and $\mathcal{A}$-preserving in any subalgebra $\mathcal{A}$ of $T(F,X)$.
The rules of figure 2, except the O rules are $RT(F)$-sound and $RT(F)$-preserving.

Lemma 1
Let $\mathcal{P}$ be a conjunction of non-trivial disequations. Let A be a subset of $T(F)$ (resp. $A = RT(F)$) such that the carrier in A of any variable occurring in $\mathcal{P}$ is infinite. Then $\mathcal{P}$ has at least a solution in A. .

We first prove lemma 1 which is necessary for proving the other results. Actually, the case of rational trees does not need a special proof, provided that $T(F)$ contains infinitely many trees of sort $\underline{s}$ whenever $RT(F)$ does. This is stated by the following lemmas:

Lemma 2 *It $t \in RT(F)$ and t', a strict subtree of t (i.e. $t' \equiv t/p$ with $t \neq \varepsilon$), have the same sort, then t is sort-unrestricted.*

Lemma 3
If $RT(F)$ contains infinitely many elements of sort $\underline{s}$, then the same property holds for $T(F)$.

B.1 Proof of Lemma 2

Assume that $p \in Pos(t)$, $p \neq \varepsilon$ and $sort(t/p) = \underline{s} = sort(t)$. Let t_0 be the term obtained by replacing all the subtrees of depth $|p|$ in t by distinct variables (with corresponding sorts). Let finally σ be a ground substitution whose domain is $Var(t_0)$. (Such a substitution does exist since $T(F)$ is supposed to contain at least one term of each sort). Then, we construct by induction the following sequence t_n of ground terms: $t_{n+1} = (t_0[p \leftarrow t_n])\sigma$. Terms t_n belong to $T(F)$ and have the sort $\underline{s}$. Moreover, they are distinct since they have distinct depths. Therefore the carrier of $\underline{s}$ in $T(F)$ is infinite.□

B.2 Proof of Lemma 3

We are going to prove that all $t \in RT(F) - T(F)$ are sort-unrestricted. Obviously, this will prove the lemma.

Let $t \in RT(F) - T(F)$. Let i_m, $m \geq 1$ be an infinite sequence of integers such that, for every m, $p_m = i_1 \cdot i_2 \cdot \ldots \cdot i_m$ is a position of t. Since S is finite, there exists two indices m_1 and m_2 such that $sort(t/p_{m_1}) = sort(t/p_{m_2})$. This proves (by lemma 2) that $\underline{s} = sort(t/p_{m_1})$ has an infinite carrier in $T(F)$. Now, let t_0 be the term obtained by replacing in t every subtree of depth $1 + |p_{m_1}|$ by a variable (with an appropriate sort). Let finally σ be a ground substitution whose domain is $Var(t_0)$. we get infinitely many ground terms having sort $sort(t)$ by replacing in $t_0\sigma$ the subterm at position p_{m_1} by a term of sort $\underline{s}$. □

B.3 Proof of Lemma 1

Note that lemma 1 refers to a problem $\mathcal{P}$ without parameter. Actually, the lemma is a consequence of the fact that an equation which contains only one variable has only finitely many solutions.[8] Of course, such a property does hold in the case of $T(F,X)$, $RT(F)$ and $T(F)$.

The case of rational trees follows from the finite tree case, because of lemma 3. Indeed, it is sufficient to take $A = T(F)$. Then a solution in A will be a solution in $RT(F)$.

Now, we prove the result by induction on the number of distinct variables in $\mathcal{P}$.

- Assume that $\mathcal{P}$ does not contain any variable. Then every $\mathcal{A}$-substitution is a solution since $\mathcal{P}$ does not contain trivial disequations.

- Assume now that the property holds for problems with less than $m-1$ variables $(m \geq 1)$. Let $\mathcal{P}$ be a conjunction of disequations and $|Var(\mathcal{P})| = m$. Let $x \in Var(\mathcal{P})$. For each disequation $s \neq t$ in $\mathcal{P}$, the equation $s = t$ has at most one solution when the variables distinct from x are considered as constants[9]. Let $\mathcal{S}$ be the set of solutions of these equations. Since there are infinitely many terms in A which have the same sort than x, there exists a term $a \in A$ such that $(x \leftarrow a) \notin \mathcal{S}$.

  Now, we can use the induction hypothesis on $\mathcal{P}(x \leftarrow a)$. Indeed, for each disequation $s \neq t$ in $\mathcal{P}$, $s(x \leftarrow a) \not\equiv t(x \leftarrow a)$ by construction. If σ is a solution in A of $\mathcal{P}\sigma$, then $\sigma(x \leftarrow a)$ is a solution in A of $\mathcal{P}$.

[8] This is the basis of the extension of the results to equational theories (see Comon 1988).

[9] Here, a solution is a substitution $(x \leftarrow t)$ where t is in $T(F, Var(\mathcal{P}) - \{x\})$. For example, the equation $f(z,x) = f(0,x)$ has no solution

□

Let $RT(F,X)$ be the algebra of rational trees obtained by substituting trees of $RT(F)$ to some variables (possibly none) of a term in $T(F,X)$. In other words, $RT(F,X)$ is the set of infinite trees, obtained by replacing finitely many occurrences of constants by variables in a tree of $RT(F)$.

