Technical Report

UCAM-CL-TR-181
ISSN 1476-2986

Number 181

UNIVERSITY OF
CAMBRIDGE

Computer Laboratory

Proof transformations for equational theories

Tobias Nipkow

September 1989

15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

https://www.cl.cam.ac.uk/

Technical reports published by the University of Cambridge
Computer Laboratory are freely available via the Internet:

https://www.cl.cam.ac.uk/techreports/

ISSN 1476-2986

Proof Transformations For Equational Theories*

Tobias Nipkow
University of Cambridge
Computer Laboratory
Pembroke Street
Cambridge CB2 3QG
England
tnn@cl.cam.ac.uk

Abstract

This paper contrasts two kinds of proof systems for equational theories: the standard ones obtained by combining the axioms with the laws of equational logic, and alternative systems designed to yield decision procedures for equational problems.

Although new matching algorithms for (among other theories) associativity, associativity + commutativity, and associativity + commutativity + identity are presented, the emphasis is not so much on individual theories but on the general method of proof transformation as a tool for showing the equivalence of different proof systems.

After studying proof translations defined by rewriting systems, equivalence tests based on the notion of *resolvant* theories are used to derive new matching and in some cases unification procedures for a number of equational theories. Finally the combination of resolvant systems is investigated.

*This work was supported by the Alvey Diamond project, SERC grants GR/E/02369 and GR/F/10811. At MIT the author was supported in part by NYNEX, NSF grant CCR-8706652, and by the Advanced Research Projects Agency of the DoD, monitored by the ONR under contract N00014-83-K-0125.

1 Introduction

We contrast two kinds of proof systems for a number of equational systems E. On the one hand there is the standard one obtained by combining E with the laws of equational logic. We know that the resulting system defines what we want it to define. However, it gives no indication how the word problem, matching, or unification can be solved in that theory. On the other hand we present proof systems that are not obviously complete for E but which immediately give rise to matching or even unification procedures.

Although new matching algorithms for associativity (A), associativity + commutativity (AC), and associativity + commutativity + identity ($AC1$) are presented, the emphasis is not so much on individual theories but on the general method of proof transformations as a tool for showing the equivalence of different proof systems. In particular a mechanizable equivalence test and its implementation are discussed. Although the test is not complete, it is powerful enough to deal with many equational systems.

The paper comes in two parts. Section 3 compares standard proof systems for the empty theory, commutativity, left/right-commutativity, A, and AC, with alternative ones. Equivalence is shown by presenting a terminating set of rewrite rules which translate proofs from one system into the other. However, these rewriting systems become more and more complex to construct and to prove terminating. Therefore Section 4 uses the notion of "resolvance" to present a uniform treatment of the theories in Section 3. It is shown that resolvant systems of equations directly yield alternative inference systems which, in certain cases, are terminating matching algorithms. Two powerful criteria for resolvance are developed and their implementation is discussed. This implementation is used to check a number of further equational systems for resolvance, which yields new matching algorithms for some of them. It is also shown that resolvance is a modular property, i.e. putting resolvant sets of equations together which do not have function symbols in common does again yield resolvant systems. Finally we discuss the relationship of our work with some recent results by Claude Kirchner.

The reader should be familiar with the basic notions of equational logic, as defined for example in [6].

2 Equational Theories, Unification, Matching, and Equality

Equational theories are defined by inference rules of the form

$$\frac{s_1 = t_1 \quad \ldots \quad s_n = t_n}{s = t}. \tag{1}$$

A system of such rules inductively defines a predicate $=$. Furthermore, since all rules are Horn clauses, they constitute a logic program which can be used to solve unification, matching or word problems w.r.t. $=$. The only problem is termination. Since a set of Horn clauses merely yields a semidecision procedure, it depends on the particular set of rules whether they actually constitute a decision procedure for equality, matching

or unification. Thus we can separate the question of correctness (does the given set of rules axiomatize the desired theory?) from that of termination.

Given a set of equations E, the equational theory induced by E is defined by E plus reflexivity, symmetry, transitivity and congruence. This system is denoted by E^+ and the predicate it defines by $=_E$. Although E^+ is by definition correct, i.e. defines the equational theory generated by E, it has a major termination problem: due to transitivity, any query (in the logic programming sense) will run forever even after all answer substitutions have been found.

The rest of the paper discusses alternative axiomatizations of various simple equational theories which yield decision procedures for matching and, in some trivial cases, even unification. Initially attention is focussed on correctness. In Section 4 the termination question is addressed and a fairly general answer is given.

Before we become technical, we have to fix some terminology. An arbitrary set of equations E is often called a *presentation*, in contrast to the equational theory $=_E$, which is closed under equational reasoning. We call an equational theory *permutative* if all its equivalence classes are finite. A set of equations is permutative if its equational theory is. An equation $s = t$ is called *regular* if $\mathcal{V}(s) = \mathcal{V}(t)$, where $\mathcal{V}$ returns the set of variables in a term. An equation $s = t$ is called *collapse-free* if both s and t are proper terms. A presentation is called collapse-free if all its members are.

3 Proof Transformation by Rewriting

In the following we examine different formulations of some well known equational theories. Each alternative axiomatization D_{rk} of E^+ consists of a collection of rules of the form (1), say D for "decomposition", together with congruence rules and reflexivity. D_{rk} and E^+ are shown to be equivalent by translating proofs from one into the other. The translation is expressed by rewrite rules on proof trees.

D and E are always chosen such that each element of D is a derived rule of E^+ and each equation in E is a derived rule of D_{rk}. Thus the transformation of proofs in D_{rk} into E^+ is trivial and is briefly described in Section 4. The translation in the opposite direction needs to get rid only of symmetry and transitivity, the two rules that cause termination problems and do not occur in D_{rk}.

3.1 The Empty Theory

We start with the basic laws of equational logic, reflexivity (r), symmetry (s), transitivity (t) and congruence (k) for a single binary function symbol ($\cdot$). The choice of function symbols is immaterial.

$$r \equiv \quad \frac{}{x = x}$$

$$s \equiv \quad \frac{x = y}{y = x}$$

$$t \equiv \frac{x = y \quad y = z}{x = z}$$

$$k \equiv \frac{x = u \quad y = v}{x \cdot y = u \cdot v}$$

In the sequel this system is called B. B axiomatizes equality in the empty theory. Of course r alone suffices for that. A constructive proof of this fact can be given by a set of transformation rules which eliminate all other rules from a proof in B.

$$\frac{\dfrac{r\dfrac{}{x = x}}{s\dfrac{}{x = x}}}{} \longrightarrow r\frac{}{x = x}$$

$$\frac{t\dfrac{r\dfrac{}{x = x} \quad r\dfrac{}{x = x}}{x = x}}{} \longrightarrow r\frac{}{x = x}$$

$$\frac{k\dfrac{r\dfrac{}{x = x} \quad r\dfrac{}{y = y}}{x \cdot y = x \cdot y}}{} \longrightarrow r\frac{}{x \cdot y = x \cdot y}$$

These rules are obviously terminating and cover all cases, i.e. reduce any proof in B to reflexivity.

