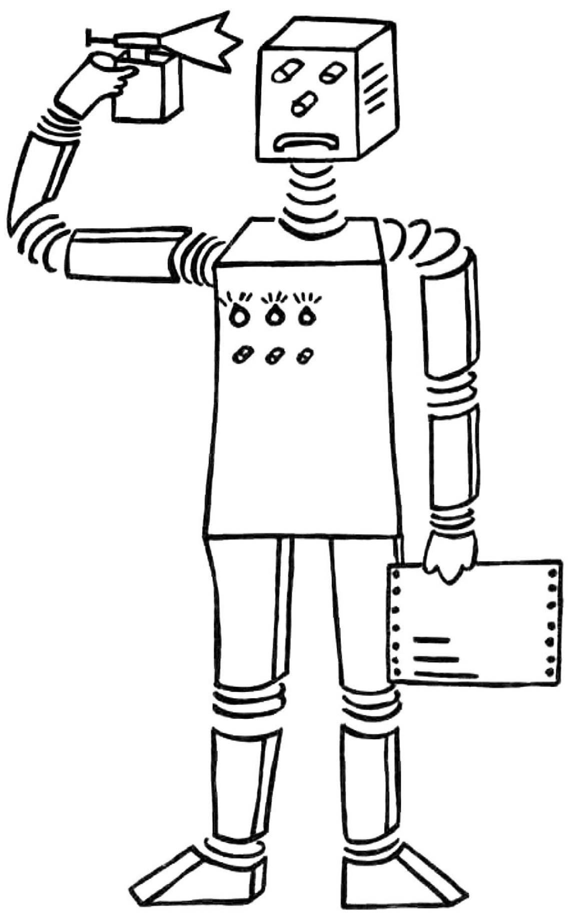
Fachbereich Informatik
Universität Kaiserslautern
Postfach 3049
D-6750 Kaiserslautern 1, W. Germany

Artificial
Intelligence
Laboratories

SEKI-REPORT



Construction of Equality Graphs

Karl-Hans Bläsius

April 1986 SEKI-REPORT SR-86-01

Construction of Equality Graphs

Karl-Hans Bläsius
Universität Kaiserslautern
Fachbereich Informatik
D-6750 Kaiserslautern

ABSTRACT

The theoretical and practical problems of equality reasoning in Automated Deduction are notorious. A new method is presented to cope with the enormous search space that usually arises when equational axioms are present. Starting from an empty graph a production system constructs graphs which represent solutions for simpler problems defined by abstraction. These graphs contain global information and are plans for guiding the search for a proof of the original problem, represented in the final graph. The construction of equality graphs is based on the idea to search for the differences between two terms by seperating toplevel symbol and subterms of a functional term. The impact of the explicit representation of information contained in the inference system on the control of inferences is discussed. Finally the method is compared to other equality reasoning methods.

1. Introduction

Equality is an important relation in theorem proving, not least because it is so frequently used in mathematical formalism. The problem of deciding whether two terms s and t are equal - relative to a given set of equations E - may occur many times during a proof. The hitherto existing experience in the field of automated theorem proving has shown that it is extremely difficult to find efficient methods of handling such equality problems automatically, since normally large search spaces arise. The following example is a standard test problem in the field: A group with $x^2 = e$ is commutative. Bundy ([Bu83] page 84 - 88) analysed this example in detail. For a breadth first paramodulation [RW69] proof, he estimated a search space of 12^{10} paramodulation steps.

Most existing automatic deduction systems are based on some sort of unification. For these systems subproblems arise of the following kind: decide whether two terms s and t are unifiable under a given (equational) theory E, i.e. instances s' and t' of the terms s and t are to be found, such that s' and t' are equal under E. Such a problem is called an equality problem and is denoted by $\langle s \equiv_E t \rangle$.

It is well known that in general (i.e. for arbitrary sets of equations E) the unifiability of two terms under a theory E is undecidable. However, for certain sets of equations E it is possible to find an algorithm which solves the equality problem $\langle s \equiv_E t \rangle$ for any terms s and t. Such classes of equations are investigated in Unificationtheory (e.g. [Hu76], [Pl72], [Si84], [SS81]). The purpose of our equality reasoning method is to solve equality problems for arbitrary sets of equations E using equality graphs to heuristically guide the search. This work is somehow similar to the investigation into universal unification algorithms (e.g. [Hul80], [Sz82], [JK83]). However whereas they usually have strong prerequisites for their algorithms to work (like Noetherianess, Confluence etc of the equational theories) our method applies to any equational axioms - hence the strong emphasis on heuristics and artificial intelligence techniques.

The experience in automated theorem proving demonstrated very early that the explicit use of the equality axioms (reflexivity, symmetry, transitivity and substitution axioms) is very inefficient. Therefore many methods have been developed to incorporate equality somehow directly into the proof procedure ([WRC67], [RW69], [Si69], [Mo69], [Br75], [Sh78], [HR78], [Di79], [LH85]).

Many equality reasoning methods are based on the subterm replacement principle: a subterm s of a term t can be replaced by a term r, if s and r are equal relative to a given set of equations E. Paramodulation [RW69] for instance is an inference rule that instantiates a term by applying a unifying substitution and replaces a subterm by an equal term.

Other equality reasoning methods are based on the difference reduction principle: The difference of two terms is searched for and tried to be minimized or removed where the type of difference guides the operator to be applied just like in GPS [NS59]. Procedures based on such methods usually compare the heads of both terms and try to make them equal, and then try to unify the pairs of corresponding subterms. In most known methods the resulting unifiers are immediately applied to the following pairs of subterms. RUE-resolution [Di79] is an example for an equality reasoning method based on the difference reduction principle: Two literals with the same predicate symbol and different sign can be resolved upon even if not all of the corresponding subterm-pairs are unifiable. The disagreement pairs (non-unifiable subterm-pairs) are added to the derived clause as negated equations.

Some equality reasoning methods combine both principles: replacement operations are performed under the control of difference reduction. [LH85]

Equality reasoning methods based on graphs have been developed: The connection graph method for resolution [Ro65] introduced by Kowalski [Ko75] was extended by Siekmann and Wrightson [SW79] to incorporate paramodulation: In paramodulated clause graphs the possible operations are represented by links. Possible resolution steps are represented by R-links and paramodulation steps by P-links. Each deduction step modifying the clause set requires some modification of the graphical structure. Links are inherited thus saving additional search for new possible resolution - or paramodulation steps. The aim of the paramodulated clause graph procedures is to transform a total initial graph (where all possible operations are represented by links) into a graph containing the empty clause. Sequences of links can be examined in order to control the deduction by way of planning, but the graph contains only local information about the potential next steps. There is no explicit information about global co-operation of the links, hence the support of the graph structure for an efficient global planning is limited.

Experiments with the paramodulated clause graph procedure demonstrated, that the search space is too large: there is an enormous set of P-links created in most of the tested examples (the order of magnitude is often more than 10 000, i.e. 10000 possible operations in each step).

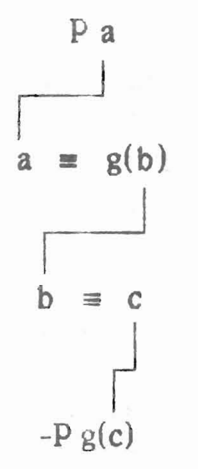
In [Bl83] we proposed a paramodulated clause graph procedure, which checks combinations of P-links for certain conditions. The aim was to reduce the search space by exploiting constraints: only those combinations of P-links fulfilling the conditions were to be considered as a potential solution for an equality problem. The analysis has shown that this method improves upon earlier methods but by and large it is inadequate also for equality reasoning (see section 2.1), since the search spaces for reasonably difficult problems are still for too large. But the experience with it led to another kind of search, based on the difference reduction principle and on graph construction: Starting from an initial graph, which is empty and contains no information, the actual graph is modified step by step while trying to reduce the difference between two terms till it results in a solution graph which represents the final proof.

