### EQUALITY REASONING IN CLAUSE GRAPHS

#### Karl-Hans Blasius

# Institut fur Informatik I, University of Karlsruhe

#### Abstract

A method to control the application of equality derivation rules in an automatic theorem proving system is presented. When handling equality in the search for a proof of a theorem two main problems arise:

- to obtain a control mechanism for the search and application of useful equality derivation steps in order to support global strategies which plan and control the whole proof, thus conducing to an efficient and complete proof procedure.
- 2) to find proper equations rendering two terms unifiable.

These problems are solved by combining the clause graph method and the Mparamodulation-if-needed" idea by introducing Morris' E-resolution into the clause graph proof procedure. The necessary equations to form possible E-resolvents are searched for in the initial graph and are inherited afterwards. The search space for possible E-resolutions will be reduced by exploiting constraints using the information in the clause graph.

# 0. INTRODUCTION

The use of equality axioms in a theorem prover based on the resolution principle has turned out to be very inefficient because too many additional resolution operations involving the equality axioms are possible. This problem is well recognized in the field (see [RW69], [Si69], [Mo69], [Br75], [Sh78], [HR78L [Di79]).

A way out is the direct incorporation of equality into the proof procedure. One of the various methods proposed with this aim in mind is paramodulation [RW69]: with one additional rule of inference, the paramodulation rule, the equality axioms become superfluous except for the reflexivity axion. But paramodulation, i.e. the replacement of terms by equal terms, can be applied almost everywhere in a clause set and therefore paramodulation alone still does not solve the problem of "how to handle equality in an automatic theorem proving system" (ATP).

Strategies or methods are required to control the enomous amount of potential steps and to make sensible use of the paramodulation rule. A promising control mechanism may result from the "paramodulation if needed" idea, which states that the paramodulation rule should only be used to reduce differences between potentially complementary unifiable literals, such that an inference step by resolution becomes possible. Two literals (in diffe-

rent clauses) are called <u>potentially complementary</u> <u>unifiable</u>, if they have the same predicate symbol and opposite sign.

There are several methods known to realize the "if needed" idea (e.g. [Sh78], [HR78], [Di79]) the most explicit realization of which is Morris' E-resolution [Mo69]. 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. This could be an optimal realization of the "if needed" idea and potentially one of the bestways of handling equality in an ATP because equations are only used when needed, i.e. to remove the differences between terms which prevent a desired resolution step. Furthermore the equations are only used if it is possible to remove such differences completely.

An implementation of a proof procedure based on E-resolution however is unfeasable without additional search and control mechanisms because two main problems remain:

- Equality of two terms with respect to the given set of equations is in general undecidable and therefore it is generally impossible to continue searching for equations until the potentially complementary unifiable literals under consideration are unifiable (or definitely not unifiable). Hence the first problem is to organize the search for equations and the application of E-resolution in such a way that the proof procedure is efficient and complete. The proof procedure based on E-resolution presented by Morris [Mo69] is designed to ensure completeness [An7o] rather than efficiency.
- 2) Before an E-resolution step can be executed the necessary equations <u>have to be found</u> and an unsophisticated and <u>exhaustive search</u> for such equations is prohibitively expensive because of the enormous search space.

In this paper we are mainly interested in the problems of an efficient search by using ideas from the paradigm of the connection graph proof procedure [Ko75].

The <u>Connection Graph Procedure</u> introduced by Kowalski in 1975 [KO75] represents all possible resolution steps by links (connections) between the complementary unifiable literals. Initially, all possible links are created to build the <u>initial</u>

graph, but afterwards the new links are inherited during the search for a proof. Hence there is no additional search for the potential application of the resolution rule since all the necessary information is present in the graph structure, i.e. in the links. The executed operations are remembered by deleting the corresponding link and powerful deletion rules are known to reduce the clause graph. In [SW79] the ideas of the connection graph proof procedure are extended to handle paramodulation ac well: in the paramodulated clause graph procedure (PCG-procedure), P-links (paramodulation links) are defined to represent possible steps by paramodulation and R-links (resolution-links) represent possible steps by resolution.

In this paper the ideas of the "paramodulation-if-needed" paradigm and the ideas of the PCG-procedure are combined essentially by incorporating E-resolution into the PCG-procedure. This combination provides an efficient search procedure for possible E-resolutions and a control mechanism to direct the search for E-resolutions thus solving the two problems 1) and 2) mentioned above.

