Some Approaches to Equality For Computational Logic: A Survey and Assessment

Dallas S. Lankford
Department of Mathematics
Southwestern University
Georgetown, TX 78626

ATP-36

SOME APPROACHES TO EQUALITY FOR COMPUTATIONAL LOGIC: A SURVEY AND ASSESSMENT

Dallas S. Lankford[1]
Department of Mathematics
Southwestern University
Georgetown, TX 78626
Spring 1977

Introduction

In this article recent approaches to equality in computational logic are reviewed. Familiarity with basic aspects of resolution (1), paramodulation (2,3,4,5), and a general knowledge of recursive function theory is assumed.

1. Brute Force Substitution Of Equals

There are two kinds of brute force substitution of equals -- adding the equality axioms as clauses to a given equality unsatisfiable set, and using a separate inference rule for substitution of equals, frequently called paramodulation (2). Experiments with these approaches indicate that they are grossly inefficient; subsequent restrictions (7,8,9,8,9,

1 This work was supported in part by a National Science Foundation grant; Automatic Theorem Proving Project, Depts. Math. and Comp. Sci., Univ. of Texas, Austin, TX 78712, rpt. ATP-36, spring 1977.

10,11,12,13,14,15) have not significantly improved efficiency. The most efficient brute force approach to date is the modification method (7,8). Basically it amounts to transforming a given set of clauses into an enlarged set of clauses so that only the unit reflexive axiom (x = x) and the transitive axiom of equality need be added for hyper-resolution to semi-decide equality unsatisfiability. The modification method is equivalent to a restriction of paramodulation which prohibits unification on variable symbols and does not allow substitution into variable positions. Before the modification method, all refutation completeness results required adding the functional reflexive axioms, all axioms of the form $f(x_1,...,x_n) = f(x_1,...,x_n)$ where n is the degree of f and f ranges over all function symbols of the given set of clauses. The inclusion of the functional reflexive axioms is known to be necessary for the refutation completeness of renameable paramodulation (12). It is not currently known if the inclusion of the functional reflexive axioms is necessary for refutation completeness of other restrictions of paramodulation (3,9,10,13,14,15).

2. Rewrite Rules

Perhaps the earliest use of equations as rewrite rules in computational logic is found in Guard, et al. (16) which led to the first semi-automatic proof of a conjecture from the literature. Others (17,18,19,20,21,22,23,24,25,26,27) also noticed the power of building-in equality with rewrite rules, and the work of Knuth (24) and Knuth and Bendix (17) gives the first fully automatic proof of a mathematical conjecture.

This section focuses on complete sets of reductions (17), also called sets of simplifiers by Slagle (18). Although "simplify" normally means "decrease length", in this article "simplify" is synonymous with "rewrite" (length may not decrease) which is consistent with the terminology introduced by Slagle (18). They are sets of rewrite rules which have two properties -- a finite termination property, and a unique termination property. Intuitively, a set of rewrite rules has the finite termination property when no expression can be infinitely simplified. For example, c ---> f(c) does not have the finite termination property because it infinitely simplifies c . And a set of rewrite rules has the unique termination property if each expression simplifies uniquely, regardless of the order in which the rewrite rules are applied. For example, c ---> d and c ---> e do not have the unique termination property because c does not simplify uniquely.

The importance of a complete set of reductions $R = \{L_1 \text{ ---> } R_1, \ldots, L_n \text{ ---> } R_n\}$ is that it yields a decision algorithm for the corresponding equational theory $E(R) = \{L_1 = R_1,$

... $, L_n = R_n\}$. Knuth and Bendix (19) show that $t = u$ is a consequence of E(R) iff, when t and u are fully simplified by R to t^* and u^*, t^* and u^* are identical. Thus the identities (consequences) of E(R) are decided by using R to simplify equations to canonical or normal forms. The identities of E(R) are just those equations $t = u$ whose normal forms $t^* = u^*$ are of the form $v = v$. In other words, R decides the word problem for E(R) .

Because complete sets of reductions are decision algorithms for the corresponding equational theories, and because there exist (simple) undecidable word problems (28,80), arbitrary finite sets of rewrite rules are not generally complete. In fact, sets of rewrite rules need not be complete for a much simpler reason; namely, commutative rewrite rules like x*y ---> y*x do not have and can not be oriented to have the finite termination property. Nevertheless, many common simple equational theories do have complete sets of reductions, and the remarkable computer experiments of Knuth and Bendix (17) provide convincing evidence that completion, the process of generating complete sets of reductions, is destined to become one of the most important methods for equality in computational logic. A fundamental aspect of completion is the following unique termination theorem.