In [Bl86] a formal presentation of our method as well as the soundness and completeness proofs are given (see also [Bl85]). In this paper we shall instead present how it works and some of its practical advantages over other approaches.

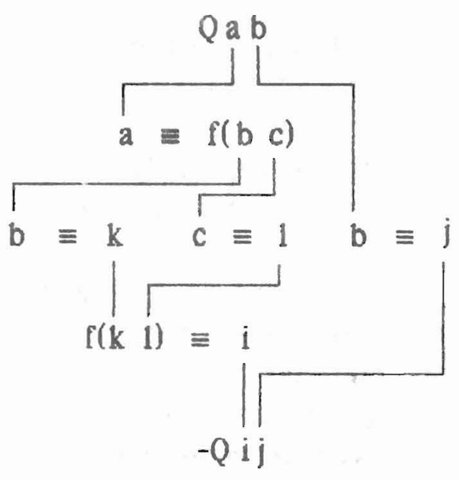
2. Equality Graphs

2.1. Constraints in Paramodulated Clause Graphs

Our first attempt to solve general "equality problems" was based on paramodulated clause graphs. Combinations of P-links were searched for which represent an executable paramodulation sequence modifying two literals such that they become resolvable [Bl83]. Such paramodulation sequences can be represented in a graph which consists of two potentially complementary literals (having the same predicate and opposite sign) and several equations connected with P-links. The graph represents a solution of the equality problem. Examples 2-1 and 2-2 show such graphs:

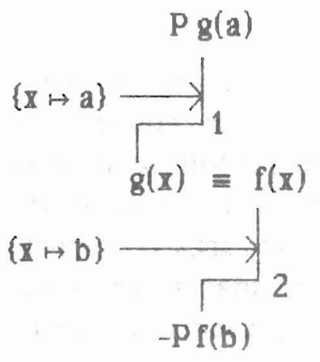


Example 2-1

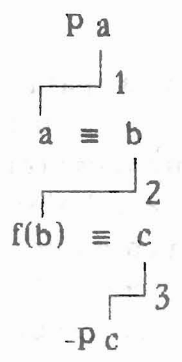


Example 2-2

Unfortunately not every graph represents an executable sequence of paramodulation steps as the following two examples show:



Example 2-3



Example 2-4

In example 2-3 the combination of the P-links 1 and 2 is impossible because their unifiers {x ↦ a} and {x ↦ b} are incompatible. In example 2-4 the P-links 1 and 2 are incompatible, because after paramodulation on link 1, link 2 cannot be inherited to the paramodulant Pb since the access depths do not coincide.

An equality graph that can not be executed is called incompatible, whereas a graph representing the solution of the equality problem for two potentially complementary literals is called compatible. The problem now is to find compatible graph structures. To this end several conditions can be stated which are necessary but not sufficient for the compatibility of a graph. In particular constraints can be formulated, which are not expensive to test but reduce the search space drastically.

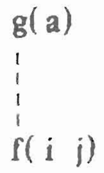
Some of the constraints to be used are: (1) All unifiers of the P-links concerned must merge to one most general unifier (i.e. the unifiers must be compatible, provided a proper variable renaming has been carried out). (2) For each maximal chain of P-links in a graph the sum of all access depths must be equal to zero and each partial sum must be less than or equal to zero. (3) Each combination of P-links containing an incompatible substructure is incompatible too.

Practical experiments with a procedure based on the above constraint satisfaction method have shown however, that the set of potential graphs (i.e. combinations of P-links), which have to be created in order to test for compatibility, is still far too large, even in relatively simple examples. Especially the P-links connecting variables (to everything else) make the procedure extremely inefficient.

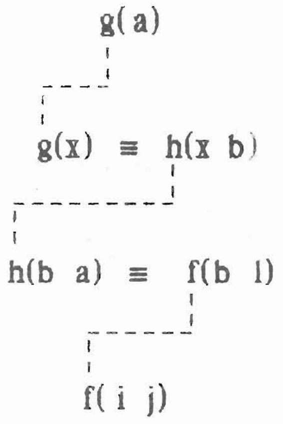
Hence the procedure was modified several times, such that only those graph structures fulfilling certain constraints were created. The experimental modifications finally led to the equality graph construction procedure (ECOP), which constructs compatible graphs without ever creating the enormous set of incompatible ones in the first place. The essential idea is: Starting from some initial state a sequence of operations is performed on each subsequent state until a compatible graph is constructed.

2.2. Constructing Solution Graphs

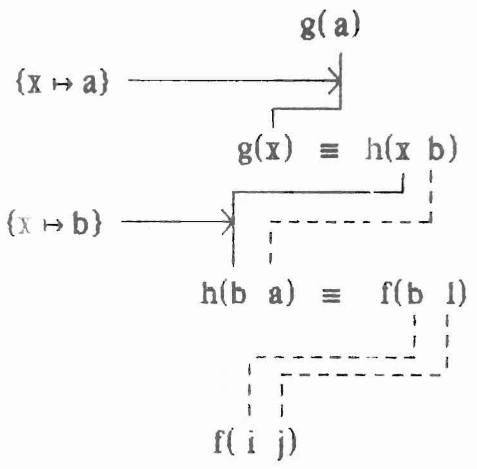
The graphs constructed while solving equality problems are called equality graphs and the resulting solution graph is called a compatible equality graph. The structure of equality graphs and their construction is demonstrated by the following example: Let $E = \{g(x) \equiv h(x\ \ b),\ h(y\ \ z) \equiv h(z\ \ y),\ h(b\ \ a) \equiv f(b\ \ l),\ b \equiv c,\ c \equiv i,\ l \equiv j\}$ and let $\langle g(a) \equiv_E f(i\ \ j) \rangle$ be the given equality problem, then the initial equality graph is



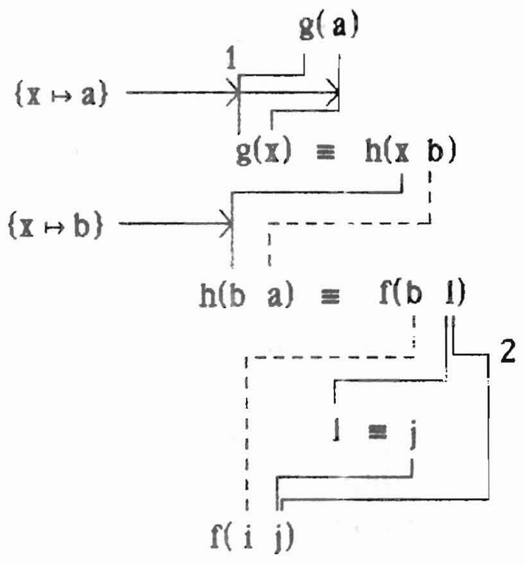
The only information of this graph is that the problem $\langle g(a) \equiv_E f(i\ \ j) \rangle$ is to be solved. The main discrepancy are the different toplevel symbols g and f. Hence this difference must be removed by some equations. There are two equations in E, which can be combined to a chain $g(x) \equiv h(x\ \ b)$ - - - - $h(b\ \ a) \equiv f(b\ \ l)$ which allows for the removal of this discrepancy and is inserted into the graph:



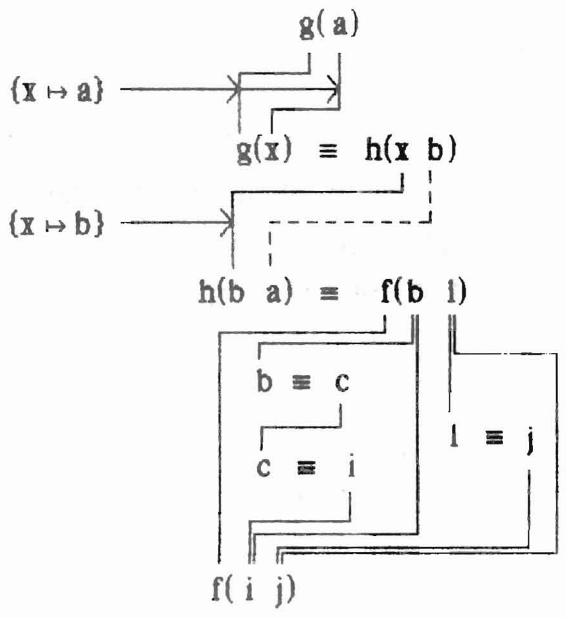
Three subproblems are created, which have to be solved: $\langle g(a) \equiv_E g(x) \rangle$ $\langle h(x\ \ b) \equiv_E h(b\ \ a) \rangle$ and $\langle f(b\ \ l) \equiv_E f(i\ \ j) \rangle$. In all three cases the heads of both terms are equal. Now the corresponding pairs of subterms generate new subproblems, some of which are trivially solved. We obtain the equality graph:



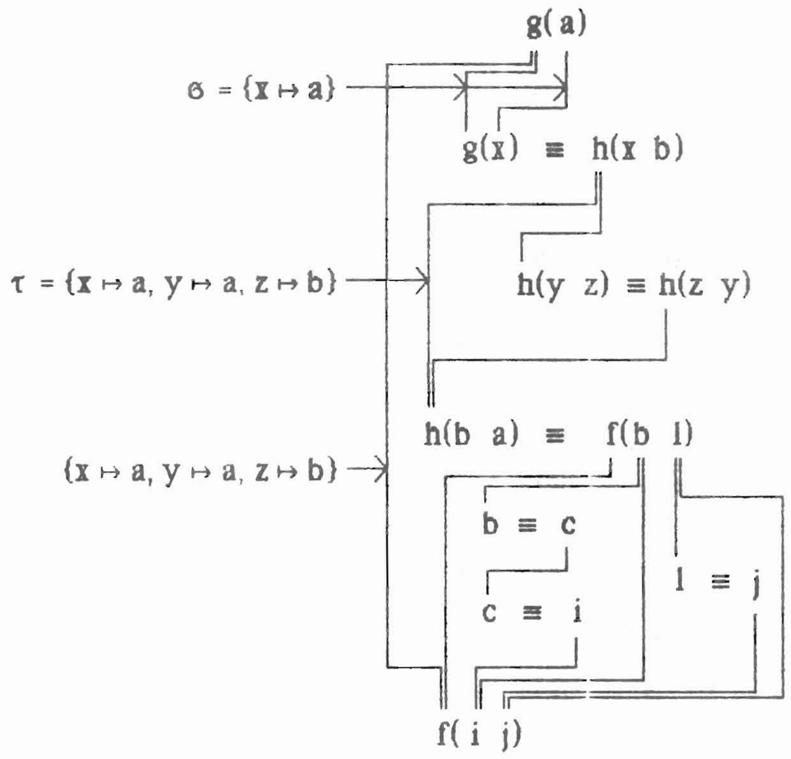
The links represent solved subproblems and are marked with a substitution, empty substitutions are omitted. Dotted lines indicate unsolved subproblems which can be selected for the next step. If the subproblem l - - - - j is selected, the following graph is constructed:



The substitutions within each chain must be checked for compatibility, i.e. they must be unified themselves and the result of this unification is marked on a separate link (2). In just the same way the solutions for the subproblems given by the corresponding subterm-pairs of two terms with the same function must be checked for compatibility and the result is also marked on a link (1). If the subproblem b - - - - i is selected for the next step, the chain b ≡ c - - - c ≡ i can be inserted with the result:



Now one unsolved subproblem remains: b - - - a. It is not possible to build a chain of equations from the given set E connecting b and a, hence this subproblem is unsolvable. However instead of solving b - - - a we can create a new subproblem at a higher term level: h(x b) - - - h(b a). There exists an appropriate equation in E: h(y z) ≡ h(z y) which is now inserted into the graph. Since the substitutions σ and τ are compatible, we have the final solution graph:



Starting from an empty graph a solution graph is constructed in a sequence of steps, where each possible intermediate graph is a solution at a certain level of abstraction (see [Pl81]) and is a global plan for the search for the solution of the original problem. The abstraction is weakened with each step, whereby the graph is refined. The dotted lines (unsolved subproblems) indicate positions where an abstraction is used. In our example an abstraction can be formulated as: disregard the second argument of the function h and the first argument of the function f. Usually such an abstraction cannot be expressed uniformly for all occurrences of the function, but depends on the position of its occurrence.

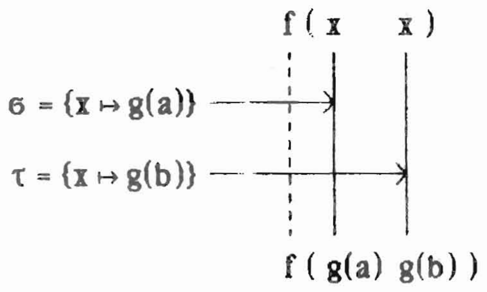
At each step the graph contains the information about the global correlation of the subproblems already solved and the information about the subproblems which are to be solved. Finally the position can be localized where the graph is to be modified in order to make solutions of subproblems compatible from a global point of view.

Equality graphs are constructed using a production system, which in a sense represents a meta calculus for the search for proofs. Equality graphs and this production system are defined in [Bl85].

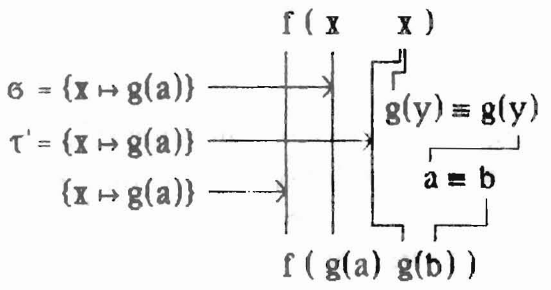
2.3. Term graphs and Substitution graphs

During the construction of equality graphs subproblems are often created of the form $\langle x \equiv_E t \rangle$ where x is a variable not contained in the term t. Such an equality problem is trivially solvable with the substitution $\{x \mapsto t\}$ without using any equations.

Let $\langle f(x\ x) \equiv_E f(g(a)\ g(b)) \rangle$ with $E = \{a \equiv b\}$ be an equality problem. There exist trivial solutions for the created subproblems but their substitutions σ and τ are not compatible:

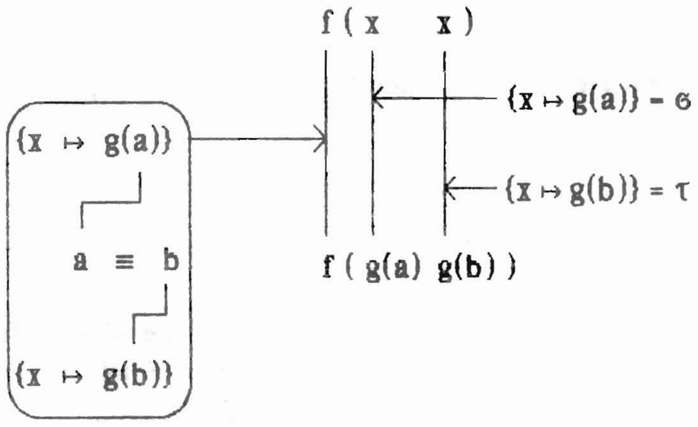


Most methods known based on the difference reduction principle use partial unifiers which are applied to the other subproblems before these are tried to be solved (e.g. [Di79], [LH85]). In our example the substitution σ would be applied to $\langle x \equiv_E g(b) \rangle$ yielding $\langle g(a) \equiv_E g(b) \rangle$. The disadvantage of the immediate application of partial unifiers will become apparent in the following section. Another approach could be the insertion of equations between x and g(b):



If the insertion of equations between the variable x and the term g(b) was allowed, then the search space would explode and especially it would be necessary to use functional reflexiv axioms. Furthermore each unifier τ' derivable with the use of equations would be equal under E to an instance of τ. Hence, if σ is unifiable with τ', then σ is also unifiable with τ under the theory E.