Lemma 1 can be extended in order to handle disequations in $RT(F,X)$. This will be useful in the following proofs. (Although we don't want to give any results concerning equational problems built on $RT(F,X)$). We shall refer to lemma 1 again for this result. As noted above, the generalization follows from the fact that an equation in $RT(F,\{x\})$ has at most one solution in $RT(F)$. This can be proved as follows: given an equation $u=v$ in $RT(F,\{x\})$ where $u\not\equiv v$, we simplify (by decomposition) the equation until either a clash is found or a non trivial equation $x=w$ is derived. This simplification steps always terminates since u and v are distinct. Then, the equation $x=w$ has at most one solution. (This is a consequence of theorem 1) property of rational trees recalled in section 2). Finally, $u=v$ has one or no solution in $RT(F)$.

B.4 Soundness and Preservation of (U_4)

Let us recall the rule (U_4).

$$(U_4) \quad \forall\vec{y},\ P\wedge(z_1=u_1\vee\ldots\vee z_n=u_n\vee R) \quad\mapsto\quad \forall\vec{y},\ P\wedge R$$

If

1. for each index i, z_i is a variable and $z_i\not\equiv u_i$.
2. for each index i, $z_i=u_i$ contains at least one occurrence of a parameter.
3. for each index i and any parameter $y\in Var(z_i,u_i)$, there is infi nitely many terms in $T(F)$ which have the same sort than y.
4. R does not contain any parameter.

The soundness of the rule is straightforward. We only prove the preservation property.

The solutions of $\forall\vec{y}:P\wedge Q$ are the solutions of both problems $\forall\vec{y}:P$ and $\forall\vec{y}:Q$. Thus, it is sufficient to prove that the set of solutions of $\forall\vec{y}:\mathcal{Q}$ equals the set of solutions of $\forall\vec{y}:R$ if $\mathcal{Q}\equiv z_1=u_1\vee\ldots\vee z_n=u_n\vee R$. Moreover, *since R does not contain any parameter,* it is sufficient to prove that $\forall\vec{y}:\mathcal{Q}$ has no solution (see the correctness and global preservation of **ED** rule for example). Finally, note that $T(F)$ is contained in any subalgebra of $T(F,X)$ as well as in $RT(F)$. Thus, it is sufficient to prove that, for any $\mathcal{A}$-substitution σ which is away from $\vec{y}$ and whose domain contains the non-parameter variables of $\mathcal{Q}$, there exists a ground substitution θ such that $\sigma\theta$ does not validate $\mathcal{Q}$.

Let σ be such an $\mathcal{A}$-substitution . $z_i\sigma\neq u_i\sigma$ is a non-trivial equation since either z_i is a parameter and $z_i\sigma\equiv z_i$ or $z_i\sigma$ does not contain variables of $Var(\mathcal{Q})$ whereas $u_i\sigma$

does contain such variables. By hypothesis on y_i we may then apply lemma 1 to $z_1\sigma \neq u_1\sigma \wedge \ldots \wedge y_n\sigma \neq u_n\sigma$.[10] Let θ be a ground solution of this system. Then $\sigma\theta$ does not validate any equation of $\mathcal{Q}$ by construction. This means that σ is not a solution of $\mathcal{Q}$. Since σ is any substitution, this means that $\mathcal{Q}$ has no solution.□

B.5 Soundness and Preservation of the rule (RT)

$$(RT) \quad \forall \vec{y}: P \wedge (d \bigvee_{i=1,\ldots,n} y_i \neq t_i \bigvee_{i=1,\ldots,m} y_{i+n} = t_{i+n}) \quad \mapsto \quad \forall \vec{y}: P \wedge d$$

If 1. d is a disjunction of equations and disequations and d does not contain any parameter,
2. $y_1, \ldots, y_n$ are distinct parameters,
3. $y_{n+1}, \ldots y_{n+m}$ are parameters,
4. for any y_i, $n+1 \leq i \leq n+m$, y_i is sort-unrestricted,
5. the sets $\{y_1 \ldots y_m\}$, $\{y_{n+1}, \ldots, y_{n+m}, t_{n+1}, \ldots, t_{n+m}\}$, $\{t_1, \ldots t_m\}$ are disjoint.

This rule is a generalization of (U_4) in the sense that we consider disjunctions of both equations and disequations. We need such a rule, otherwise disequations such as $y \neq f(y)$ cannot be removed in the algebra of rational trees. Indeed, the rule (U_2) cannot be applied in this case (nor the rule (U_5))[11].