In the sequel E will be some set of equations and D a set of inference rules. We want to show that for given D and E, any proof in $E^+ = E \cup B$ can be rewritten to one in $D_{rk} = D \cup \{r, k\}$. This transformation can be done on a rule by rule basis.

Axioms from E can be eliminated in one step because E and D will always be chosen such that all axioms in E are equivalent to some combination of rules in D_{rk}.

To simplify elimination of s and t, we show that B has some further properties:

$$\frac{t\dfrac{x = y \quad y = z}{s\dfrac{x = z}{z = x}}}{} \longrightarrow t\frac{s\dfrac{y = z}{z = y} \quad s\dfrac{x = y}{y = x}}{z = x} \tag{2}$$

$$\frac{s\dfrac{k\dfrac{x = u \quad y = v}{x \cdot y = u \cdot v}}{u \cdot v = x \cdot y}}{} \longrightarrow k\frac{s\dfrac{x = u}{u = x} \quad s\dfrac{y = v}{v = y}}{u \cdot v = x \cdot y} \tag{3}$$

$$\frac{t\dfrac{r\dfrac{}{x = x} \quad x = z}{x = z}}{} \longrightarrow x = z \qquad t\frac{x = z \quad r\dfrac{}{z = z}}{x = z} \longrightarrow x = z \tag{4}$$

$$\frac{t\dfrac{k\dfrac{u = w \quad v = x}{u \cdot v = w \cdot x} \quad k\dfrac{w = y \quad x = z}{w \cdot x = y \cdot z}}{u \cdot v = y \cdot z}}{} \longrightarrow k\frac{t\dfrac{u = w \quad w = y}{u = y} \quad t\dfrac{v = x \quad x = z}{v = z}}{u \cdot v = y \cdot z} \tag{5}$$

Rules (2) and (3) show that symmetry can be pushed through transitivity and congruence. Therefore symmetry can be pushed to the leaves of any proof in $B \cup E$, and hence can be eliminated altogether, provided that E is closed under s, i.e. $s{=}t \in E$ implies $t{=}s \in E$. In the sequel E will always have that property, and elimination of s is automatic. This is the formal justification of the fact that equational proofs don't need explicit symmetry if all axioms can be used in both directions.

The only remaining task is the elimination of t. Rules (4) and (5) show that it is sufficient to cover the cases $t(\rho, k)$, $t(k, \rho)$, and $t(\rho, \rho')$ for all rules ρ, ρ' in D.

Before we look at particular systems D and E, we simplify our notation. Although the above rewrite rules rules are fairly involved already, they are not quite precise in that they don't say how the subtrees are relocated. The proper formulation of, for example, rule (3) is

$$\dfrac{\dfrac{\dfrac{P \quad Q}{x=u \quad y=v}\,k}{x{\cdot}y = u{\cdot}v}\,s}{u{\cdot}v = x{\cdot}y} \longrightarrow \dfrac{\dfrac{\dfrac{P}{x=u}\,s \quad \dfrac{Q}{y=v}\,s}{u=x \quad v=y}\,k}{u{\cdot}v = x{\cdot}y}$$

where P and Q are the proof trees that prove $x = u$ and $y = v$. However, this rule is more complicated than it needs to be. All that is required is the pattern of proof rules as in

$$s(k(P, Q)) \longrightarrow k(s(P), s(Q)).$$

We now assume that r, s, t, and k are functions on proof trees or skeletons. The actual formulae being proved are determined by these proof skeletons. Under this new interpretation the above rewrite rules translate to

$$\left.\begin{array}{rcl} s(r) & \longrightarrow & r \\ t(r,r) & \longrightarrow & r \\ k(r,r) & \longrightarrow & r \\ s(t(x,y)) & \longrightarrow & t(s(y),s(x)) \\ s(k(x,y)) & \longrightarrow & k(s(x),s(y)) \\ t(r,x) & \longrightarrow & x \\ t(x,r) & \longrightarrow & x \\ t(k(x,y),k(u,v)) & \longrightarrow & k(t(x,u),t(y,v)) \end{array}\right\} T$$

Notice that $x, y, z, \ldots$ stand for proof skeletons, not terms or formulae. The termination of T can now be shown by a mechanical system like LP [5].

3.2 Commutativity (C)

An alternative axiomatization of commutativity is obtained by adding

$$c \equiv \frac{x = v \quad y = u}{x{\cdot}y = u{\cdot}v}.$$

to r and k. This system is trivial and well known. For example Claude Kirchner [8] derives c automatically from commutativity. Commutativity is proved from c by

composing both premises with reflexivity, i.e. the conclusion of $c(r,r)$ is $x{\cdot}y = y{\cdot}x$. The same device works for all subsequent equational theories.

The following further rewrite rules are needed to translate proofs:

$$\begin{aligned} t(c(x,y),c(u,v)) &\longrightarrow k(t(x,v),t(y,u)) \\ t(c(x,y),k(u,v)) &\longrightarrow c(t(x,v),t(y,u)) \\ t(k(x,y),c(u,v)) &\longrightarrow c(t(x,u),t(y,v)) \end{aligned}$$

Again, LP manages to show that the union of T with the above rules is a terminating system.

It is interesting to note that both for the empty theory and for commutativity, the system D_{rk} could be derived from $B \cup E$ automatically using the CEC system [2,4] which is a general purpose system for the completion of sets of conditional equations. The following equational theories are more complex and automatic tools failed to help.

3.3 Left/Right-Commutativity ($C_{l/r}$)

The first non-trivial example is left/right-commutativity. For simplicity we consider only $E = \{(x{\cdot}y){\cdot}z = (x{\cdot}z){\cdot}y\}$, right-commutativity. Left-commutativity is symmetric.