Therefore the unification of substitutions is performed under the theory E, and the discrepancies are removed where they occur:



σ and τ are the only unifiers derivable for the subproblems given in this example. It is neither allowed to insert any equations between x and g(b) nor to use functional reflexive axioms neither is σ applied to the second subproblem, but both substitutions are successfully unified under the theory E.

Hence there are two types of equality graphs corresponding to different kinds of subproblems. The unification of terms under a theory E is divided into two parts (subproblems):

1) local unification of two terms under E without applying the unifier to the next subterm-pairs
2) unification of a list of substitutions under E

The problem $\langle s \equiv_E t \rangle$ itself and each subproblem created recursively is solved by division into these two different kinds of subproblems. According to these kinds of subproblems two types of equality graphs are defined:

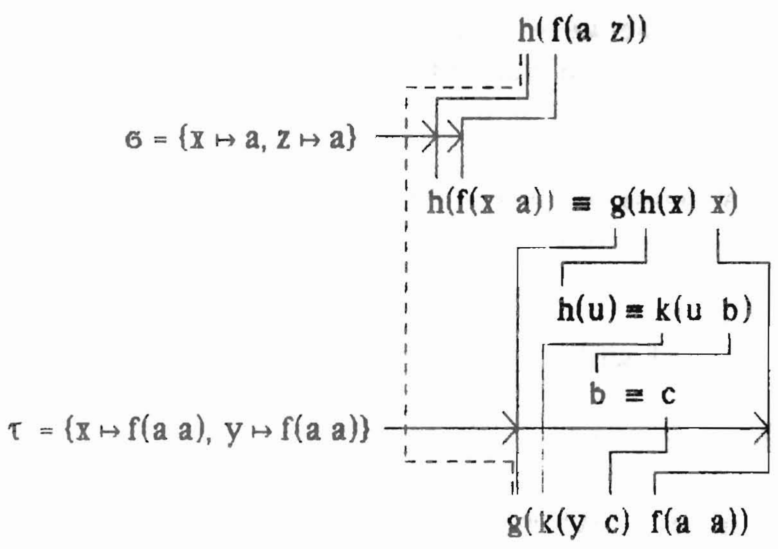
1) term graphs, representing proof plans for the unification of two terms under E
2) substitution graphs, representing proof plans for the unification of a list of substitutions under E

3. Alternative Subgraphs

For a given subproblem there are often many possible equality-chains which could be inserted and which would lead to different solutions. When solving a subproblem it is in general not possible to know which of the alternative partial solutions is the best from a global point of view, hence it is often necessary to consider several alternatives. Since it is too inefficient to construct a new subgraph for each possible equality-chain - there would be an enormous search space of graphs - we shall present two methods of how to handle alternative partial solutions.

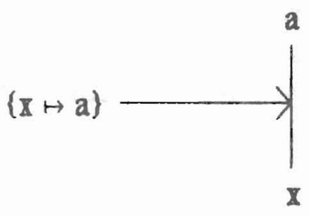
3.1. Subgraph Replacement

In special cases a subgraph for an already solved subproblem may be replaced by another subgraph representing another solution (i.e. another unifier) for the same subproblem. For example let $E = \{h(f(x\ a)) \equiv g(h(x)\ x),\ b \equiv c,\ h(u) \equiv k(u\ b),\ f(f(u\ v)\ w) \equiv f(u\ f(v\ w))\}$ and let $\langle h(f(a\ z)) \equiv_E g(k(y\ c)\ f(a\ a)) \rangle$ be an equality problem. The following equality graph may be constructed:

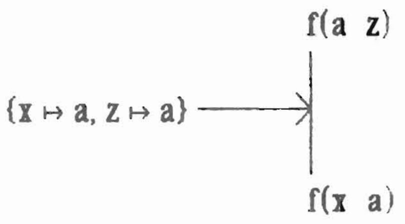


The dotted line represents the subproblem to unify the substitutions σ and τ. If this subproblem is solved, then the main problem would be finally solved. But it is unsolvable: σ and τ are not unifiable under E, since the unification fails for the variable x. Positions causing the incompatible assignments of the variable x can be localized within the graph, and the graph can be modified locally at that position with no effect on other parts of the graph, i.e. not disturbing the hitherto existing plan for a proof.

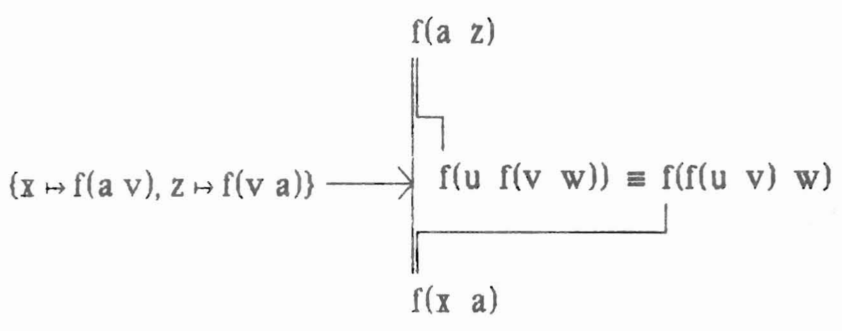
At occurrences of x other unifiers should be derivable, but it makes no sense to replace the subgraph



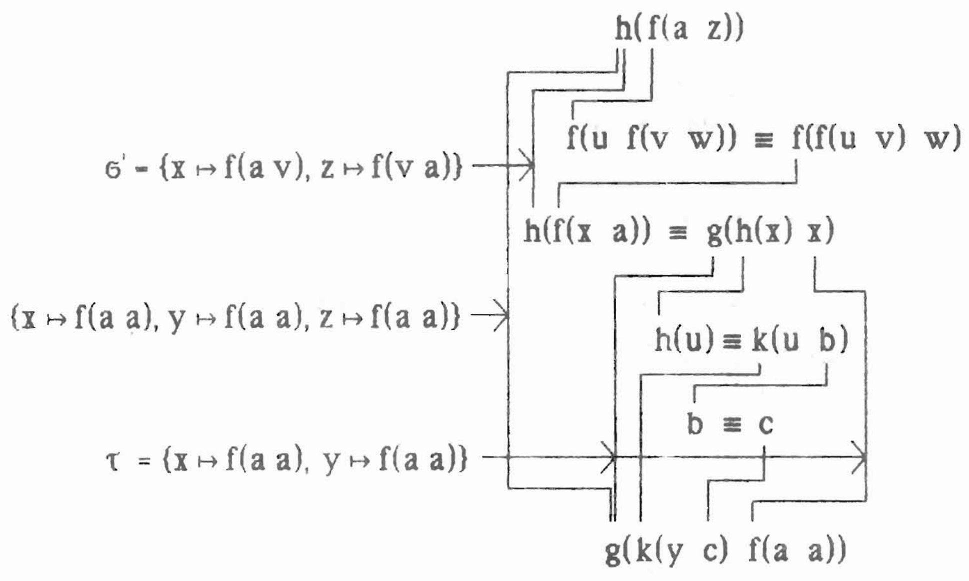
by another one where equations are inserted between a and x, because all unifiers derivable this way would be equal to {x ↦ a} under E, and would therefore not be unifiable with τ. Inserting equations in supergraphs of a — x could lead to new substitutions really different from σ. For example we can insert the equation f(u f(v w)) ≡ f(f(u v) w) between f(a z) and f(x a) and replace the subgraph



by the subgraph



with the result



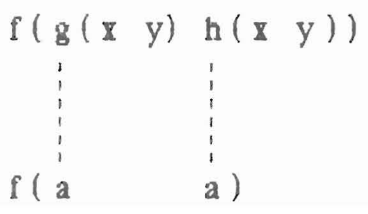
h(f(a z))
σ' = {x ↦ f(a v), z ↦ f(v a)}
f(u f(v w)) ≡ f(f(u v) w)
h(f(x a)) ≡ g(h(x) x)
{x ↦ f(a a), y ↦ f(a a), z ↦ f(a a)}
h(u) ≡ k(u b)
b ≡ c
τ = {x ↦ f(a a), y ↦ f(a a)}
g(k(y c) f(a a))

which is a solution for the given problem. The replacement operation is appropriate since the first solution of the subproblem is no longer required. The replacement of subgraphs neither destroys the solutions of other subproblems nor destroys the global plan.