Like in the previous proofs, we only need to prove the completeness of $\forall \vec{y}: y_1 \neq t_1 \vee \ldots \vee y_n \neq t_n \vee y_{n+1} = t_{n+1} \vee \ldots \vee y_{n+m} \neq t_{n+m} \mapsto \bot$. In other words, we have to prove that the left hand side does not have any solution in $RT(F)$. Let σ be a solution of the left hand side. We shall derive a contradiction by exhibiting a substitution on the parameters which validates $\mathcal{Q} \equiv y_1 = t_1\sigma \wedge \ldots \wedge y_n = t_n\sigma \wedge y_{n+1} \neq t_{n+1}\sigma \wedge \ldots \wedge y_{n+m} \neq t_{n+m}\sigma$. The equational part of $\mathcal{Q}$ has at least one solution θ_0 in $RT(F)$. (This is a consequence of theorem 1 and of our assuming conditions 2 and 3 for the application of the rule). Now, applying θ_0 to the disequational part of $\mathcal{Q}$, we get a problem $\mathcal{Q}_0 \equiv \; y_{n+1} \neq t_{n+1}\sigma\theta_0 \wedge \ldots \wedge y_{n+m} \neq t_{n+m}\sigma\theta_0$ on which lemma 1 can be applied: there is at least a solution θ_1 to this problem. Now, $\theta = \theta_0\theta_1$ validates $\mathcal{Q}$. □

B.6 Soundness and Preservation of the other rules

The complete proofs for the other rules are not given here. The soundness and the preservation of rules (U_3) and (CR_3)[12] follow from the properties of (U_4). More precisely, (U_3) is obtained by taking $R \equiv \bot$ in (U_4) and noticing that two terms in $T(F)$ are sufficient when we only look at one equation. In order to prove the correctness of (CR_3), we first note that it is sufficient to prove that $\exists w, w_1 \neq u_1 \wedge \ldots w_n \neq u_n \mapsto \top$ is correct (under the same restrictions). Then notice that the set of solutions of this problem is the complement of the set of solutions of its negation, on which the rule (U_4) can be applied.

[10] The variables of this problem which do not belong to $Var(\mathcal{Q})$ are considered as constants. This happens when $\mathcal{A}$ is a subalgebra of $T(F,X)$ which contains some variables of X.

[11] We forgot this rule in a previous version of the paper. We are grateful to the referee who noticed the lack of completeness

[12] The rule (CR_3) is not used in this paper.

Clashes, decompositions and occur-checks are classical.

C Proof of Proposition 3

proposition 3
The explosion rule E is $\mathcal{A}$-sound and $\mathcal{A}$-globally preserving when $\mathcal{A}$ is either $T(F)$, $RT(F)$ or any quotient of $T(F)$.
ED is $\mathcal{A}$-sound and $\mathcal{A}$-globally preserving for any $\mathcal{A}$.

About rule **E**:

$$(E) \quad \forall\vec{y}:P \quad \rightharpoonup \quad \exists w_1,\dots w_p, \forall\vec{y}:P \wedge s = f(w_1,\dots,w_p)$$

If $\{w_1,\dots,w_p\} \cap (Var(P) \cup \vec{y} \cup I) = \emptyset$ & $f \in F$ and s is a member of an equation or a disequation of P and no parameter occurs in s

It is sufficient to prove the global preservation of $\top \rightharpoonup \exists w_1,\dots w_p, \forall\vec{y}: s = f(w_1,\dots,w_p)$ since the variables $w_1,\dots,w_p$ do not occur in P. (Recall that the solutions of $\forall\vec{y}: P \wedge Q$ are the solutions of both $\forall\vec{y}: P$ and $\forall\vec{y}: Q$.) Moreover, we assumed that s does not contain any parameter, it is thus possible to remove the "$\forall\vec{y}$" in the right hand side.

Now, the global preservation is a consequence of the assumptions on $\mathcal{A}$. Indeed, let σ be any $\mathcal{A}$-solution of $\top$. Then $s\sigma \in \mathcal{A}$ can be written $f(s_1,\dots,s_p)$ for some $f \in F$ and $s_1,\dots s_n \in \mathcal{A}$[13]. Then, applying the rule with that f on the right, we get a problem $\exists w_1,\dots w_p : s = f(w_1,\dots,w_p)$. And there exists a substitution $\phi = (w_1 \leftarrow s_1,\dots w_p \leftarrow s_p)$ such that $\sigma\phi$ validates $s = f(w_1,\dots,w_p)$. This means that σ is a solution of the right hand side.

About rule **ED**:

$$(ED) \quad \forall\vec{y}:P \wedge (P_1 \vee P_2) \quad \rightharpoonup \quad \forall\vec{y}:P \wedge P_1$$

if $Var(P_1) \cap \vec{y} = \emptyset$ or $Var(P_2) \cap \vec{y} = \emptyset$

P_1 and P_2 play symmetric roles. Indeed, $\vee$ is commutative, therefore

$$\forall\vec{y}: P \wedge (P_1 \vee P_2) \rightharpoonup \forall\vec{y}: P \wedge P_2$$

is an instance of **ED**. Without loss of generality, we may assume that $\vec{y} \cap Var(P_1) = \emptyset$ and that $P \equiv \top$. We have now to prove that every solution of $\forall\vec{y}: P_1 \vee P_2$ is either a solution of P_1 or a solution of $\forall\vec{y}: P_2$. Let σ be a solution of $\forall\vec{y}: P_1 \vee P_2$ and let θ be any substitution whose domain is contained in $\vec{y}$. By definition, $\sigma\theta$ validates $P_1 \vee P_2$. Then either there exists a substitution θ such that $\sigma\theta$ validates P_1 and, in this case, σ validates P_1 since $Var(P_1) \cap \vec{y} = \emptyset$ or, for every substitution θ, $\sigma\theta$ validates P_2. In the first case, σ is a solution of P_1. In the second case, σ is a solution of $\forall\vec{y}: P_2$. □