Unique Termination Theorem Given a set $R = \{L_1 \text{ ---> } R_1, \ldots, L_n \text{ ---> } R_n\}$ of rewrite rules known to have the finite termination property, R has the unique termination property iff for each paramodulant t = u of two equations of E(R), where paramodulation is restricted to left sides into left sides and unification on variables is not allowed, if t^* and u^* are the result of fully simplifying t and u by R, then t^* and u^* are identical expressions.

This theorem was first established by Knuth and Bendix (17) for sets of rewrite rules that satisfy certain order conditions, then extended by Lankford (19,22) to any set of rewrite rules known to have the finite termination property. This theorem provides an algorithm which decides unique termination for sets of rewrite rules known to have the finite termination property. The unique termination algorithm forms all paramodulants with the aforementioned restrictions, then simplifies the restricted paramodulants as far as possible, and finally determines whether or not all simplified paramodulants have the form t = t. There is no need to generate all the restricted paramodulants immediately; they may be generated, simplified, and tested individually.

Here is an example which illustrates the unique termination algorithm. It can be shown (17,19,22) that

2.1 (x*y)*z ---> x*(y*z) ,

2.2 x*1 ---> x , and

2.3 1*x ---> x

have the finite termination property. First generate all restricted paramodulants:

2.4	(w*(x*y))*z = (w*x)*(y*z)	2.1 and 2.1,
2.5	x*(y*1) = x*y	2.1 and 2.2,
2.6	x*z = x*(1*z)	2.2 and 2.1,
2.7	x*y = x*(y*1)	2.2 and 2.1,
2.8	1 = 1	2.2 and 2.3, and
2.9	x*z = 1*(x*z)	2.3 and 2.1.

Next notice that 2.8 has the form t = t and that 2.4, 2.5, 2.6, 2.7, and 2.9 simplify to the form t = t . Thus, 2.1 - 2.3 is a complete set of reductions.

Recall that arbitrary sets of rewrite rules are not always complete sets of reductions. Incompleteness is caused by not having the finite termination property or not having the unique termination property. Let us now consider the latter case, namely one or more restricted paramodulants t = u are fully simplified to t* = u* with t* and u* not identical. Suppose that t* ---> u* or u* ---> t* can be added to the current set of rewrite rules so that the enlarged set of rewrite rules has the finite termination property. And suppose this two step process, (1) generating fully simplified paramodulants via the unique termination algorithm and (2) enlarging the current set of rewrite rules by expressing each new non-trivial paramodulant as a rewrite rule which preserves finite termination, is iterated. One of three possibilities occurs: (1) eventually it is impossible to enlarge the current set of rewrite rules while preserving finite termination, (2) the two step process continues infinitely, or (3) eventually the two step process terminates with a complete set of reductions. This two step process is called completion.

Completion actually includes a third step which accounts for a substantial part of the method's power and efficiency. This third step (previously omitted for simplicity) eliminates redundancies in the enlarged set of rewrite rules. Redundancy elimination consists of taking each rewrite rule L ---> R , then fully simplifying L = R by the other rewrite rules to L* = R* , and finally replacing L ---> R by L* ---> R* or R* ---> L* , provided finite termination is preserved. Although redundancy elimination clearly presents opportunities for completion to halt with an equation that cannot be oriented as a rewrite rule, it often generates equations where L* and R* are identical, which causes L ---> R to be deleted and the current set of rewrite rules to shrink. Thus, completion is a recursively generated sequence of sets of rewrite rules, where the cardinalities of the sets may randomly go up and down.

The following experiment from Knuth and Bendix (17) illustrates completion. Take the three left minimal group axioms

2.10 $1*x \dashrightarrow x$,

2.11 $(x^{-1})*x \dashrightarrow 1$, and

2.12 $(x*y)*z \dashrightarrow x*(y*z)$

and produce paramodulants one by one. In order to simplify the presentation, many details are omitted. In particular, details of paramodulation generation are not given; the ordering which orients new rewrite rules is not discussed; and only the parents from which new rules are derived are listed beside each new rule.

2.13 $(x^{-1})*(x*y) \dashrightarrow y$ — 2.11 and 2.12.