It turns out that if $D = \{c_r\}$, where

$$c_r \equiv \frac{x = w{\cdot}v \quad w{\cdot}y = u}{x{\cdot}y = u{\cdot}v},$$

then D_{rk} axiomatizes C_r. The elimination of t is achieved by the following rules:

$$\begin{aligned} t(k(x,y),c_r(u,v)) &\longrightarrow c_r(t(x,u),t(k(r,y),v)) && (6)\\ t(c_r(x,y),k(u,v)) &\longrightarrow c_r(t(x,k(r,v)),t(y,u)) && (7)\\ t(c_r(x,k(y_1,y_2)),c_r(u,v)) &\longrightarrow t(c_r(x,r),c_r(t(k(y_1,y_2),u),v)) && (8)\\ t(c_r(x,c_r(y_1,y_2)),c_r(u,v)) &\longrightarrow t(c_r(x,r),c_r(t(c_r(y_1,y_2),u),v)) && (9)\\ t(c_r(x,r),c_r(r,v)) &\longrightarrow k(t(x,v),r) && (10)\\ t(c_r(x,r),c_r(k(u_1,u_2),v)) &\longrightarrow k(t(x,t(k(u_1,r),v)),u_2) && (11)\\ t(c_r(x,r),c_r(c_r(u_1,u_2),v)) &\longrightarrow c_r(t(x,c_r(u_1,r)),t(c_r(r,u_2),v)) && (12) \end{aligned}$$

The first two rules cover the cases that either of t's subtree is labelled with k. The next two rules translate from $t(c_r(.,.),c_r(.,.))$ to $t(c_r(.,r),c_r(x,.)$, and the last three rules translate the latter pattern, depending on the form of x.

The above set of rules is not easily proved to terminate, and systems like LP fail to do so. The point is that the termination argument is hidden in the equations that are being proved, which are not part of the proof skeletons. To exploit this information, we view t as a function defined on terms over $\{r,k,c_r\}$. Looking at the form of the transformation rules we can see that their termination, or equivalently totality of t, can be shown by proving that some measure function on the arguments of t decreases when going from left to right. This measure is the size of the equation (or either side of it) that forms the invisible conclusion of the proof tree rooted in t. One way to see that

the measure decreases is to decorate the rewrite rules with the equations being proved. For example rule (10) becomes

$$t\,\dfrac{c_r\,\dfrac{x = u\cdot v \quad r\,\dfrac{}{u\cdot y = u\cdot y}}{x\cdot y = (u\cdot y)\cdot v} \quad c_r\,\dfrac{r\,\dfrac{}{u\cdot y = u\cdot y} \quad u\cdot v = z}{(u\cdot y)\cdot v = z\cdot y}}{x\cdot y = z\cdot y} \longrightarrow k\,\dfrac{t\,\dfrac{x = u\cdot v \quad u\cdot v = z}{x = z} \quad r\,\dfrac{}{y = y}}{x\cdot y = z\cdot y}$$

On the rhs t proves $x = z$, which is strictly smaller than $x\cdot y = z\cdot y$, the conclusion of the tree labelled by t on the lhs.

Alternatively, we notice that the size of the conclusion of k and c_r is strictly greater then the size of their hypotheses. Hence in any rule with lhs $t(\ldots k(x,y)\ldots)$ or $t(\ldots c_r(x,y)\ldots)$, the measure of x and y is smaller than the measure of the full lhs. Hence x and y are ok as arguments to t on the rhs.

If the termination proof is done in complete detail, one notices that in rules (8) and (9) the outer occurrence of t on the rhs proves the same formula as the t on the lhs, i.e. the measure is not decreased. However, none of the two rules can be applied twice in a row. Therefore the complexity measure becomes a pair. Its first component is what we had before, which all other rules decrease. The second component indicates the applicability of rule (8) or (9). It is decreased by those two rules, which do not change the first component.

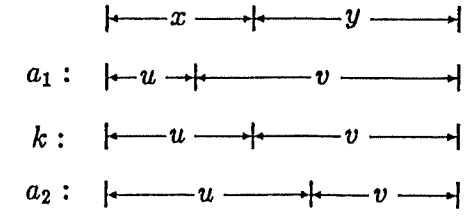
As we have seen, both the transformation rules and their termination proofs become quite complex even for very simple equational theories. For that reason, both are omitted in the following examples. I have carried them out by hand and found them very similar to what we saw in this section, only more tedious. This prompted me to look for automatic methods which are presented in Section 4.

3.4 Associativity (A)

Associativity can be axiomatized with k, r,

$$a_1 \equiv \frac{x = u\cdot w \quad w\cdot y = v}{x\cdot y = u\cdot v} \quad \text{and} \quad a_2 \equiv \frac{x\cdot w = u \quad y = w\cdot v}{x\cdot y = u\cdot v}.$$

A geometric interpretation of this fact can be given where terms are interpreted as strings or lines and $\cdot$ is concatenation. If the two lines $x\cdot y$ and $u\cdot v$ are equal, there are three cases:



It can be shown that Plotkin's associative unification procedure [13] can be derived from the procedure embodied in the inference rules k, r, a_1 and a_2 by imposing a particular search strategy.

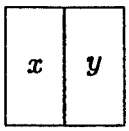
3.5 Associativity + Commutativity (AC)

For AC let $D = \{c, a_1, a_2, ac_1, ac_2, ac\}$, where

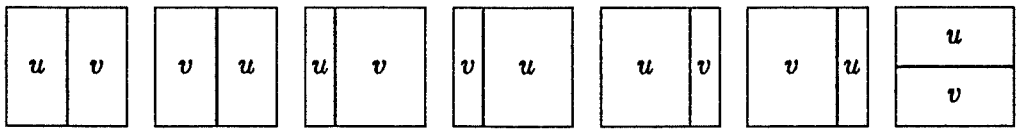
$$ac_1 \equiv \frac{x = v \cdot w \quad w \cdot y = u}{x \cdot y = u \cdot v} \qquad ac_2 \equiv \frac{x \cdot w = v \quad y = w \cdot u}{x \cdot y = u \cdot v}$$

$$ac \equiv \frac{x = x_1 \cdot x_2 \quad y = y_1 \cdot y_2 \quad x_1 \cdot y_1 = u \quad x_2 \cdot y_2 = v}{x \cdot y = u \cdot v}.$$

Again there is a geometric interpretation of these rules, this time in terms of areas or multisets. If $x \cdot y$ is of the form



and $u \cdot v$ is the same area, there are 7 ways in which they can cover each other:



k c a_1 ac_1 a_2 ac_2 ac

4 Resolvant Theories

The collection of equational theories presented above and the complexity of some of the completeness proofs raises the question whether there is some principle behind them all which might even be automated. For some of the simpler examples (C, $C_{l/r}$) this question was first answered by Claude Kirchner [7,8] using the notion of *resolvant* presentation.