[13] f denotes both the operator on $T(F,X)$ and the operator on $\mathcal{A}$

D Proof of Theorem 3

Let us recall the statement of theorem 3:

theorem 3
Let $\mathcal{A} = T(F)$. The non deterministic application of the rules given in figure 6, 7 and 8 to any equational problem terminates. Moreover, irreducible problems for these rules are in parameterless solved form.

D.1 Proof of Termination

Let us recall the definitions of the "interpretation functions":

- Given a disjunction of equations and disequations d, $\phi_1(d)$ is the number of distinct parameters in d.
- Given a disjunction of equations and disequations $d = e_1 \vee \ldots \vee e_n$, $\phi_2(d)$ is the multiset $\{MSP(e_1), \ldots, MSP(e_n)\}$ where $MSP(e)$ is defined by:
  - $MSP(e) = 0$ if a member of e is a solved parameter
  - Otherwise, $MSP(s = t) = MSP(s \neq t) = \max(\text{size-param}(s), \text{size-param}(t))$.
- If d is again a disjunction of equations and disequations, $\phi_3(d)$ is the number of equations and disequations whose member is a variable.
- If $\mathcal{P} \equiv \exists\vec{w}, \forall\vec{y} : \ d_1 \wedge \ldots \wedge d_n$ is a problem in conjunctive normal form, $\psi_1(\mathcal{P})$ is the multiset

$$\{(\phi_1(d_1), \phi_2(d_1), \phi_3(d_1)), \ldots, (\phi_1(d_n), \phi_2(d_n), \phi_3(d_n))\}$$

- If $\mathcal{P}$ is again an equational problem, then $\psi_2(\mathcal{P})$ is the total size of $\mathcal{P}$ (i.e. the number of operators and variable symbols in $\mathcal{P}$).

D.1.1 Termination of the set of rules when E is not considered

The array of figure 11 summarizes the variations of $\psi_1, \psi_2, , \phi_1, \phi_2, \phi_3$ by applications of the rules; at the intersection of row R and column ϕ_i appears one of the symbols $=, \leq, <$ corresponding to the variations of ϕ_i by application of the rule. For every non-obvious result, a note refers to a more detailed explanation.

(1) $(M_1), (M_3), (M_1'), (M_3')$ **strictly decrease** ψ_1**.**

This is a consequence of the control: t does not contain any parameter. Therefore, the functions ϕ_1 and ϕ_2 do not change by application of these rules. On the other hand, z is a variable and u is not a variable. Thus ϕ_3 strictly decreases for some disjunction of equations and disequations.

(2) $(M_2), (M_4)$ **do not modify** ϕ_2

This is a consequence of both control and definition of $MSP(e)$. Indeed, the only

	ψ_1	ϕ_1	ϕ_2	ϕ_3	ψ_2
$(M_1),(M_3),(M_1'),(M_3')$	$<_{(1)}$	$=_{(1)}$	$=_{(1)}$	$<_{(1)}$	
$(M_2),(M_4)$	$<$	$=$	$=_{(2)}$	$<$	
U	$<$	$<$			
D	$<$	$=$	$<_{(3)}$		
C, **T**, **O**,(EP)	$\leq$	$\leq$	$\leq$	$\leq$	$<$

Figure 11: Monotonicity of the interpretation functions

thing which is modified by application of (M_2) or (M_4) is the equation $z = u$ (resp. the disequation $z \neq u$) which is turned into $t = u$ (resp $t \neq u$). In both cases, z cannot be a solved parameter since it has at least two occurrences. Then either size-param(t) $\leq$ size-param(u) and $MSP(z = u)$ = size-param(u) = $MSP(t = u)$ (resp. $MSP(z \neq u) = MSP(t \neq u)$) or u is is a solved variable. And, in the latter case, $MSP(z = u)$ = $MSP(t = u)$ = 0 (resp. $MSP(z \neq u) = MSP(t \neq u) = 0$).