Paramodulation and resolution concern, at most, two clauses, whereas E-resolution is a generalization of resolution concerning many clauses, therefore new structures, called paths, will be used to represent such macro-operations. A path represents a sequence of p-links and contains the necessary information to perform the respective macro-operation. During the search for a proof it is possible to select links as well as paths for the derivation of new clauses. In [B183] ER-paths (E-£esolution-paths) representing possible E-resolution steps are defined. An ER-path connects two potentially complementary unifiable literals and the equations which make both literals unifiable.

The above stated problems of E-resolution may then be rephrased as:

- 1) the integration of ER-paths into the proof procedure
- 2) the problem of searching for ER-paths.

All notions used but not defined in this paper are consistent with [Ko75], [Lo78], [SW79] (e.g. Clause, Resolution, Connection Graph Proof Procedure, Link).

# 1. INTEGRATION OF PATHS INTO THE PROOF PROCEDURE

Just like R-links and P-links 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, <u>PER-link</u> (potential E\_-resolution link) connecting potentially complementary unifiable literals is introduced into the graph. These PER-links provide the top-level information for the proof procedure to search for the corresponding

ER-paths, and during this search three cases can occur:

- a) All possible ER-paths corresponding to a PER-link are found.
- b) It is detected that no ER-path for a given PERlink exists.
- c) The search for ER-paths is stopped because of space or time limitation.

For each of these cases different operations are performed:

- a) If all ER-paths are found, then the PER-link is replaced by the ER-paths. All possible ways to remove the differences between the literals of the PER-link are known and given by the ER-paths. If an ER-path is selected in the proof, then the E-resolution corresponding to the ER-path is executed.
- b) There exists no ER-path between the literals of the PER-link, i.e. the differences of the respective terms cannot be removed completely and therefore the PER-link is deleted.
- c) If the search is stopped and no (or not all) ER-paths are found, then the PER-link remains in the graph. If a PER-link is selected, then an operation to reduce the differences between the literals of the PER-link is executed using the information obtained from the incomplete search.

PER-links are treated as a special case and are only used in global strategies as auxiliary links which control the search for paths and enable sensible operations to reduce differences of potentially complementary unifiable literals as compensation for the lacking ER-paths (case c above).

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. Hence each ER-path used in the proof was already present in the initial graph or has been inherited from an initial ER-path.

This method of using ER-paths has three main advantages:

- (i) ER-paths and PER-links contain important information for global strategies which plan and control the whole search for a proof (see [BESSHW81], [SS81], [oh82]). This information is particularly useful because it is already available at the beginning of the proof.
- (ii) An efficient handling of equality is possible, because the search is done in the <u>small ini-tial graph</u> only. Furthermore only a few paths have to be searched for; most paths are found by inheritance.
- (iii) Strong additional <u>deletion rules</u> are available which greatly reduce the potential number of paths as well as the size of the graph. For example in case b) above the deletion of a PER-link may lead to a pure clause just as for the usual clause graph proof procedure,

thus causing the well-known snowball effect of deletions to start.

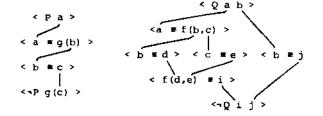
#### 2. THE SEARCH FOR PATHS USING CONSTRAINTS

An unsophisticated and exhaustive search for ER-paths is prohibitively expensive because of the very large search space. The size of this essentially depends on the number of equations (ne) in the clause set, on the average number of subterms (ns) in a literal, and on the ratio (r) of positive to total unification tests for possible equality replacements. Finally it also depends on the search depth (sd). A breadth first search procedure, like that of Morris [Mo69] must execute ((2 \* ne \* ns \* r) \* 1/r) calls of the unification algorithm. Also it has to carry out (2 \* ne \* ns \* r) \* equality replacements in order to get all possible E-resolvents for one PER-link. For example, if a clause set has ten equations, an average of five subterms per literal and a unification success rate of 10%, then a search depth of ten steps requires 10 calls of the

unification algorithm and 10 replacement operations. To make the situation worse Morris [Mo69] observed that in usual clause sets for most PER-links there does not exist an E-resolvent, but since the search must be done completely most of the effort is spent in vain.

In order to obtain a practicable search procedure the enormous search space has to be reduced. In the following we propose a reduction of the search space by <u>exploiting constraints</u>. The constraints use the information contained in the P-links.