A set of equational axioms E is *resolvant* if $s =_E t$ implies that there is an equational derivation of this fact which uses at most one application of an equation at the root of a term. Notice that by a *derivation* $s =_E t$ we refer to a list of terms $s = s_0, \ldots, s_n = t$ such that s_{i+1} can be obtained from s_i by replacing a subterm which is an instance of one side of an equation in E by the corresponding instance of the other side. In particular we call the step $s_k =_E s_{k+1}$ a *peak* if s_k is an instance of one side of an equation in E and s_{k+1} the corresponding instance of the other side. With this terminology we can say that E is resolvant if $s =_E t$ implies that there is an equational derivation of this fact with at most one peak. For a formal definition of resolvance we need the following simple predicates:

$$\begin{aligned}
s \mathrel{\dot{=}}_E t &\Leftrightarrow \exists f, s_i, t_i.\ s = f(s_1, \ldots, s_n) \wedge t = f(t_1, \ldots, t_n) \wedge s_1 =_E t_1 \wedge \ldots \wedge s_n =_E t_n \\
s \mathrel{\dot{\dot{=}}}_E t &\Leftrightarrow \exists p{=}q \in E, \sigma.\ s = \sigma p \wedge t = \sigma q \\
s \mathrel{\dot{\dot{=}}}_E t &\Leftrightarrow \exists s', t'.\ s \mathrel{\dot{=}}_E s' \mathrel{\dot{=}}_E t' \mathrel{\dot{=}}_E t \\
s \mathrel{\dot{\dot{=}}}_E t &\Leftrightarrow s \mathrel{\dot{=}}_E t \vee s \mathrel{\dot{\dot{=}}}_E t
\end{aligned}$$

Notice that in general only $\doteq_E$ is decidable, provided E is finite. If E is also permutative, all four predicates are in principle decidable, although in practice a complexity theoretic barrier may quickly be reached.

Definition 1 *E is called* resolvant *iff $s =_E t$ implies $s \doteq_E t$.*

An equational theory is called *syntactic* by Kirchner [8] if it is generated by a finite set of resolvant axioms. In the sequel we tacitly restrict ourselves to collapse-free equational presentations. The theory can be made to work without that restriction but that requires some further case distinctions.

The point is that each equational theory presented above is syntactic but only C and $C_{l/r}$ are resolvant. Although [8] gives some sufficient conditions for syntacticness, they are not met by $C_{l/r}$, A or AC. The aim of this section is to present a more refined criterion for resolvance, its implementation, and its application to both the examples above and some further theories.

For resolvant presentations there is a simple translation from equational axioms to inference rules: an equation $f(s_1,\ldots,s_m) = g(t_1,\ldots,t_n)$ yields

$$\frac{x_1 = s_1 \;\ldots\; x_m = s_m \quad t_1 = y_1 \;\ldots\; t_n = y_n}{f(x_1,\ldots,x_m) = g(y_1,\ldots,y_n)} \tag{13}$$

where the x_i and y_j are new variables. In [8] this translation is called *Gen*. All the inference rules presented above can be generated this way. However, in many cases this leads to rules with trivial equations of the form $x = y$ among the hypotheses. These can be deleted without loss of generality if x is replaced by y everywhere else.

Starting with C, $C_{l/r}$, or A we obtain the rules shown in Sections 3.2 to 3.4. For AC we needed three more rules. Those are generated by the equations

$$P = \{(x{\cdot}y){\cdot}z = (x{\cdot}z){\cdot}y,\ x{\cdot}(y{\cdot}z) = y{\cdot}(x{\cdot}z),\ (x{\cdot}y){\cdot}(u{\cdot}v) = (x{\cdot}u){\cdot}(y{\cdot}v)\}$$

The reason is that AC by itself is not resolvant but $AC^+ = AC \cup P$ is. Note that all elements of P are equational consequences of AC.

In the sequel let D be the set of inference rules obtained from E as above, let K be the set of all congruence rules, and let $D_{rk} = D \cup \{r\} \cup K$. In particular let k_f be the congruence rule for function symbol f.

We can now give a general scheme for translating proofs in D_{rk} to those in E^+, the direction that had not been tackled in Section 3. For every rule $\rho \in D$, $\rho(r,\ldots,r)$ is the proof of some equation $e \in E$. If ρ is rule (13), we have the following translation from D_{rk} to E^+:

$$\rho(p_1,\ldots,p_m,q_1,\ldots,q_n) \longrightarrow t(k_f(p_1,\ldots,p_m),t(e,k_g(q_1,\ldots,q_n))) \tag{14}$$

The importance of resolvant presentations stems from the following theorem.

Theorem 1 *If E is resolvant, D_{rk} is a sound and complete inference system for $=_E$.*

Proof Soundness of D_{rk} follows from the fact that each rule in D is a derived rule in E^+, as witnessed by (14). The completeness proceeds by induction on the length of equational proofs. Let $s =_E t$ and distinguish 3 cases.

If $s = t$, this has an immediate proof by reflexivity.

If $s \mathrel{\dot{=}}_E t$, it follows that $s = f(s_1, \ldots, s_n)$, $t = f(t_1, \ldots, t_n)$, and $s_i =_E t_i$ for all i. Since the derivation of the latter equations are shorter than the derivation of $s =_E t$, there is a proof skeleton p_i in D_{rk} for each of them. Hence there is a proof of $s =_E t$ with the skeleton $k_f(p_1, \ldots, p_n)$.

If $s \mathrel{\dot{\dot{=}}}_E t$, it follows that there is an equation $f(s_1, \ldots, s_m) = g(t_1, \ldots, t_n)$ in E and

$$s = f(r_1, \ldots, r_m) \mathrel{\dot{=}}_E f(s'_1, \ldots, s'_m) \mathrel{\dot{\dot{=}}}_E g(t'_1, \ldots, t'_n) \mathrel{\dot{=}}_E g(u_1, \ldots, u_n) = t$$

such that s'_i and t'_j are instances of s_i and t_j respectively and that $s_i =_E s'_i$ and $t'_j =_E t_j$ for all i and j. Since the derivation of the latter equations are shorter than the derivation of $s =_E t$, there are proof skeletons p_i and q_j for each of them. Hence the skeleton $\rho(p_1, \ldots, p_m, q_1, \ldots, q_n)$, where ρ is the rule (13) derived from $f(s_1, \ldots, s_m) = g(t_1, \ldots, t_n)$, proves $s =_E t$. □

In addition, we have:

Theorem 2 *If E is permutative, the interpretation of D_{rk} as a Prolog program yields a terminating matching algorithm.*

Proof An equation $l = r$ is called a *matching problem* if $\mathcal{V}(l) = \{\}$. Because E is permutative, it must be regular, and hence any solution σ to a matching problem $l = r$ must be such that $\mathcal{V}(\sigma r) = \{\}$.

Interpreting D_{rk} as a Prolog program means that the order of the goals is important and they are kept as a list (of equations). We use the ML notation for lists, i.e. $[e_1, \ldots, e_n]$ is a list of length n, and @ denotes list concatenation.