2.14 $(1^{-1})*x \dashrightarrow x$ — 2.11 and 2.13.

2.15 $(x^{-1})^{-1}*1 \dashrightarrow x$ — 2.11 and 2.13.

2.16 $(x^{-1})^{-1}*y \dashrightarrow x*y$ — 2.12 and 2.15.

Next the redundancy elimination procedure replaces 2.15 by

2.17 $x*1 \dashrightarrow x$,

and then completion continues with

2.18 $1^{-1} \dashrightarrow 1$ — 2.12 and 2.17.

Here the redundancy elimination procedure simplifies 2.14 to $x = x$, which is then deleted, and completion continues with

2.19 $(x^{-1})^{-1} \dashrightarrow x$ — 2.16 and 2.17.

Now the redundancy elimination procedure simplifies 2.16 to $x*y = x*y$, which is then deleted, and completion continues. Subsequent deletions are of the kinds illustrated above.

2.20 $x*x^{-1} \dashrightarrow 1$ — 2.11 and 2.19.

2.21	$((x*y)^{-1})*(x*(y*z)) \dashrightarrow z$	2.12 and 2.13.
2.22	$x*(x*(x*y)^{-1}) \dashrightarrow 1$	2.12 and 2.20.
2.23	$x*((x^{-1})*y) \dashrightarrow y$	2.12 and 2.20.
2.24	$y*(z*(x*((y*(z*x))^{-1})) \dashrightarrow 1$	2.12 and 2.21.
2.25	$y*(z*((y*z)^{-1}*x)) \dashrightarrow x$	2.12 and 2.22.
2.26	$x*((y*x)^{-1}) \dashrightarrow y^{-1}$	2.13 and 2.21.
2.27	$y*((x*y)^{-1}*z) \dashrightarrow x^{-1}*z$	2.12 and 2.26. Delete 2.24.
2.28	$y*(z*(x*(y*z))) \dashrightarrow x^{-1}$	2.13 and 2.26. Delete 2.25.
2.29	$(x*y)^{-1} \dashrightarrow y^{-1}*x^{-1}$	2.13 and 2.26. Delete 2.21, 2.23, 2.27, and 2.28.

Completion halts with a complete set of reductions for group theory consisting of 2.10, 2.11, 2.12, 2.13, 2.17, 2.18, 2.19, 2.20, 2.23, and 2.29.

Besides the ultimate obstacle of undecidable word problems, a major disadvantage of completion is that current methods of detecting finite termination (17,19,22) fail to decide finite termination. Detect finite termination means that an algorithm is given a set of rewrite rules as input and outputs one of two answers -- "yes, the set of rules has the finite termination property," or "this algorithm can't determine whether the set of rules has the finite termination property." The methods currently used are all based on having a subset $<$ of T^2 that satisfies the following two properties, where T is the set of all terms and T^2 is the Cartesian product of T with itself.

Replacement Induced Order Property If $t > u$, and w is the result of replacing one occurrence of t in v by u, then $v > w$.

Finite Decrease Order Property There does not exist an infinite sequence $t_1, t_2, t_3, \ldots$ of terms such that $t_1 > t_2 > t_3 > \ldots$.

The finite termination methods of Guard, et al. (16) use a similar ordering $>$ that also well-orders all terms. For a given pair of terms t and u satisfying $t > u$, it does not generally follow that $tX > uX$ for all substitutions X where tX and uX are the substitution instances of t and u by X. Thus their approach does not really use rewrite rules, but rather equations, both of whose sides must be used for substitution. The advantage of their method is that because the set of all terms is well-ordered, all equations can be used for simplification. The disadvantage of their method is that although unique termination can be defined, most common algebras do not have complete (in this new sense) sets of reductions, nor does completion (in this new sense) generally terminate with a complete set of reductions.

The finite termination property is characterized by the previous two properties together with the following two properties.

Instiation Order Preserving Property If $t > u$ and X is a substitution, then $tX > uX$.

Order-Rewrite Compatibility Property $L_1 > R_1$, ..., and $L_n > R_n$, where $R = \{L_1 \dashrightarrow R_1, \ldots, L_n \dashrightarrow R_n\}$.

Finite Termination Theorem If $R = \{L_1 \dashrightarrow R_1, \ldots, L_n \dashrightarrow R_n\}$ is a set of rewrite rules, then R has the finite termination property iff there exists a subset $>$ of T^2 satisfying the replacement induced order property, the finite decrease order property, the instantiation order preserving property, and the order-rewrite compatibility property.