For a clause graph  $\Leftrightarrow$  and a PER-link there are usually many P-links connecting the effective equations and the two literals. If these P-links are visualized graphically, certain structures emerge, e.g. an equality-chain in the simple example 1. But in most cases the structure is a more complex equality-net as example 2 demonstrates:

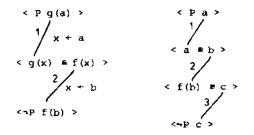


Example 1 Example 2

An <u>equality-net</u> (EN) and the <u>compatibility</u> of an equality-net are defined in [B183] as a special graph structure of P-links. Each compatible equality-net builds an ER-path and represents a possible E-resolution step. Conversely there exists for each possible E-resolution a compatible EN in the graph which builds the appropriate ER-path. These conditions are ensured because the PCG-Procedure is complete [SW79]. In order to find the ER-paths the compatible equality-nets must now be searched for.

To this end several <u>conditions</u> can be stated which are <u>necessary but not sufficient</u> for the compatibility of an EN. Such constraints are checked step by step thereby drastically reducing the search space.

Two examples will show possible reasons for the incompatibility of equality-nets.



Example 3

Example 4

In Example 3 the combination of the P-links 1 and 2 is impossible because their unifiers  $[x \leftarrow a]$  and  $[x \leftarrow b]$  are incompatible. In example 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 depth does not coincide.

Some of the constraints to be used are:

- In a compatible EN all unifiers of the p-links concerned must merge to one mgu (i.e. the unifiers must be compatible, provided a proper variable renaming has been carried out).
- 2) In a compatible EN: for each maximal equalitychain in the EN the sum of all access depths must be equal to zero and each partial sum must be less than or equal to zero.
- 3) In a compatible EN, no chain of two links in the EN is incompatible, i.e. if an equality-chain of length 2 is incompatible, then each EN which contains this equality-chain is also incompatible. (Note: incompatible is defined in [B183] and is different from not compatible.)

The search procedure can be improved by introducing additional constraints.

Although such constraints are powerful it is not expensive to test them, for example only some integers must be added to detect the incompatibility of an EN caused by condition 2.

After reduction of the search space the most promising ENs are selected for a full compatibility test and the surviving candidates are used to build the ER-paths.

The method outlined here of handling equality is described in [B183] where the formal notions are developed and additional results are presented. This method is presently being implemented and evaluated as part of the Markgraf Karl Refutation Procedure [BESSHW81], [Oh82].

# [SS81] **LITERATUR** [An7o] R. Anderson: Completeness results for E-Resolution. Proc. Spring Joint Conf., 1970, 653-656 [BESSHW81] K.H. Blasius, N. Eisinger, J. Siekmann, G. Smolka, A. Herold, C. Walther: The Markgraf Karl Refutation Procedure. Proc. IJCAI, 1981, 511-518 [B183] K.H. Blasius: Equality-Reasoning in clause graphs. Universitat Karlsruhe, Interner Bericht (to appear) [Br75] D. Brand: Proving Theorems with the Modification Method. SIAM Journal of Comp., vol 4, no. 4, 1975 [Di79] V.J. Digricoli: Resolution by Unification and Equality. Proc. 4th Workshop on Automated Deduction, 1979, Texas [HR78] M.C. Harrison, N. Rubin: Another Generalization of Resolution. JACM, vol 25, no. 3, July 1978 [Ko75] R. Kowalski: A Proof Procedure Using Connection Graphs. JACM 22, 4, 1975 [LO78] D. Loveland: Automated Theorem Proving: A logical Basis. North Holland, 1978 [Mo69] J.B. Morris: E-Resolution: An Extension of Resolution to include the Equality Relation. Proc. IJCAI, 1969, 287-294 [0h82] H.J. Ohlbach: The Markgraf Karl Refutation Procedure, the Logic Engine. Universitat Karlsruhe, Interner Bericht, 1982 [RW69] G. Robinson, L. Wos: Paramodulation and TP in first order theories with equality. Machine Intelligence 4, 135-150 [Sh78] R.E. Shostak: An Algorithm for Reasoning About Equality. CACM, July 1978, vol 21, no. 7

[S169]

E.E. Sibert:

103-133

the Equality Axiom.

A machine-oriented Logic incorporating

Machine Intelligence, vol 4, 1969,

J. Siekmann, G. Smolka:
Selection Heuristics, Deletion
Strategies and N-Level Terminator
Configurations for the Connection
Graph Proof Procedure.
Universitat Karlsruhe, Research Report,
1981

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