First we establish that the current goal is always a matching problem if it was one initially. Resolution of a rule in D_{rk} with a matching problem $l = r$ will always lead to a list of subgoals of the form

$$H = H_l @ H_r = [l_1 = s_1, \ldots, l_m = s_m, t_1 = r_1, \ldots, t_n = r_n] \tag{15}$$

such that each $l_i = s_i$ is a matching problem, and $\bigcup_{j=1}^{n} \mathcal{V}(t_j) \subseteq \bigcup_{i=1}^{m} \mathcal{V}(s_i)$. The first property follows from the form of the rules, the second one is a consequence of the fact that E is regular. Both properties together imply that during execution the current goal will always be a matching problem: either it was one to start with ($l_i = s_i$), or a solution of the goals to the left of it have instantiated all variables on its lhs by ground terms ($t_j = r_j$).

To prove termination, we need some more definitions. If $S \subseteq \mathrm{N}$, $max(S) = m$ if $m \in S$ and all elements in S are less or equal m, and $max(S) = 0$ if there is no such m. The function *depth* computes the depth of a term, where the depth of variables and constants is 1. The function

$$maxd_E(s) = max\{depth(t) \mid s =_E t\}$$

computes the maximal depth of any E-equivalent term. Since E is permutative, $maxd_E(s)$ is never 0. The ordering $<$ on N is extended to multisets over N in the

canonical way [3]. Multiset union is written $\sqcup$. The function C takes two lists of equations and produces a multiset of natural numbers:

$$\begin{aligned} C([], L) &= \{\} \\ C([l = r]@R, L) &= \{max\{maxd_E(\sigma l) \mid \sigma \vdash_E L\}\} \sqcup C(R, L@[l = r])\} \end{aligned}$$

where $\sigma \vdash_E L$ means that σ solves all equations $s = t$ in L, i.e. $\sigma s =_E \sigma t$.

Finally we define the *complexity* of a list of equational goals G by $C(G, [])$. In words: each goal $l = r$ in G is assigned the maximal depth of l under instantiations resulting from a solution of all goals to the left; the overall complexity is the multiset of all these integers.

Some important facts about C are

1. $C(\sigma R, \sigma L) \leq C(R, L)$
2. If $\sigma \vdash_E H$ implies $\sigma \vdash_E H'$ for all σ, then $C(R, H) \leq C(R, H')$.

Each resolution step transforms the list of goals from $[l = r]@G$ to $H@G'$, where $G' = \theta G$, θ is the unifying substitution, and H is defined in (15) above. Since $C(H@G', []) = C(H_l, []) \sqcup C(H_r, H_l) \sqcup C(G', H)$ and $C([l = r]@G, []) = C([l = r], []) \sqcup C(G, [l = r])$, it suffices to show $C(G', H) \leq C(G, [l = r])$, $C(H_l, []) < C([l = r], [])$, and $C(H_r, H_l) < C([l = r], [])$ in order to prove $C(H@G', []) < C([l = r]@G, [])$.

$C(G', H) \leq C(G, [l = r])$ follows from the two facts about C above because the rules in D_{rk} guarantee that $\sigma \vdash_E H$ implies $\sigma l =_E \sigma r$.

To show $C(H_l, []) < C([l = r], [])$ and $C(H_r, H_l) < C([l = r], [])$ we notice that $C([l = r], []) = maxd_E(l) \neq 0$. Looking at the equations in H_l, we find that all l_i are proper subterms of l and hence that $maxd_E(\sigma l_i) = maxd_E(l_i) < maxd_E(l)$, which establishes $C(H_l, []) < C([l = r], [])$. If H_r is nonempty, resolution must have taken place with a rule in D derived from an equation $s = t$ in E. Thus $\sigma \vdash_E H_l$ implies $l =_E \sigma s =_E \sigma t$. Hence σt_j is ground and a proper subterm of σt, and therefore $maxd_E(\sigma t_j) < maxd_E(\sigma t) = maxd_E(l)$. This proves $C(H_r, H_l) < C([l = r], [])$ and concludes the termination proof. □

Theorem 2 is important because it is the first time that a subclass of resolvant presentations has been identified which yield *terminating* matching algorithms. On the other hand, there is a trivial terminating matching algorithm for permutative theories: in trying to match the pattern r to the variable free term s, enumerate the finite set of t's with $s =_E t$ and try to match r and t in the empty theory. The time and space complexity of these algorithms remains to be investigated.

We will now stop to think in terms of inference rules and confine our attention to equations.

4.1 A Generalized Criterion

Given a set of equations E we want to test whether E is resolvant. Our criterion actually shows how to go from an arbitrary derivation $s =_E t$ to one with at most one peak. The transformation is an inductive process which combines adjacent peaks.

If we write $e, e' \in E$ in the sequel, we really mean that there are two equations e and e_1 in E, such that $e' = \sigma e_1$, where σ is a renaming of the variables in e_1 away from those in e. We also assume that E is closed under symmetry, i.e. $s{=}t \in E$, implies $t{=}s \in E$.

Defining

$$p{=}q \Downarrow_E u{=}v \;\Leftrightarrow\; \forall\sigma.\, \sigma q \mathbin{\dot{=}}_E \sigma u \;\Rightarrow\; \sigma p \mathbin{\dot{=}}_E \sigma v$$

we get

Lemma 1 *E is resolvant iff $p{=}q \Downarrow_E u{=}v$ holds for all equations $p{=}q, u{=}v \in E$ where $q = f(\ldots)$ and $u = f(\ldots)$.*

Proof We concentrate on the "if"-part as the "only if"-part is trivial. For E to be resolvant, $s =_E t$ must imply $s \mathbin{\dot{=}}_E t$. We show that under the given assumptions any derivation $s =_E t$ can be reduced to $s \mathbin{\dot{=}}_E t$. The reduction merges adjacent peaks. This means we translate a derivation $s_0 \mathbin{\dot{=}}_E s_1 \mathbin{\dot{=}}_E s_2 \mathbin{\dot{=}}_E s_3$ into one of the form $s_0 \mathbin{\dot{=}}_E s_3$. By induction the number of peaks in any derivation can always be reduced to 0 or 1.

A derivation $s_0 \mathbin{\dot{=}}_E s_1 \mathbin{\dot{=}}_E s_2 \mathbin{\dot{=}}_E s_3$ must be of the form $\sigma p \mathbin{\dot{=}}_E \sigma q \mathbin{\dot{=}}_E \sigma u \mathbin{\dot{=}}_E \sigma v$ for some $p{=}q, u{=}v \in E$ with $q = f(\ldots)$ and $u = f(\ldots)$. By assumption this implies $\sigma p \mathbin{\dot{=}}_E \sigma v$, which is the required reduction. □

We will now concentrate on ways of turning the resolvance criterion embodied in this lemma into a finite test.