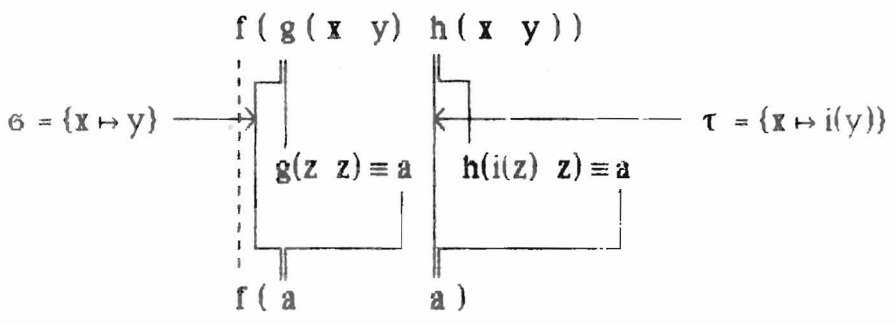
An immediate application of partial unifiers (unifiers of subproblems) to other subproblems would prevent us from finding the proper position for a subgraph replacement. Combinations of instantiations may pass through a large part of the graph, the origin of which cannot be found in case of a conflict.

3.2. Multiple Graphs

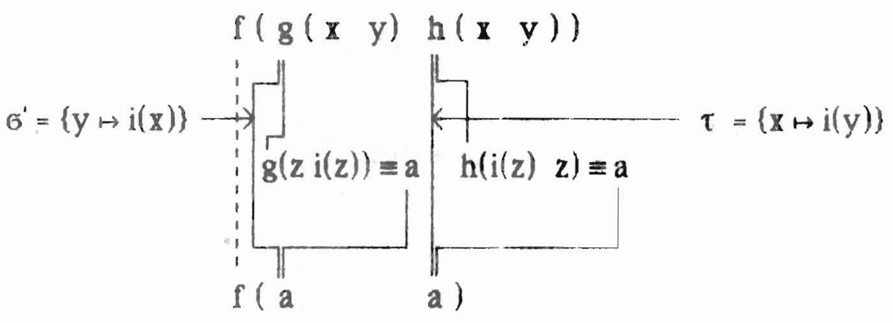
The following example demonstrates the necessity of multiple graphs. Let E = {g(z z) ≡ a, g(z i(z)) ≡ a, h(i(z) z) ≡ a, h(z z) ≡ a} and let ‹ f(g(x y) g(x y)) $\equiv_E$ f(a a) › be an equality problem. For each subproblem given by the corresponding subtermpairs



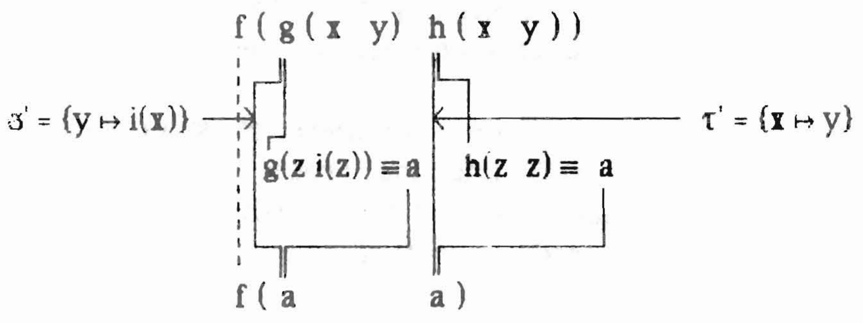
a solution may be constructed:



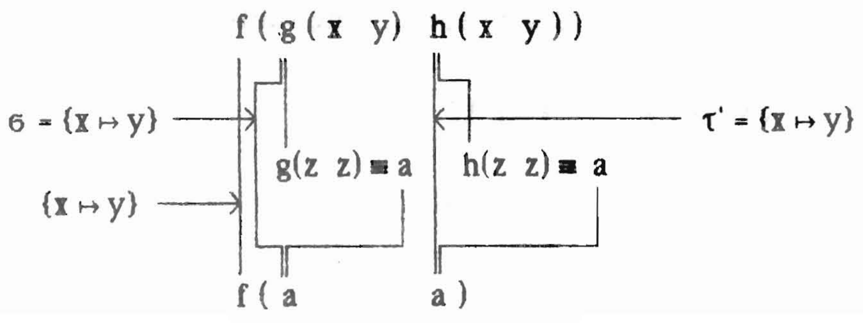
Since σ and τ are not unifiable under E, another solution σ' for the first subproblem has to be constructed, followed by a graph replacement operation:



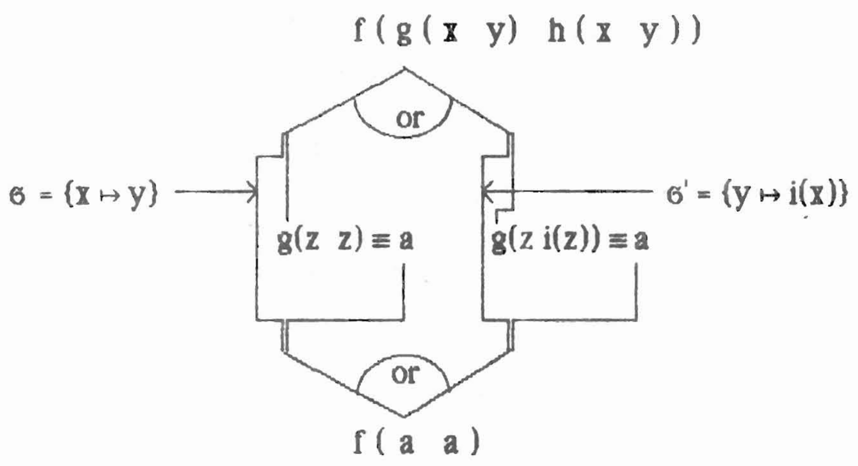
σ' and τ are still not unifiable and an alternative solution τ' for the second subproblem has to be constructed, followed by a graph replacement operation:



But σ' and τ' are not unifiable too. In order to find a solution for our problem the partial solution represented by the first σ must be reconstructed or the first graph replacement operation must be cancelled:



To avoid the search and reconstruction of the same partial solution many times (which is the case for many other equality reasoning methods, see section 5.2.3) our method is extended to allow the insertion of alternative partial solutions into the graph connected with an or-link:



Not one unifier, but a list of several unifiers (a list of alternative partial solutions) is attached to one subproblem. To check the compatibility of partial solutions a matrix of substitutions has to be evaluated.

$$\begin{pmatrix} \sigma_{11} & \sigma_{21} & \cdots & \sigma_{m1} \\ \vdots & \vdots & & \vdots \\ \sigma_{1n_1} & \sigma_{2n_2} & \cdots & \sigma_{mn_m} \end{pmatrix} \Big\updownarrow \text{or} \qquad \xleftrightarrow{\text{and}}$$

All substitutions in one column are alternative solutions for one subproblem and in different columns there are solutions for different subproblems. Each possible combination of substitutions (partial solutions) $(\sigma_{1j1}, \sigma_{2j2}, \dots, \sigma_{mjm})$ containing exactly one substitution (partial solution) from each column may be a solution for the superior problem (if these substitutions are compatible). $n_1 * n_2 * \dots * n_m$ such combinations exist.