(3) The decomposition rules strictly decrease ψ_1

- Assume that $\mathcal{P} \mapsto_{D_1} \mathcal{P}'$. We may write :

  $$\psi_1(\mathcal{P}) = \{(a_1,b_1,c_1),\ldots,(a_n,b_n,c_n),(a,b,c)\}$$

  where

  $$a = \phi_1(f(t_1,\ldots,t_m) = f(u_1,\ldots,u_m)),$$

  $$b = \{\text{max(size-param(}f(t_1,\ldots,t_m)\text{)),size-param(}f(u_1,\ldots,u_m))\}$$

  and c does not matter. We may write :

  $$\psi_1(\mathcal{P}') = \{(a_1,b_1,c_1),\ldots,(a_n,b_n,c_n),(a_1',b_1',c_1'),\ldots,(a_m',b_m',c_m')\}$$

  where $a_i' =$ max(size-param(t_i),size-param(u_i)). Now, for each index i, $a_i' \leq a$ and $b_i' < b$, since, as precised in the control, $f(t_1,\ldots,t_m) = f(u_1,\ldots u_m)$ contains at least one occurrence of a parameter. This means $\{(a,b,c)\} > \{(a_1',b_1',c_1')\ldots,(a_m',b_m',c_m')\}$. Therefore ψ_1 is strictly decreasing.

- Assume that $\mathcal{P} \mapsto_{D_2} \mathcal{P}'$. We can write:

$$\psi_1(\mathcal{P}) = \{d_1,\ldots,d_n,(a,\{b_1,\ldots,b_k,MSP[f(t_1,\ldots,t_m) \neq f(u_1,\ldots,u_m)]\},c)\}$$

  and

$$\psi(\mathcal{P}') = \{d_1,\ldots,d_n,(a,\{b_1,\ldots,b_k,MSP(t_1 \neq u_1),\ldots,MSP(t_m \neq u_m)\},c')\}$$

Then each $MSP(t_i \neq u_i)$ is strictly smaller than

$$MSP[f(t_1,\ldots,t_m) \neq f(u_1,\ldots,u_m)]$$

since either $f(t_1,\ldots,t_m)$ or $f(u_1,\ldots,u_m)$ has to contain an occurrence of a parameter (because of the control). This proves that ψ_1 is again strictly decreasing.

- Assume that $\mathcal{P} \mapsto_{D_3} \mathcal{P}'$. We can write:

$$\psi_1(\mathcal{P}) = \{d_1,\ldots,d_n,(a,\{b_1,\ldots,b_k,MSP[f(t_1,\ldots,t_m) = f(u_1,\ldots,u_m)]\},c)\}$$

and

$$\begin{aligned}\psi_1(\mathcal{P}') \quad = \quad & \{d_1,\ldots,d_n,(a_1,\{b_1,\ldots,b_k,MSP(t_1 = u_1)\},c'),\ldots, \\ & (a_m,\{b_1,\ldots,b_k,MSP(t_m = u_m)\})\}\end{aligned}$$

Moreover, for each index i, $a_i \leq a$ and

$$MSP(t_i = u_i) < MSP[f(t_1,\ldots,t_n) = f(u_1,\ldots,u_n)]$$

because of the control. This proves again that ψ_1 strictly decreases.

D.1.2 Handling the Rule E

Assume that $\mathcal{P} \mapsto_E \mathcal{P}'$ and that $\psi_1(\mathcal{P}) = \{d_1,\ldots d_n\}$. Then

$$\psi_1(\mathcal{P}') = \{d_1,\ldots d_n,(0,\{0\},1)\}.$$

We want to prove that, if $\mathcal{P}' \mapsto \mathcal{P}_1$, then $\Phi(\mathcal{P}_1) < \Phi(\mathcal{P})$.

Because of the control, the rules **C**, **T**, **O**, **D**, **U**, (EP) do not apply on $\mathcal{P}$. Thus, they cannot be applied to $\mathcal{P}'$. On the other hand, **E** cannot be applied on $\mathcal{P}_1$ since merging are applied before **E** and x is supposed to occur as a member of an equation $x = u$ or a disequation $x \neq u$ of $\mathcal{P}$, where u contains at least one occurrence of a parameter. Moreover x is not a parameter. Therefore, every transformation $\mathcal{P}' \mapsto \mathcal{P}_1$ uses a merging rule between $x = f(w_1,\ldots,w_p)$ and $x = v$ (or $x \neq v$) where v does contain at least one occurrence of a parameter. (See the control on the merging rules).

This means that $\psi_1(\mathcal{P}_1) = \{d_1,\ldots,d_{n-1},d',(0,\{0\},1)\}$, where $d_n = (a_1,a_2,a_3)$, $d' = (a_1,a_2,a_3-1)$ and $a_1 \geq 1$. Now, $d_n > d'$ and $d_n > (0,\{0\},1)$, therefore, $\psi_1(\mathcal{P}) > \psi_1(\mathcal{P}_1)$.

Now, suppose that there exists an infinite transformation chain $\mathcal{P} \mapsto \ldots \mapsto \mathcal{P}_n \mapsto \ldots$. Then, we could extract an infinite chain on which Φ is strictly decreasing. This is absurd. □

D.2 Proof of Completeness

We have to prove that any problem which is irreducible for the rules of figures 6, 7 and 8 is in parameterless solved form, or, equivalently, that any problem which contains an occurrence of a parameter is reducible by one of these rules. Thus, we investigate all the possible cases of an occurrence of a parameter.

