AN ALGORITHM FOR REASONING ABOUT EQUALITY

Robert E. Shostak
Stanford Research Institute
Menlo Park, California 94025

I Introduction

A deductive system for program verification must be able to reason proficiently about equality. Equality is often handled in an ad hoc and incomplete way—most usually with a rewrite rule that substitutes equals for equals with some heuristic guidance. We present a simple algorithm for reasoning about equality that is fast, complete, and useful in a variety of theorem-proving situations. We also present a proof of the theorem on which the algorithm is based.

II An Example

The following formula is of the kind one encounters in verifying programs involving array indexing:

$$(I=J \wedge K=L \wedge A[I]=B[K] \wedge J=A[J] \wedge M=B[L]) \supset A[M]=B[K]).$$

Here, A and B are function symbols while I,...M are universally quantified variables.

One might approach such a formula by working backwards from the conclusion, substituting equals for equals until the left-hand-side is transformed into the right-and-side:

$$A[M]=A[B[L]]=A[B[K]]=A[A[I]]=A[A[J]]=A[J]=A[I]=B[K]$$

One could also work from BfK] rather than from AfM"], or from both simultaneously; the links needed in the chain are the same in either case.

While this "backward substitution" method and other methods that transform formulas through a sequence of substitutions are logically sound, they are not well-suited to machine deduction because there is no easy way of selecting the right substitution to make at each step.

Intuitively, it would not seem necessary to generate terms beyond a certain depth. However, the critical depth (the smallest depth necessary to consider) cannot be calculated solely as a function of the depths of the terms appearing in the original formula; in particular, the critical depth is not simply the maximum of these depths. For example, no backward-substitution proof can be carried out for the above formula without generating a term of at least depth 3. Even if one could conveniently calculate the critical depth, one would still, in general, generate many more terms than are necessary.

This difficulty with substitution transformation methods is not inherent in the problem. The next section presents a more efficient method that considers only the terms appearing in the original formula.

III The Procedure

Our method is a decision procedure for the subclass of predicate calculus with function symbols and equality whose formulas have only universal quantifiers in prenex form. While the decidability of this subclass is well-known, the classical decision procedure for it (W. Ackermann [1]) produces a combinational explosion that makes that method infeasible for non-trivial problems.

The procedure is as follows. The matrix of the formula F is first negated and placed in disjunctive normal form. Next, all atomic formulas other than equalities are replaced by equalities as follows. For each n-ary predicate symbol P_i^n occurring in the formula, a new n-ary function symbol f_i^n is introduced. Each atomic formula $P_i^n(t_1, t_2, \ldots t_n)$ occurring in the formula is then replaced by the equality $f_i^n(t_1, t_2..., t_n) = c$, where c is a constant. The modified d.n.f. is intersatisfiable with the original one, and is satisfiable if and only if one of its disjuncts is satisfiable. Each disjunct, moreover, consists of a conjunction of equalities and negations of equalities. The problem is thus reduced to testing the satisfiability of each such conjunction.

It remains to test each conjunction for satisfiability. Let S be the set of equalities and negations of equalities occurring in the conjunction to be tested. Let T be the set of terms and subterms of terms occurring in S, and define the binary relation $\dot{=}$ as the smallest relation over T x T (where $u_1, u_2 \ldots u_n$, t_1, t_2, v_1, $v_2 \ldots v_n$ denote terms and f denotes a function symbol) that:
(1) Contains all pairs $\langle t_1, t_2 \rangle$ for which $'t_1 = t_2' \in S$
(2) Is reflexive, symmetric, and transitive
(3) Contains the pair $\langle f(u_1, u_2 \ldots u_r), f(v_1, v_2, \ldots v_r) \rangle$ whenever it contains the pairs $\langle u_i, v_i \rangle$, $1 \leq i \leq r$, and $f(u_1, u_2 \ldots u_r)$, $f(v_1, v_2 \ldots v_r)$ are both in T.

The test for satisfiability of S depends on the following theorem (to be proved later):

Theorem. S is unsatisfiable $\Leftrightarrow$ there exist terms t_1, $t_2 \in T$ such that $'t_1 \neq t_2' \in S$ and $t_1 \dot{=} t_2$.

The theorem tells us that to determine the satisfiability of S it suffices to consider the negated equalities of S one at a time. If one is found (say $t_1 \neq t_2$) for which $t_1 \dot{=} t_2$, S is unsatisfiable; otherwise S is satisfiable. Note that the definition of $\dot{=}$ involves only terms in T.

To use the theorem, we must be able to calculate whether a given pair of terms is in the relation $\dot{=}$. This can be done by building the relation from the definition: (1) is used as a basis, and (2) and (3) are repeatedly applied until no new terms are generated. Since $\dot{=}$ is an equivalence relation, one can conveniently represent it during the construction as a collection of sets of elements of T, each set containing elements known to be in the relation with the other elements of that set.