In the sequel let $\bar{s}$ denote the instantiation of every variable in s by a unique free constants. Thus $\bar{s}$ is equivalent to τs, where τ is a fixed injective substitution from variables to free constants. Therefore any first-order formula $\bar{P}$ holds iff σP holds for all σ.

Lemma 2 *Let $p{=}q, u{=}v \in E$ such that $q = f(q_1, \ldots, q_n)$, and $u = f(u_1, \ldots, u_n)$, and let $E' = E \cup \{\overline{q_1} = \overline{u_1}, \ldots, \overline{q_n} = \overline{u_n}\}$. Then $\bar{p} \mathbin{\dot{=}}_{E'} \bar{v}$ implies $p{=}q \Downarrow_E u{=}v$.*

Proof In the following we make use of the fact that if $A \vdash P \Rightarrow Q$ then $A \vdash P$ implies $A \vdash Q$.

$$\begin{aligned}
& \bar{p} \mathbin{\dot{=}}_{E'} \bar{v} \\
\Leftrightarrow\;& E \cup \{\overline{q_1} = \overline{u_1}, \ldots, \overline{q_n} = \overline{u_n}\} \vdash \bar{p} \mathbin{\dot{=}} \bar{v} \\
\Leftrightarrow\;& E \vdash \overline{q_1} = \overline{u_1} \wedge \ldots \wedge \overline{q_n} = \overline{u_n} \Rightarrow \bar{p} \mathbin{\dot{=}} \bar{v} \\
\Leftrightarrow\;& E \vdash \bar{q} = \bar{u} \Rightarrow \bar{p} \mathbin{\dot{=}} \bar{v} \\
\Rightarrow\;& \bar{q} =_E \bar{u} \Rightarrow \bar{p} \mathbin{\dot{=}}_E \bar{v} \\
\Leftrightarrow\;& \forall\sigma.\, \sigma q =_E \sigma u \Rightarrow \sigma p \mathbin{\dot{=}}_E \sigma v \\
\Leftrightarrow\;& p{=}q \Downarrow_E u{=}v
\end{aligned}$$

□

Example 1 In the presence of commutativity in E, we need to test $x{\cdot}y{=}y{\cdot}x \Downarrow_E u{\cdot}v{=}v{\cdot}u$. By Lemma 2 it suffices to show $\bar{x}{\cdot}\bar{y} \mathbin{\dot{=}}_{E'} \bar{v}{\cdot}\bar{u}$ where $E' = E \cup \{\bar{y} = \bar{u}, \bar{x} = \bar{v}\}$: $\bar{x}{\cdot}\bar{y} \mathbin{\dot{=}}_{E'} \bar{v}{\cdot}\bar{u}$.

If E contains $x{\cdot}y = y{\cdot}x$, $(x{\cdot}y){\cdot}z = x{\cdot}(y{\cdot}z)$, and $(x{\cdot}y){\cdot}z = (x{\cdot}z){\cdot}y$ we need to show $(x{\cdot}y){\cdot}z{=}x{\cdot}(y{\cdot}z) \Downarrow_E u{\cdot}v{=}v{\cdot}u$ and hence $(\bar{x}{\cdot}\bar{y}){\cdot}\bar{z} \mathbin{\dot{=}}_{E'} \bar{v}{\cdot}\bar{u}$ where $E' = E \cup \{\bar{x} = \bar{u}, \bar{y}{\cdot}\bar{z} = \bar{v}\}$: $(\bar{x}{\cdot}\bar{y}){\cdot}\bar{z} \mathbin{\dot{=}}_{E'} (\bar{y}{\cdot}\bar{x}){\cdot}\bar{z} \mathbin{\dot{=}}_{E'} (\bar{y}{\cdot}\bar{z}){\cdot}\bar{x} \mathbin{\dot{=}}_{E'} \bar{v}{\cdot}\bar{u}$

However, Lemma 2 is not necessary for resolvance:

Example 2 Right-commutativity is resolvant and $(x{\cdot}y){\cdot}z{=}(x{\cdot}z){\cdot}y \Downarrow_{C_r} (u{\cdot}v){\cdot}w{=}(u{\cdot}w){\cdot}v$ holds although Lemma 2 is not applicable: $(\bar{x}{\cdot}\bar{y}){\cdot}\bar{z} \doteqdot_{E'} (\bar{u}{\cdot}\bar{w}){\cdot}\bar{v}$, where $E' = C_r \cup \{\bar{x}{\cdot}\bar{z} = \bar{u}{\cdot}\bar{v}, \bar{y} = \bar{w}\}$, does not hold.

The problem stems from the fact that in E' we only assume that $\overline{q_i} = \overline{r_i}$, without any further distinctions. We have to take into account what the derivation of $q_i =_E r_i$ looks like. For right-commutativity it is sufficient to take a closer look at $x{\cdot}z =_E u{\cdot}v$. We can assume that this derivation has at most one peak.

If $x{\cdot}z \mathrel{\dot{=}}_E u{\cdot}v$, we have $E'_1 = E' \cup \{\bar{x} = \bar{u}, \bar{z} = \bar{v}\}$ and therefore $(\bar{x}{\cdot}\bar{y}){\cdot}\bar{z} \doteqdot_{E'_1} (\bar{u}{\cdot}\bar{w}){\cdot}\bar{v}$.

Otherwise there are terms r,s,t such that $x{\cdot}z \mathrel{\dot{=}}_E (r{\cdot}s){\cdot}t \doteqdot_E (r{\cdot}t){\cdot}s \mathrel{\dot{=}}_E u{\cdot}v$, which gives rise to $E'_2 = E' \cup \{\bar{x} = \bar{r}{\cdot}\bar{s}, \bar{z} = \bar{t}, \bar{r}{\cdot}\bar{t} = \bar{u}, \bar{s} = \bar{v}\}$. Therefore $(\bar{x}{\cdot}\bar{y}){\cdot}\bar{z} \mathrel{\dot{=}}_{E'_2} ((\bar{r}{\cdot}\bar{w}){\cdot}\bar{s}){\cdot}\bar{t} \doteqdot_{E'_2} ((\bar{r}{\cdot}\bar{w}){\cdot}\bar{t}){\cdot}\bar{s} \mathrel{\dot{=}}_{E'_2} (\bar{u}{\cdot}\bar{w}){\cdot}\bar{v}$.

Thus we have shown that $(\bar{x}{\cdot}\bar{y}){\cdot}\bar{z} \doteqdot_{E'_i} (\bar{u}{\cdot}\bar{w}){\cdot}\bar{v}$ holds for $i = 1, 2$.