A parameter occurs in an equation or a disequation between two non-variable terms
In this case, a decomposition rule, a clash rule or a **T** rule can be applied.

A parameter occurs in an equation or a disequation between a non-variable term and an unknown
If no other rule can be applied, then the **E** rule applies.

A parameter y occurs as a member of a disequation
Then one rule among (U_1), (U_2), (T_2), (O_2) applies

Other cases of an occurrence of a parameter in an equation
The parameter has to be a member of an equation, otherwise we are in one of the first two cases above. Then, either one among the rules (T_1), (O_1), (U_5), (U_3) can be applied or, assuming that no other rule applies, we fall into the scope of (U_4).

A parameter occurs in the head of the problem.
Because of the four previous cases, it is possible to assume that there is no equation nor disequation in $\mathcal{P}$ containing a parameter. Thus **EP** may apply.

□

Some comments

- The merging rules are not used in the completeness proof. Thus the completeness still holds when these rules are not considered. However, since the termination still holds when dealing with such rules, they may also be considered for improving the efficiency.
- Occur checks are used in the completeness proof. Therefore, the proof do not apply to rational trees. However, it is then possible to use mergings. Together with the **RT** rule this provides a completeness result in rational trees, at least when the starting problems do not contain equations in the disjunctions. (The proof is left to the reader).

E Proof of Theorem 4

theorem 4
Let $\mathcal{A} \subseteq T(F,X)$. The non deterministic application of the rules of figure 9 and figure 10 to any parameterless problem terminates. Moreover, irreducible problems for these rules are definitions with constraints.

E.1 Proof of Termination

Let Φ be a function decreasing for any application of the rules and whose codomain is a well ordered set. Let us introduce some concepts which are necessary for the expression of this function. $\mathcal{P} \equiv d_1 \wedge \ldots \wedge d_n$ is a problem in conjunctive normal form: the d_i's are disjunctions of one or more equations and disequations. If d_i is $x = t$ where x is a variable and $x \not\equiv t$, x is called an *almost solved variable of* $\mathcal{P}$[14]. A *solved variable of* $\mathcal{P}$ is an almost solved variable of $\mathcal{P}$ which occurs only once in $\mathcal{P}$.

Let $\Phi = (\phi_1, \phi_2, \phi_3, \phi_4)$ where:

$\phi_1(\mathcal{P})$ is the number of variables of $\mathcal{P}$ which are not almost solved,

$\phi_2(\mathcal{P})$ is te,he number of unsolved variables of P

$\phi_3(d_1 \wedge \ldots \wedge d_n)$ where $d_1, \ldots, d_n$ are disjunctions of (one or more) equations and disequations is the multiset $\{M(d_1), \ldots, M(d_n)\}$ where $M(d)$ is the multiset of numbers $MS(e)$, for each equation and each disequation in d. $MS(e)$ is equal to 0 if e contains a solved variable; otherwise it is equal to the maximal size of its two members,

$\phi_4(\mathcal{P})$ is the number of occurrences of a variable as a member of an equation or a disequation.

Now, we summarize the variations of the functions ϕ_i in an array.

	ϕ_1	ϕ_2	ϕ_3	ϕ_4
R	$=_{(1)}$	$<_{(2)}$		
M	$=_{(1)}$	$\leq_{(3)}$	$\leq_{(4)}$	$<$
T, **C**, **O**, **ED**	$\leq_{(1)}$	$\leq_{(5)}$	$<_{(5)}$	
D	$\leq_{(1)}$	$\leq_{(6)}$	$<_{(7)}$	
E	$<_{(8)}$			

(1) ϕ_1 is never strictly increasing. Let $x = t$ be an equation of $\mathcal{P}$ which is not inside a disjunction and such that $x \not\equiv t$. Removing or impairing this equation can only be done by (M_1) or (R_1) or as a side effect of (T_2) or (O_1) resulting in the problem $\perp$ and, obviously, ϕ_1 decreases. It remains to look at three cases:

1. (R_1) or (M_1) transforms the equation $x = t$ into an equation $t = u$. Such a transformation requires that there is in $\mathcal{P}$ another equation $x = u$. Thus, after the application of the rules, x is still almost solved.
2. (R_1) or (M_1) transforms $x = t$ in $x = x$.
   In this case, there must be another occurrence of $x = t$. Thus, x remains almost solved.

[14] This must not be confused with the notion of "almost solved parameter" introduced in the previous proof.

3. (R_1) or (M_1) transforms the equation $x = t$ in an equation $x = u$ which is already in $\mathcal{P}$.
   Either t is not a variable or t occurs as a member of another equation of $\mathcal{P}$. In both cases, the number of almost solved variables is not changed.

In all other cases, ϕ_1 is trivially decreasing.

(2) ϕ_2 is strictly decreased by application of R. Because of the control imposed to the rule (R_1), z is not a solved variable of the problem to which the rule is applied (since z must occur in P). On the other hand it is a solved variable after application of (R_1) since t does not contain any occurrence of z.