For example, for the set S = {I=J, K=L, A[I]= B[K], J=A[J], M=B[L], A[M]≠ B[K]} that arises from the earlier example, $\dot{=}$ is constructed as follows. From the basis (1), one obtains:

{{I,J}, {K,L}, {A[I], B[K]}, {J,A[J]}, {M,B[L]}}

Using (2):

{{I,J,A[J]} {K,L} {A[I], B[K]}, {M,B[L]}}

Using (3): $\{\{I,J,A[J]\}\ \{K,L\}\ \{A[I], B[K]\}, \{M,B[L]\}, \{A[I], A[J]\}, \{B[K], B[L]\}\}$

Using (2): $\{\{I,J, A[J], A[I], B[K], B[L], M\},\{K,L\}\}$

Using (3): $\{\{I,J,A[J],A[I],B[K],B[L],M\}, \{K,L,\}, \{A[M],A[I]\}, \{A[M],A[J]\}\}$

Using (2): $\{\{I,J,A[J],A[I],B[K],B[L],A[M]\}, \{K,L\}\}$

Since(3) yields no new pairs, the construction is complete. Since $A[M] \doteq B[K]$, S must be unsatisfiable.

The rules for building up $\doteq$ can be implemented quite efficiently. D. Oppen and G. Nelson* have recently coded a very fast implementation that represents terms as graphs and uses the Tarjan [4] set-union algorithm in the closure step. Oppen and Nelson have shown that their implementation requires only order n^2 deterministic time and linear space, where n is the length of the input S.

IV Proof of the Theorem

The main import of the theorem on which the algorithm is based is that it suffices to "consider" only the terms occurring in the formula to be decided. The proof is largely concerned with extending the model provided by $\doteq$ from T to the entire Herbrand Universe.

Theorem S is satisfiable $\Leftrightarrow \nexists\ t_1, t_2 \in T$ such that $t_1 \doteq t_2$ and $'t_1 \neq t_2' \in S$.

Proof $\Rightarrow$ Suppose S is satisfiable, $t_1, t_2 \in T$ and $t_1 \doteq t_2$. Let M be a model for S. Because M satisfies the reflexivity, symmetry, transitivity and substitutivity axioms of equality, $t_1 \doteq t_2$ implies that t_1 and t_2 must have the same values in M. Hence, $'t_1 \neq t_2'$ is not in S.

$\Leftarrow$ Suppose there are no terms t_1, t_2 in T such that $t_1 \doteq t_2$ and $'t_1 \neq t_2' \in S$. We will show that S is satisfiable by constructing a Herbrand model M for S.

We first construct the term universe $T_\infty = \bigcup_{i=0}^{\infty} T_i$ of S inductively as follows:

$T_0 = T \quad T_{i+1} = \{f(t_1,\ldots,t_r) \mid t_i \in T_i\} \cup T_i$
(where f ranges over all function symbols occurring in S).

Next, pick a representative term from each of the equivalence classes induced by $\doteq$ on T, and define the function $a : T \to T$ that assigns to each term in T the representative of its class.

The model M is now constructed inductively as follows:

I. If $t \in T_0$, let $v_M(t) = a(t)$

II. If $t \in T_{j+1} - T_j$, $j \geq 0$, and $t = f(t_1, t_2 \ldots t_r)$, then let

$$v_M(t) = \begin{cases} v_M(f(x_1,\ldots,x_r)) \text{ if } \exists f(x_1,\ldots,x_r) \in T_j \\ \quad \text{and } v_M(x_i) = v_M(t_i),\ 1 \leq i \leq r \\ f(v_M(t_1),\ldots,v_M(t_r)) \quad \text{otherwise} \end{cases}$$

Note that M is a Herbrand model, i.e., it always assigns values from the Berbrand Universe. The notation "$f(v_M(t_1), \ldots, v_M(t_r))$" is intended to represent the function symbol denoted by f followed by the terms obtained by evaluating $v_M(t_i)$ for each i.

To see that M satisfies S, first note that

$'t_1 = t_2' \in S \Rightarrow t_1 \doteq t_2 \Rightarrow a(t_1) = a(t_2) \Rightarrow v_M(t_1) = v_M(t_2)$

and

$'t_1 \neq t_2' \in S \Rightarrow t_1 \not\doteq t_2 \Rightarrow a(t_1) \neq a(t_2) \Rightarrow v_M(t_1) \neq v_m(t_2)$.

It remains to show that

$v_M(x_i) = v_M(y_i)$, $1 \leq i \leq r$, implies that

$v_M(f(x_1,\ldots,x_r)) = v_M(f(y_1,\ldots,y_r))$.

This follows straightforwardly by induction on the maximum of the term universe heights of

$$f(x_1,\ldots,x_r),\ f(y_1,\ldots,y_r)$$

Q.E.D.

V Acknowledgement

The author is grateful to Drs. R. E. Boyer, J. Strother Moore, E. Horowitz, and the reviewers for their observations and suggestions.

VI References

1. Ackermann, W., Solvable Cases of the Decision Problem, North-Holland Publishing Co., Amsterdam (1954), pp. 102-103.

2. Mendelson, E., Introduction to Mathematical Logic, D. Van Nostrand Co., Inc., Princeton, New Jersey (1964).

3. Robinson, G., and Wos, L., "Paramodulation and Theorem-Proving in First-order Theories with Equality." Machine Intelligence IV (1969), ed. by D. Michie and B. Meltzer, pp. 135-150.

4. Tarjan, R., "Efficiency of a good but not linear set-union algorithm," J.ACM, V. 22, No. 2, (April 1975), pp. 215-225.

*Personal communication. A paper describing this work is forthcoming.