With the use of multiple graphs (which represent some kind of structure sharing) each partial solution is searched for and constructed at most once. The derivability of all unifiers computable from the matrix requires the derivability of

$$\sum_{i=1}^{m} \prod_{j=1}^{i} n_j$$

partial solutions without the use of multiple graphs and

$$\sum_{i=1}^{m} n_i$$

partial solutions if multiple graphs are employed.

4. Control of Inferences

The power of a problem solving procedure depends significantly on its control mechanism. In order to solve difficult problems it is paramount to find general and domain specific heuristic knowledge and to represent this knowledge in the problem solving system. The smaller the class of control strategies admissible for an inference system, the tighter its limitations in taking full advantage of heuristic knowledge. Inference systems supporting some kind of planning may provide for strong search space reductions. Furthermore inference systems can be improved by integrating preprocessors and postprocessors simplifying the given problem and generalizing partial solutions, respectively.

Strategies, heuristic control, planning, preprocessors and postprocessors require information which must be explicitly represented by the inference system otherwise they have nothing to operate upon. In particular the states of the inference system should be richly structured to provide immediate access to such information.

In this section we give an impression of the impact of the explicit representation of information contained in our inference system.

4.1. Weights and Limits

Heuristics for inference systems based on the subterm replacement principle (e.g. paramodulation, demodulation) are usually computed from the information implicit in a term (literal, clause) [WOL84]. For example weight, ordering, size, nesting depth etc are defined and are often based on certain symbol occurrences, like the number of different variables in a literal, etc. The main purpose of these heuristics is to control and reduce the complexity in terms of the defined measure.

Since the information of terms is available, this class of heuristics is also applicable in methods based on the difference reduction principle [Di85]. Hence such heuristics are applicable in ECOP too.

4.2. Difference reduction

Equality reasoning systems based on the difference reduction principle compare two terms. A measure for the difference of two terms can be defined and heuristics can be developed to control the reduction of these differences. Just like the operators in GPS [NS59] they estimate which equations could best reduce these differences. As before the difference measure may depend on term structures and symbol occurrences. In contrast to the previous heuristics based on weights and limits, difference reduction heuristics not only restrict term sizes etc. but control the distance between terms, i.e. are directed towards the goal to make two terms equal.

Heuristics of this type are for example used by Digricoli [Di85] who defines a heuristic ordering by degree of unification. His ordering depends on the equality of the toplevel function symbols and on the number of unifiable subterms.

The heuristics and preprocessors applicable in difference reduction methods are also applicable in our method. In the example of section 2.2 the subproblem $h(x\ b)$ - - - - $h(b\ a)$ has to be solved. Since the term b occurs in both terms at the first subterm level but in different positions, a heuristic should be available proposing the insertion of an equality-chain between these terms at toplevel: if possible a commutativity axiom, instead of unifying the corresponding pairs of subterms.

In our system we integrated two preprocessors: one tests whether both terms under consideration can be made equal with respect to its symbol occurrences. If this test fails, no further operation need be performed because of the unsolvability of the subproblem. The second preprocessor matches the given subproblem with the equations derived so far in order to find a quick solution.

It is more difficult to insert heuristics and preprocessors as explained in this section into subterm replacement inference systems, since the information about the goal is not available in the calculus.

4.3. Solving Conflicts

In our method the global links and the structure of the graph provide the information for heuristics, which control the selection of subproblems and the subgraph replacements. Hence heuristic planning from a global point of view becomes possible.

In the example of section 3.1 a conflict occurs in the attempt to assign two different terms $f(a\ a)$ and a to the variable x. By subgraph replacement at one of the occurrences of the variable x other partial solutions can be derived possibly leading to compatible unifiers. It seems to be more likely to find new partial solutions at the position where a was assigned to x (such that the new assignment to x is extended to a term with leading function symbol f), than to find new partial solutions yielding the constant a or a variable instead of $f(a\ a)$ at the appropriate position. Therefore in our example a heuristic should be available to prefer the insertion of an equation-chain between $f(a\ z)$ and $f(x\ a)$, such that after the insertion x is assigned to a term with topsymbol f:

$$f(a\ z) \text{ --- } f(\,.\,.\,) \equiv \dots \equiv f(\underbrace{f(\,.\,.\,)\,.\,) \text{ --- } f(x}_{}\ a).$$

The link indicates the condition which should be used by the heuristic guiding the subgraph replacement. Hence a partial solution assigning a to x is to be replaced by one assigning a term $f(\,.\,.\,)$ to x which is more likely to be unifiable with $f(a\ a)$ in order to get two compatible partial solutions.

To sum up most equality reasoning systems based on the principle of subterm replacement support only heuristics concerning weights and limits, whereas the inference systems based on the difference reduction principle also support heuristics guiding the difference reduction of two terms.

In our method, the states of the inference system have a structure rich enough to incorporate both kinds of heuristics. Above all heuristics can be incorporated supporting the planning of the search for a proof from a global point of view. Such heuristics require global infomation about the correlation of partial solutions, which is explicitly represented in the graph structure. Thus it is possible to develop and integrate a wide varity of heuristics at different levels of a problem solving process.

Heuristics concerning weights and limits are well examined in the field. Unfortunately that is not the case for the other classes of heuristics, although they seem to be more powerful.

5. Evaluation

5.1. Implementation

The equality graph construction method has been implemented as an independent equality prover, currently integration into the "Markgraf Karl Refutation Procedure" [KM84] is under way. The implementation includes some pre- and postprocessors as described in [Bl85].

Only a preliminary version of the control component has been implemented at the present time. Subgraphs are selected rather arbitrarily and most of the applicable rules to insert equation chains are applied at once producing several alternative equality graphs. The integration of heuristics as explained in section 4 has as yet not been exploited.

Although no refined selection exists at the moment, the experimental results with the system are promising: The standard test problems in the field were easily solved by the ECOP procedure (see [Bl85]).

5.2. Comparison with other Approaches

5.2.1. Graph Procedures

Most theorem proving procedures based on graphs have the advantage of being independent of the order of the execution sequence: For one compatible equality graph there exist in general many paramodulation sequences performing the same steps in a different order. For the example $\langle f(a\ b) =_E f(c\ d) \rangle$, $E = \{a = c, b = d\}$ there exists only one compatible equality graph:

$$\begin{tikzcd}
f(a \arrow[d, dash] & b) \arrow[d, dash] \\
a \equiv c \arrow[d, dash] & b \equiv d \arrow[d, dash] \\
f(c & d)
\end{tikzcd}$$

but the paramodulation steps
$$\begin{tikzcd}
 & f(c\,b) \arrow[r] & f(c\,d) \\
f(a\,b) \arrow[ur, dash] \arrow[dr, dash] & & \\
 & f(a\,d) \arrow[r] & f(c\,d)
\end{tikzcd}$$
are possible.

Like other graph based theorem proving procedures more information has to be stored and hence more memory is required as compared to non graph based methods. This disadvantage is compensated by a smaller search space, which is particularily important when the system is applied to more complex problems.

5.2.2. Subterm Replacement versus Difference Reduction

Although subterm replacement is a very natural operation, the difference reduction principle seems to be more powerful for computational equality reasoning.

As demonstrated in section 4 difference reduction based methods represent more information in the states of the inference system. Hence they support the development of proper heuristics, which direct the deduction process towards the goal of making two terms equal. Most of the heuristics based on weights and orderings are also applicable in difference reduction based systems, and in domains where term rewriting systems have been sucessful, such reduction systems can be used as preprocessors.