(3) Merging rules do not increase ϕ_2. These rules do not introduce new variables nor duplicate a variable which did occur only once. Indeed, t cannot be a variable.

(4) Mergings do not increase ϕ_3. This is a consequence of the control:

- either $size(t) \leq size(u)$ and, by definition, ϕ_3 is unchanged by application of the rule
- or u is a solved variable. In this last case,

  $$MS(z = u) = MS(t = u) = MS(z \neq u) = MS(t \neq u) = 0.$$

  ϕ_3 is thus unchanged.

(5) T, C, O, ED decrease ϕ_3. This is obvious since $\mathcal{P}$ is supposed to be in conjunctive normal form.

(6) Decompositions do not increase ϕ_2. Indeed, the rule (D_3) only duplicates unsolved variables.

(7) ϕ_3 is strictly decreased by application of a rule in D. Let $n = MS(e)$ where e is the equation (or disequation) to which the decomposition is applied. Of course, $n > 0$.

- If $\mathcal{P} \mapsto_{D_1} \mathcal{P}'$, then

  $$\phi_3(\mathcal{P}) = \{\{n\}, a_1, \ldots, a_k\}$$

  and

  $$\phi_3(\mathcal{P}') = \{\{n_1\}, \ldots, \{n_m\}, a_1, \ldots, a_k\}$$

  where $n_i < n$. Thus, $\phi_3(\mathcal{P}') < \phi_3(\mathcal{P})$.

- If $\mathcal{P} \mapsto_{D_2} \mathcal{P}'$, then

  $$\phi_3(\mathcal{P}) = \{\{n, b_1, \ldots, b_l\}, a_1, \ldots, a_k\}$$

and

$$\phi_3(\mathcal{P}') = \{\{n_1,\ldots,n_m,b_1,\ldots,b_l\},a_1,\ldots,a_k\}$$

where $n_i < n$. Thus, $\phi_3(\mathcal{P}') < \phi_3(\mathcal{P})$.

- If $\mathcal{P} \mapsto_{D_3} \mathcal{P}'$, then

$$\phi_3(\mathcal{P}) = \{\{n,b_1,\ldots,b_l\},a_1,\ldots,a_k\}$$

and

$$\phi_3(\mathcal{P}') = \{\{n_1,b_1,\ldots,b_l\},\ldots,\{n_m,b_1,\ldots,b_l\},a_1,\ldots,a_k\}$$

where $n_i < n$. Thus, $\phi_3(\mathcal{P}') < \phi_3(\mathcal{P})$.

(8) ϕ_1 **is strictly decreasing by application of the explosion.** Indeed, if it is possible to explode x, x is a member of a disequation $x \neq u$. Moreover, **R** and **C** cannot be applied. Thus, x is not a member of an equation of the problem. Therefore x is not almost solved. Since the explosion rule adds an equation $x = u$ where u does not contain any variable, it implies decreasing ϕ_1.

Since the lexicographic composition and the multiset extensions of well founded orderings are also well-founded, there does not exist an infinite sequence of problems $\mathcal{P}_i$ such that $\Phi(\mathcal{P}_i)$ is strictly decreasing. Therefore, the non deterministic application of the rules of figures 9 and 10 terminates. □

E.2 Completeness Proof

We show that every parameterless problem which is not a definition with constraints can be reduced by the rules of figures 9 and 10. Let $\mathcal{P}$ be such a problem.
If $\mathcal{P}$ contains disjunctions. Then we may apply **ED**.

If $\mathcal{P}$ contains an equation or a disequation whose members are not variables. Then it is possible to apply one of the rules **D**, **C**, **T**.

If a variable x of an equation $x = t$ of $\mathcal{P}$ occurs twice in $\mathcal{P}$, assume moreover that, if t is a variable, then t occurs also twice in $\mathcal{P}$. Then either $x \in Var(t)$ and it is possible to apply (O_1) or $x \notin Var(t)$ and it is possible to apply (R_1).

If a variable x of a disequation $x \neq t$ is sort-restricted
Then, if no other rule can apply, the explosion rule may be used.[15]

If there is a disequation $x \neq x$, then rule (T_2) can apply.

□

[15] Actually, such a case does not occur when $\mathcal{A} = T(F,X)$ since we assumed that X contains infinitely many variables of each sort.

F Proof of Proposition 5

proposition 5
Let $\mathcal{A}$ be either $T(F)$ or $RT(F)$. A problem in *definition with constraints* **has at least one solution if and only if it is syntactically different from $\bot$.**

Let $\mathcal{P} \equiv \exists z_1, \ldots, z_k : x_1 = t_1 \wedge \ldots \wedge x_n = t_n \wedge x'_1 \neq u_1 \wedge \ldots \wedge x'_m \neq u_m$ be a problem in definition with constraints.

It is sufficient to prove that the problem obtained by binding every free variable in $\mathcal{P}$ with an existential quantifier can be reduced to $\top$. Therefore, we may assume, without loosing generality that $\mathcal{P}$ does not contain free variables.