The case distinction of the previous example can be formalized as follows:

$$\begin{aligned}
C_E(f(s_1,\ldots){=}f(t_1,\ldots)) &= \{\{s_1 = t_1,\ldots\}\} \cup \\
&\quad \{\{s_1 = u_1,\ldots,v_1 = t_1,\ldots\} \mid f(u_1,\ldots){=}f(v_1,\ldots) \in E\} \\
C_E(f(s_1,\ldots){=}g(t_1,\ldots)) &= \{\{s_1 = u_1,\ldots,v_1 = t_1,\ldots\} \mid f(u_1,\ldots){=}g(v_1,\ldots) \in E\} \\
C_E(x{=}y) &= \{\{x = y\}\}
\end{aligned}$$

The three clauses that define C_E are ordered in the sense of ML or Prolog. For example the last one, which is the default case, applies only if the first two do not. The next two facts show that the definition of $C_E(s = t)$ returns exactly those equalities between subterms that can arise if $s' \mathrel{\dot{=}}_E t'$ for some instances s' and t' of s and t respectively.

$$C \in C_E(s = t) \;\Rightarrow\; \bar{s} \mathrel{\dot{=}}_{E \cup \bar{C}} \bar{t} \tag{16}$$

$$\sigma s \mathrel{\dot{=}}_E \sigma t \;\Rightarrow\; \exists C \in C_E(s = t).\forall p{=}q \in C.\ \sigma p =_E \sigma q \tag{17}$$

With these facts we can prove the following lemma which formalizes the procedure outlined in Example 2.

Lemma 3 *If $\bar{p} \mathrel{\dot{=}}_{E \cup \bar{C}} \bar{v}$ holds for all $p{=}q, u{=}v \in E$ such that $q = f(q_1,\ldots,q_n)$ and $u = f(u_1,\ldots,u_n)$, and all $C \in M = \{E_1 \cup \ldots \cup E_n \mid E_i \in C_E(q_i{=}u_i)\}$, then $p{=}q \Downarrow_E u{=}v$ holds for all $p{=}q, u{=}v \in E$, i.e. E is resolvant.*

Proof The proof is similar to the one of Lemma 1 in that it proceeds by reducing peaks. However, we are more specific about the order in which adjacent peaks are removed from a derivation and its subderivations: we reduced a derivation $\sigma p \mathrel{\dot{=}}_E \sigma q \mathrel{\dot{=}}_E \sigma u \mathrel{\dot{=}}_E \sigma v$, where $p{=}q, u{=}v \in E$, $q = f(q_1,\ldots,q_n)$ and $u = f(u_1,\ldots,u_n)$, only if all subderivations $\sigma q_i =_E \sigma u_i$ are normalized, i.e. of the form $\sigma q_i \mathrel{\dot{=}}_E \sigma u_i$. In that case we know from (17) that there is a $C_i \in C_E(q_i = u_i)$ such that $\sigma s =_E \sigma t$ holds for all $s{=}t \in C_i$. Letting $C = C_1 \cup \ldots \cup C_n \in M$ we obtain

$$\begin{aligned}
\bar{p} \mathrel{\dot{=}}_{E \cup \bar{C}} \bar{v} &\Leftrightarrow E \vdash (\forall i, s{=}t \in C_i.\ \bar{s} = \bar{t}) \Rightarrow \bar{p} \mathrel{\dot{=}} \bar{v} \\
&\Rightarrow (\forall i, s{=}t \in C_i.\ \bar{s} =_E \bar{t}) \Rightarrow \bar{p} \mathrel{\dot{=}}_E \bar{v} \\
&\Rightarrow (\forall i, s{=}t \in C_i.\ \sigma s =_E \sigma t) \Rightarrow \sigma p \mathrel{\dot{=}}_E \sigma v \\
&\Leftrightarrow \sigma p \mathrel{\dot{=}}_E \sigma v
\end{aligned}$$

the required reduction. □

This concludes the exposition of the theoretical basis for our resolvance checker.

4.2 Automating It

The procedures suggested by Lemmas 2 and 3 lead to a fairly large number of cases that have to be examined. The principal problem with automating this procedure is the undecidability of the concepts involved, i.e. the predicates $\neq_E$ etc. As remarked above, these predicates, and hence Lemmas 2 and 3, become decidable for permutative E. However, in practice it turns out that the resulting search space is far too large for a naive enumeration procedure.

A prototype system for automating these tests has been implemented in Prolog. To overcome the problems just mentioned, a number of heuristics for testing $\neq_E$ were implemented. If these turn out to be insufficient (as they have for a number of presentations) the predicate $\neq_E$ can be implemented separately for each E.

4.3 More Theories

Using the implementation described in the previous section, the following list of presentations was shown to be resolvant:

$$\begin{aligned}
AC1^+ &= \{(x{\cdot}y){\cdot}(u{\cdot}v) = (x{\cdot}u){\cdot}(y{\cdot}v), 1{\cdot}x = x, x{\cdot}1 = x\} \\
D &= \{x{\cdot}(y+z) = x{\cdot}y + x{\cdot}z\} \\
I &= \{x{\cdot}x = x\} \\
CI &= C \cup I \\
DC &= D \cup \{x+y = y+x\} \\
DC_r &= D \cup \{(x+y)+z = (x+z)+y\} \\
DA &= D \cup \{(x+y)+z = x+(y+z)\} \\
C_S &= \{(x{\cdot}x){\cdot}y = y{\cdot}(x{\cdot}x)\} \\
C_P &= \{(x{\cdot}y){\cdot}z = z{\cdot}(x{\cdot}y)\}
\end{aligned}$$

I and CI yield terminating unification algorithms, the rest only terminating matching algorithms. For D this has already been exploited by Mzali [11].

We also conjecture that $ACI^+ = AC^+ \cup I$ and $ACI1^+ = AC1^+ \cup I$ are resolvant presentations. However, the methods of this paper seem inadequate to prove it.

4.4 Combining Resolvant Theories

In this section we investigate the combination of resolvant presentations. We are given a collection of equational presentations E_i, $i \in I$, over pairwise disjoint signatures Σ_i. Let $E = \bigcup_{i\in I} E_i$ and $\Sigma = \bigcup_{i\in I} \Sigma_i$. Adapting the terminology of [10] we call a property $\mathcal{P}$ of equational presentations *modular* if E has property $\mathcal{P}$ iff all E_i have property $\mathcal{P}$.