Using some small examples we compare paramodulation with our method:

In ECOP only inferences are allowed which reduce the difference of two terms. For example in $\langle f(a\,b) \equiv_E f(d\,b) \rangle$, $E = \{a \equiv c\}$ the term $f(a\,b)$ can be paramodulated by $a \equiv c$ with the result $f(c\,b)$, but in difference reduction based systems no operation is possible, because using the equation $a \equiv c$ would not reduce the difference to $f(d\,b)$.

In ECOP it is not allowed to insert equations between a variable and a term, unless the term contains the variable as a subterm. This restriction is much stronger, than analogous restrictions in paramodulation procedures, which prevent paramodulation into variables, because

1) the restriction is effective at both sides of an equality problem as in the following example: $\langle f(a) \equiv_E f(x) \rangle$, $E = \{a \equiv b\}$. The equation $a \equiv b$ may be paramodulated into $f(a)$ but the insertion of the equation $a \equiv b$ between $f(a)$ and $f(x)$ is forbidden.
2) the restriction is also effective for all equations involved as demonstrated by $\langle f(a) \equiv_E g(y) \rangle$, $E = \{f(x) \equiv g(x),\ a \equiv b\}$:

The paramodulation sequences
$$\begin{tikzcd} & f(b) \arrow[r] & g(b) \\ f(a) \arrow[ur, no head] \arrow[dr, no head] & & \\ & g(a) \arrow[r] & g(b) \end{tikzcd}$$

are derivable, i.e. after paramodulation with $f(x) \equiv g(x)$ the variable x is instantiated and paramodulation at this position is possible. In our method however the only solution is

$$\begin{tikzcd} f(a) \arrow[d, no head, "/"'] \\ f(x) \equiv g(x) \arrow[d, no head, "/"] \\ g(y) \end{tikzcd}$$

and the insertion of $a \equiv b$ is not allowed at any position.

5.2.3. Subterm Replacement and Difference Reduction versus ECOP

The main advantage of the equality graph construction method is the possibility of subgraph replacement and OR-branch extension. Local modifications in the graph can be controlled by global planning, for example partial solutions which are not compatible with other partial solutions when viewed from a more global level, can be detected and replaced or modified. Local modifications in the graph do not disturb the global plan and do not render other partial solutions useless. Expressed conversely, one bad operation does not make subsequent good operations at other positions of the same structure worthless, as is the case in many other equality reasoning methods.

Consider the example of section 3.1 with the equality problem $\langle\, h(f(a\ z)) \equiv_E g(k(y\ c)\ f(a\ a))\,\rangle$. The following paramodulation sequence may be derived:

$$\begin{array}{lll} h(f(a\ z)) \ +\ h(f(x\ a)) \equiv g(h(x)\ x) & \longrightarrow & g(h(a)\ a) \\ g(h(a)\ a) \ +\ h(u) \equiv k(u\ b) & \longrightarrow & g(k(a\ b)\ a) \\ g(k(a\ b)\ a) \ +\ b \equiv c & \longrightarrow & g(k(a\ c)\ a) \end{array}$$

With regard to the given equality problem this paramodulation sequence seems to be desirable and well directed towards the goal of making the terms $h(f(a\ z))$ and $g(k(y\ c)\ f(a\ a))$ equal. The difference between both terms is reduced with each step, under the assumption of an appropriate measure for the difference of two terms.

But although the difference between $g(k(a\ c)\ a)$ and $g(k(y\ c)\ f(a\ a))$ is relatively small, this difference cannot be removed by the given equations. No continuation of the stated paramodulation sequence can lead to the final goal, only backtracking and the choice of another step can help.

If after backtracking the initial term $h(f(a\ z))$ is first paramodulated with $f(u\ f(v\ w)) \equiv$

f(f(u v) w) yielding h(f(f(a v) w)) then the same paramodulation sequence as before must be searched for and executed again (a well known disadvantage of all blind backtracking methods, which led to the development of nondependency directed backtracking and reason maintenance systems):

$$\begin{array}{lcl} h(f(a\ z)) + f(u\ f(v\ w)) \equiv f(f(u\ v)\ w) & \rightarrow & h(f(f(a\ v)\ w)) \\ h(f(f(a\ v)\ w)) + h(f(x\ a)) \equiv g(h(x)\ x) & \rightarrow & g(h(f(a\ v))\ f(a\ v)) \\ g(h(f(a\ v))\ f(a\ v)) + h(u) \equiv k(u\ b) & \rightarrow & g(k(f(a\ v)\ b)\ f(a\ v)) \\ g(k(f(a\ v)\ b)\ f(a\ v)) + b \equiv c & \rightarrow & g(k(f(a\ v)\ c)\ f(a\ v)) \end{array}$$

After one bad or one missing paramodulation step the subsequent steps might be worthless although they are just the right steps to solve subproblems which are encountered: Partial solutions cannot be combined arbitrarily.

The effect that partial solutions can be lost is not only a property of the subterm replacement methods, but also occurs in reasoning based on difference reduction: In the method of Digricoli for example an application of the RUE-rule can result in the clause h(a) ≠ k(y c), a ≠ f(a a). All operations to eliminate the first literal are worthless and cannot be reused after the choice of another disagreement set or another substitution yielding h(f(a a)) ≠ k(y c) , f(a v) ≠ f(a a) for example. The same operations as before must be searched for and performed in order to remove the literal h(f(a a)) ≠ k(y c).

The immediate application of unifiers to other subproblems produces completely different versions of the subproblem $\langle h(x) \equiv_E k(y\ c) \rangle$ that is $\langle h(a) \equiv_E k(y\ c) \rangle$ and $\langle h(f(a\ a)) \equiv_E k(y\ c) \rangle$. In both paramodulation and RUE-resolution the solution of the second version of the subproblem cannot be reduced to the first version but must be recomputed, again. That is, it cannot be realized, that the solution of $\langle h(x) \equiv_E k(y\ c) \rangle$ need not refer to the variable x.

In ECOP we have more global information in the graph, which supports the planning of the proof. Subgraph replacements or extensions by OR-branches are allowed to solve conflicts, without losing the partial solutions already found in other parts of the graph.

The lack of the possibility for repair might be the reason why Digricoli as well as Lim and Henschen allow instantiations in their methods not determined by unification or partial unification (see [Di79] page 46, [Di85] and [LH85]).

5.2.4. Anti Waltz Effect

Like other applications in artificial intelligence equality reasoning procedures are confronted with large search spaces.

For the problem of interpreting line drawings as geometrical objects, Waltz proposed a method of exploiting local constraints to keep the search space under control. The great success of his method was surprising and is founded on the fact, that combinations of a few elementary interpretations are in most cases very early detected as incompatible, hence the combinatorial explosion is minimized or even eliminated. In other words if n elements are to be combined, in most cases the incompatible combinations are already detected when only two, three, or four of its elements are combined.

Unfortunately in equality reasoning the opposite happens. Due to the many variables that generally occur in the formulas, combinations of two, three or four operations are compatible in most cases. If n elements are to be combined, the incompatibility is in most cases not detected until at least n-1, n-2, or n-3 operations are combined. This effect is the main reason why our earlier approach for equality reasoning [B183] based on constrain satisfaction methods similar to Waltz failed. This "Anti Waltz Effect" as well as the potential to operate on every subterm of each formula seems to be the main reason for the crux caused by equational axioms in automatic theorem proving.

The search space explodes because many combinations of possible operations are compatible. Incompatibilities are detected far too late. The search performed until detection of an incompatibility is useless in most cases, even if partial problems have been solved. After backtracking the partial problems must be solved again.

In order to work against the "Anti Waltz Effect" we try to solve incompatibilities by repairing the graph through subgraph replacement or OR-branch extension. This retains the existing plan and does not destroy other partial solutions. The graph and thus the inference system contains the information necessary to perform such repairs, which is again very much in the spirit of reason maintenance systems and nondependency directed backtracking methods.

5.2.5. Derivability of Unifiers

The equality graph construction method presented in this paper overcomes the necessity to derive all different unifiers as is the case in other equality reasoning methods.