Again, it is sufficient to prove that the problem obtained by taking the negation of $\mathcal{P}$ can be reduced to $\bot$. In the following, $\mathcal{P}'$ will denote the problem obtained in this way :

$$\mathcal{P}' \equiv \forall z_1, \ldots, z_k : x_1 \neq t_1 \vee \ldots x_m \neq t_m \vee x'1 = u_1 \vee \ldots \vee x'_n = u_n$$

Each variable in $\mathcal{P}'$ is assumed to be a parameter.

F.1 The Case of Finite Trees

Because of the first property of definition with constraints (definition 10), it is possible to apply rule (U_2) to $\mathcal{P}'$. We then get a problem

$$\mathcal{P}'' \equiv \forall z_1, \ldots z_k : x'_1 = u_1 \vee \ldots \vee x'_n = u_n$$

Because of property 2 in definition 10, it is now possible to apply rule (U_4) which leads to $\bot$.

F.2 The Case of Infinite Trees

It is sufficient to see that is possible to apply the rule (RT) to $\mathcal{P}'$. Conditions 3 and 4 in the (RT) rule indeed correspond to point 3 in definition 11. Condition 2 corresponds to point 1 in definition 11. Finally, properties 2 and 4 in definition 11 insure condition 5.

G Rational Trees

The completeness and termination results (proposition 4 and theorems 2, 3 and 4) can be extended to the case of rational trees. However, this needs some more rules. Here, we only sketch very briefly this extension. (This is not our aim in this paper).

G.1 Parameterless Solved Forms

First, as noticed in appendix D, the rules of figures 6, 7, and 8 are still correct and terminating when $\mathcal{A} = RT(F)$, if we remove the **O** rule. Moreover, adding the **RT** rule does not impair the termination, since it obviously strictly decreases ψ_1. Unfortunately, this is not sufficient for insuring the completeness w.r.t. parameterless solved forms. Indeed, a problem such as

$$\mathcal{P} \equiv \forall y : y = f(y) \vee y \neq f(f(y))$$

cannot be transformed. Indeed, the **RT** rule requires that left hand sides of equations and disequations in a disjunction be disjoint and the (M_4) rule requires the inequality size-param(t) $\leq$ size-param(u), which does not hold here. Moreover it is not possible to remove any of these two requirements. For example, allowing the merging (M_4) in the above problem, leads to a problem

$$\forall y: \; f(f(y)) = f(y) \vee y \neq f(f(y))$$

which is transformed back into $\mathcal{P}$ by decomposition. Thus, we would loose termination.

If we only use the rules given in section 3, the completeness only holds when starting from ELD-problems.

Nevertheless, it is possible to handle the general case, using a method similar to the one given in Colmerauer (1984). Let us give an outlook of this method:
Assume that $d \equiv d_1 \vee d_2$ where d_1 is a disjunction of equations and d_2 is a disjunction of disequations. Assume that no rule can be applied. Let $y = t$ be an equation of d_1 and

$$d_3 \equiv d_2 \vee y \neq t \equiv y_1 \neq t_1 \vee \ldots \vee y_n \neq t_n \vee y \neq t$$

.

Now, using **D** and **M**, we reduce d_3 into an irreducible disjunction of disequations d_4. Three cases are possible:

1. $d_4 \equiv \top$.
2. $d_4 \equiv y_1 \neq u_1 \vee \ldots \vee y_n \neq u_n$
3. $d_4 \equiv y_1 \neq u_1 \vee \ldots \vee y_n \neq u_n \vee y_{n+1} \neq u_{n+1} \vee \ldots \vee y_{n+m} \neq u_{n+m}$ where $m \geq 1$.

In the first case, $\mathcal{S}(d, RT(F)) = \mathcal{S}(d_1 \vee d_2', RT(F))$ where d_2' is obtained by removing the equation $y = t$ from d_2).

In the second case, $\mathcal{S}(d, RT(F)) = \mathcal{S}(\top, RT(F))$.

In the third case, we can replace $y = t$ in d by $y_{n+1} = u_{n+1} \vee \ldots \vee y_{n+m} = u_{n+m}$ without modifying the set of solutions in $RT(F)$.

Each of these three cases leads to a transformation rule which is sound and preserving in $RT(F)$. We don't prove the correctness of such transformations. Essentially, it is the same transformation as in Colmerauer's paper (see also Lassez, Maher & Marriott (1986) and Maher (1988)). It must be noted that such transformations do terminate since the parameters occurring as a member in a disequation of d do not occur as a member of an equation in $y_{n+1} = u_{n+1} \vee \ldots \vee y_{n+m} = u_{n+m}$. Therefore, the number of parameters which occur both as a member of an equation and as a member of a disequation in d is strictly decreasing. When there is no such shared variables, then it is possible to apply the (RT) rule. Consequently, it is now possible to eliminate the parameters.

G.2 Definition with Constraints

Exactly the same problem occurs when we try to reach definition with constraints in rational trees. (Take the negation of the above example). Again a method similar to Colmerauer's algorithm does work.