As an immediate consequence of Lemma 1 we obtain that

Lemma 4 *Resolvance is modular.*

Proof Assume that all E_i are resolvant. Given two equations $p{=}q, u{=}v \in E$ such that $q = f(\ldots)$ and $u = f(\ldots)$, disjointness of the Σ_i implies that both equations must come from the same E_k. Hence $p{=}q \Downarrow_{E_k} u{=}v$ and thus $p{=}q \Downarrow_E u{=}v$ must hold, which implies resolvance of E.

Now assume that E is resolvant and let s, t be two terms over Σ_k with $s =_{E_k} t$ and thus in particular $s =_E t$. Resolvance of E implies $s \doteq_E t$. Using disjointness of the signatures and Fact 1 in [12] $s \doteq_{E_k} t$ follows. Thus E_k is resolvant. □

This lemma does for the combination of resolvant presentations what [15,14,12] do for the combination of arbitrary unification or matching algorithms. The simplicity of its proof is partly due to the fact that it says nothing about termination: even if each E_i yields a terminating unification or matching algorithm, E might not. For permutative theories however we are able to show that termination carries over. The reason is that

Lemma 5 *Permutativity is modular.*

Proof This proof relies heavily on notions defined in [12] which appear in quotation marks.

Assume that all E_i are permutative. The proof is by induction on the "theory height" of terms over Σ, i.e. the maximum number of alternations of signature along some path in a mixed term considered as a DAG. Any term s over Σ can be written as $C[s_1, \ldots, s_m]$, where C is a proper "context" over some Σ_k and the s_i are the "immediate alien subterms". By induction hypothesis each s_i has a finite E-equivalence class. Since E is both regular and collapse-free, Fact 2 in [12] shows that any term t with $s =_E t$ is of the form $D[t_1, \ldots, t_n]$ such that each t_j is E-equivalent to some s_i (and vice versa), and $C[[s_1]_{=_E}, \ldots] =_{E_k} D[[t_1]_{=_E}, \ldots]$. Thus there are only finitely many different t_j and D, and therefore only a finite number of t with $s =_E t$, i.e. E is also permutative.

The reverse implication is trivial. □

Thus we can safely combine permutative resolvant presentations to obtain a matching algorithm for the joint theory.

5 Related Work

As was mentioned earlier on, this paper is strongly related to the work of Claude Kirchner, who coined the term "resolvant" [8]. He also suggested some criteria for checking resolvance and a completion procedure which turns a non-resolvant theory into a resolvant one. However, for most equational theories discussed above his criteria are too weak to determine that they are resolvant. This prompted the development of the improved tests in this paper. Recently, Claude Kirchner and Francis Klay have given a nice and simple characterization of resolvant theories:

Theorem 3 (Kirchner, Klay [9]) *If E is a collapse-free set of equations over the signature Σ, the set*

$$\{\sigma f(x_1, \ldots) = \sigma g(y_1, \ldots) \mid f, g \in \Sigma, \sigma \in cU_E(f(x_1, \ldots) = g(y_1, \ldots))\}$$

is a resolvant set of axioms which generates the same equational theory as E. (The x_i and y_j are all distinct and cU_E denotes a complete set of E-unifiers).

Thus any equational theory with a finitary unification problem has a finite resolvant presentation. This presentation can be computed by means of a unification algorithm for the theory.

Although this theorem yields very short proofs for the resolvance of the theories in Sections 3.1 to 3.5, it is not easy to apply to some of the theories in Section 4.3. For example D has a fairly involved unification algorithm (see [1]), and the author is not aware of a published unification algorithm for DC or DA. The problem is that a notion that was conceived to explain and automate the generation of unification algorithms, namely resolvance, is itself reduced to unification.

If a complete unification algorithm for some theory E is not known, it may still be possible to compute a resolvant presentation by combining Theorem 3 with the results of this paper: instead of computing a complete set of unifiers, enumerate all unifiers one by one and keep on testing whether the resulting presentation is yet resolvant. Of course this procedure may still not succeed because we can reach a resolvant presentation without realizing it, due to the incompleteness of the test we described.

Acknowledgements

Part of this research was carried out at the Laboratory for Computer Science at MIT. I would like to express my gratitude to both John Guttag and Larry Paulson for giving me the freedom to pursue this line of research. Larry Paulson and David Wolfram helped to improve the presentation.

References

[1] S. Arnborg, E. Tidén: *Unification Problems with One-Sided Distributivity*, Proc. 1st Conf. Rewriting Techniques and Applications, LNCS 202 (1985), 398-406.

[2] H. Bertling, H. Ganzinger, R. Schäfer: *CEC: A System for Conditional Equational Completion*, PROSPECTRA-Report M.1.3-R-7.0, Universität Dortmund (1988).

[3] N. Dershowitz, Z. Manna: *Proving Termination with Multiset Orderings*, CACM 22 (1979), 465-476.

[4] H. Ganzinger, Private system demonstration, June 1989.

[5] S.J. Garland, J.V. Guttag: *An Overview of LP, The Larch Prover*, in: Proc. 3rd Intl. Conf. Rewriting Techniques and Applications, LNCS 355 (1989), 137-151.

[6] G. Huet, D.C. Oppen: *Equations and Rewrite Rules - A Survey*, in: Formal Languages: Perspectives and Open Problems, R. Book (ed.), Academic Press (1982).

[7] C. Kirchner: *Méthodes et outils de conception systématique d'algorithmes d'unification dans les théories équationnelles*, Thèse d'état de l'Université de Nancy I (1985).

[8] C. Kirchner: *Computing Unification Algorithms*, Proc. Symp. on Logic in Computer Science (1986), Cambridge, MA, 206-216.

[9] C. Kirchner, F. Klay: *A Note on Syntacticness*, Presentation at the 3rd Intl. Workshop on Unification, Lambrecht, FRG, June 1989.

[10] A. Middeldorp: *Modular Aspect of Properties of Term Rewriting Systems Related to Normal Forms*, Proc. 3rd Intl. Conf. Rewriting Techniques and Applications, LNCS 355 (1989), 263-277.

[11] J. Mzali: *Matching with Distributivity*, Proc. 8th Intl. Conf. on Automated Deduction, LNCS 230 (1986), 496-505.

[12] T. Nipkow: *Combining Matching Algorithms: The Regular Case*, Proc. 3rd Intl. Conf. Rewriting Techniques and Applications, LNCS 355 (1989), 343-358.

[13] G.D. Plotkin: *Building-in Equational Theories*, in: Machine Intelligence, Vol. 7, Halsted Press (1972), 73-90.

[14] E. Tidén: *Unification in Combinations of Collapse-Free Theories with Disjoint Sets of Function Symbols*, in: Proc. CADE-8, LNCS 230 (1986), 431-449.

[15] K. Yelick: *Unification in Combinations of Collapse-Free Regular Theories*, JSC 3 (1987), 153-181.