The exclusion of equality-chains between variables and terms restricts the derivability of unifiers for the given problem: only one of the class of all unifiers equal under E is derivable.

Due to the introduction of OR-branches representing alternative solutions for the same problem each partial solution is derivable at most once.

Strategies can be incorporated restricting the derivability of unifiers, in such a way that after a subproblem has been solved, no other solutions, i.e. other unifiers for this subproblem are derivable, until a conflict in the unification of substitutions occurs in a global environment. This restriction also involves quite different unifiers which are not unifiable under E. Only when the unification of two substitutions fails, another solution for one of the subproblems causing the conflict is derivable and inserted into the hitherto existing graph by subgraph replacement or OR-branch extension.

The restriction of the derivability of unifiers by means of globally planned graph operations is comparable to the restriction of the derivability of instances through the introduction of unification.

6. Applicability

In contrast to methods like E-Resolution or RUE-Resolution our method is not a uniform proof procedure for first order predicate calculus with equality. It is much more like an independent universal unification procedure. Hence it can be used in all problem solving systems in which unification and equality are important. Since ECOP is not a decision procedure (like most ordinary unification algorithms), the incorporation into a theorem proving procedure like the Markgraf Karl Refutation Procedure [KM84] requires a mechanism to control the co-operation between equality reasoning and other tasks by a heuristic allocation of resources.

For the integration of ECOP into the MKRP-system we propose to use E-resolution [Mo69] as the inference rule and ECOP to search for possible E-resolution steps. An E-resolution step can be regarded as a sequence of paramodulation steps such that two potentially complementary unifiable literals become unifiable, followed by the appropriate resolution step. Paramodulation and resolution concern at most two clauses, whereas E-resolution is a generalization of resolution concerning many clauses. To integrate E-resolution into clause graphs new structures are required: ER-paths (E-resolution-paths) are defined to represent possible E-resolution steps. An ER-path connects two potentially complementary literals and the equations which make both literals unifiable.

In an extended clause graph proof procedure ER-paths should also be searched for and created at the beginning, when the initial graph is formed. This information should then be inherited during the subsequent search for a proof. But because of the undecidability of equality of two terms not all the necessary ER-paths can be found in the initial graph. For that reason a new link type called PER-link (potential E-resolution link) is introduced into the graph connecting potentially complementary unifiable literals (with same predicate and opposite sign). These PER-links provide the top-level information for the proof procedure to search for the corresponding ER-paths.

If an ER-path is selected during the derivaton, then the E-resolution corresponding to the ER-path is executed and if a PER-link is selected the search for ER-paths is continued, calling ECOP. If ECOP determines the unsolvability of a given equality problem then the corresponding PER-link is deleted. During the search for a proof ER-paths as well as PER-links are inherited and used to compute new ER-paths and new PER-links.

Conclusion

Three points can be stated concerning further development of existing equality reasoning methods:

1. Equality graph construction gives full support for the global planning of the whole

proof and supplies the information for a wide varity of heuristics.

2. The Anti Waltz Effect in equality reasoning makes standard constraint satisfaction methods unfeasable. Instead of this a repair facility is integrated into the inference system to solve conflicts of compatibility by subgraph replacement or extension.
3. Similar to the lifting effect of unification which restricts the derivability of instances, the derivability of unifiers is restricted by our method. This reduces the search space drastically.

Future research should investigate the development of heuristics for all three categories, further examine the problems of subgraph replacement, and should allow for conditional equations.

ACKNOWLEDGEMENT

I would like to thank Norbert Eisinger, Jörg Siekmann and Graham Wrightson for their advice in the preparation of this work.
This work was supported by "Sonderforschungsbereich 314 (Künstliche Intelligenz) der Deutschen Forschungsgemeinschaft".

REFERENCES

[Bl83] K.H. Bläsius: Equality-Reasoning in clause graphs. Proc. IJCAI, 936-939, 1983

[Bl85] K.H. Bläsius: Equality-Reasoning with Equality-paths.
Proc. German Workshop on Artificial Intelligence, 1985

[Bl86] K.H. Bläsius: Correctness and Completeness of the equality graph construction method. (forthcoming)

[Br75] D. Brand: Proving Theorems with the Modification Method.
SIAM Journal of Comp., vol 4, No. 4, 1975

[Bu83] A. Bundy: The Computer Modelling of Mathematical Reasoning.
Academic Press, London, 1983

[Di79] V.J. Digricoli: Resolution by Unification and Equality.
Proc. 4th Workshop on Automated Deduction, Texas, 1979

[Di85] V.J. Digricoli: The Management of Heuristic Search in Boolean Experiments with RUE Resolution. Proc. IJCAI-85, Los Angeles, 1985

[HR78] M.C. Harrison, N. Rubin: Another Generalization of Resolution.
JACM, vol 25, no. 3, July 1978

[Hu76] G. Huet: Resolution d'equations dans des languages d'ordere 1,2,...,ω
These d'Etat, Univ. de Paris, VII, 1976

[Hul80] J.M. Hullot: Canonical Forms and Unification. Proc. 5th Workshop on Automated Deduction, Springer Lecture Notes, vol. 87, 318-334, 1980

[JK83] J.P. Jouannaud, C. Kirchner, H. Kirchner: Incremental Construction of Unification Algorithms in Equational Theories. Proc. of the Intern. Conf. on Automata, Languages and Programming, Springer Lecture Notes, vol. 154, 361-373, 1983

[KM84] Karl Mark G. Raph: The Markgraf Karl Refutation Procedure.Interner Bericht, Memo-Seki-MK-84-01, Fachbereich Informatik, Universität Kaiserslautern, 1984

[Ko75] R. Kowalski: A Proof Procedure Using Connection Graphs. JACM 22, 4, 1975

[LH85] Y. Lim, L.J. Henschen: A New Hyperparamodulation Strategy for the Equality Relation. Proc. IJCAI-85, Los Angeles, 1985

[Mo69] J.B. Morris: E-resolution: An Extension of Resolution to include the Equality Relation. Proc. IJCAI, 287-294, 1969

[NS59] A. Newell, J.C. Shaw, H. Simon: Report on a General Problem Solving Program. Proc. Int. Conf. Information Processing (UNESCO). Paris, 1959

[Pl72] G. Plotkin: Building in Equational Theories. Machine Intelligence, vol. 7, 1972

[Pl81] D. Plaisted: Theorem Proving with Abstraction. Artifical Intelligence 16, 47-108, 1981

[Ro65] J.A. Robinson: A Machine-Oriented Logic Based on the Resolution Principle. JACM 12, 1965

[RW69] G. Robinson, L. Wos: Paramodulation and TP in first order theories with equality. Machine Intelligence 4, 135-150, 1969

[Sh78] R.E. Shostak: An Algorithm for Reasoning About Equality. CACM, vol 21, no. 7, 1978

[Si69] E.E. Sibert: A machine-oriented Logic incorporating the Equality Axiom. Machine Intelligence, vol 4, 103-133, 1969

[Si84] J. Siekmann: Universal Unification. Proc. of Conf. on Autom. Deduction, Springer Lecture Notes, vol. 170, 1984

[SS81] J. Siekmann, P. Szabo: Universal Unification and Regular ACFM Theories. Proc. IJCAI-81, Vancouver, 1981

[SW79] J. Siekmann, G. Wrightson: Paramodulated Connectiongraphs. Acta Informatica, 1979

[Sz82] P. Szabó: Unifikationstheorie erster Ordnung. Dissertation, Universität Karlsruhe, 1982

[WRC67] L. Wos, G. Robinson, D. Carson, L. Shalla: The Concept of Demodulation in Theorem Proving. J. ACM 14, 698 - 709, 1967

[WOL84] L. Wos, R. Overbeek, E. Lusk, J. Boyle: Automated Reasoning, Introduction and Applications. Pentice Hall, 1984