A proof of the finite termination theorem is given in (27).

A finite termination detector is an algorithm which detects finite termination. The finite termination detectors of Knuth and Bendix (17) and Lankford (19,22) depend upon guessing a solution for $>$ of the finite termination theorem. Knuth and Bendix (17) guess the ordering $>$ before starting completion, although in several examples they overcome initial failures by useful methods which allow them to reguess. Lankford (19,22) generalizes their approach and introduces methods based on partially guessing > at the outset and using a decision algorithm for elementary algebra to complete the guess dynamically while completion is in progress. However, Stickel (94) has recently pointed out that these approaches severely restrict which orderings > may be guessed. What is ultimately desired is a decision algorithm for finite termination. A recent result by Lankford (27) provides such a decision algorithm for ground (variable free) rewrite rules. But the decidability of the general finite termination is presently an open question.

Another disadvantage of completion is that some equations, such as commutative equations, fail to have the finite termination property. Judging by experimental evidence (17,22,31), most finite termination detector failures are caused by equations which permute symbols, for example $f(x,f(y,z)) = f(y,f(z,x))$. If finite termination orderings > could be found so that failure equations are all of predictable forms, and if those predictable forms could be built into unification algorithms, then it might be worthwhile to expand the notion of rewrite rule accordingly, and develop unique termination algorithms for the expanded notion. Unfortunately, little is known about how to build arbitrary equational theories into unification algorithms (18,20,30,34). On the other hand, commutative unification and commutative completion can be developed (29) and other significant generalizations seem feasible.

The methods of Knuth and Bendix (17) were developed for equational theories, or equivalently, for sets of unit clauses of the form $\{t_1 = u_1, \ldots, t_n = u_n, NOT(v = w)\}$. Indipendently, Slagle (18) initiated the more general study of combining complete and incomplete sets of reductions with resolution on some kinds of non-unit clauses. Lankford (19,22,27,29) combined and generalized both approaches. The basic idea (18,19,22) is to treat as many equations as possible as rewrite rules, using unit rewrite rules to keep things simplified as far as possible. An important theoretical lesson learned from these studies is that for refutation completeness, the rewrite rules, whether complete set or not, must be allowed to interact with the other predicates in a process called narrowing (18,22). This lays to rest the hope of entirely building-in equality even for theories whose equational parts are decidable.

"Narrowing" is somewhat of a misnomer because it consists of paramodulating with left sides of rewrite rules and rewriting the paramodulant as far as possible. So even if a rewrite rule decreases length, after unification, replacement and rewriting, the resulting "narrowed" clause may be longer than the parent.

Studies of narrowing (18,19,22) give the designer a reasonably good idea of how to use complete or incomplete sets of reductions, but an important theoretical problem remains to be solved. How does one apply rewrite rule methods to sets of clauses which contain equations in non-unit clauses? The obvious approach has not been established without adding the (undesirable!) functional reflexive axioms. An apparently closely related problem is uniform replacement (paramodulation causes a single occurrence of one side of an equation to be replaced, while uniform replacement causes each occurrence throughout the entire clause to be replaced). The ground completeness of uniform replacement has been established in the unit case (78,79), and lifts without the functional reflexive axioms. But repeated efforts to induct using the excess literal parameter method (93) have not yielded a proof of the ground uniform replacement conjecture.

Experiments which support the use of equations as rewrite rules have been done in both purely equational theories and in more general first order and higher order theories. Experiments with purely equational theories include those conducted by Ballantyne and Lankford (31), Guard, et al. (16), Huet (23), Knuth (24), Knuth and Bendix (17), and Nevins (20). The human-oriented deductive system developed by Nevins (20) is not limited to purely equational theories, and is in many ways a heuristic anticipation of completion and narrowing. The first explicit computer experiments with completion and narrowing by Ballantyne and Lankford (31) improved speed and search space size of one of Nevins' examples by about two orders of magnitude. Other diverse uses of rewrite rules are included in Ballantyne and Bledsoe (26), Bledsoe (25), Bledsoe, et al. (21), Boyer and Moore (32), and Milner (33)

3. Building Equality Into Unification Algorithms

Plotkin (34) suggests that equational theories be built into unification algorithms and provides some guidelines for doing so. But the fact that associativity alone cannot generally be built-in is a major disadvantage, and moreover, associative unification seems inefficient (20) when compared with a rewrite rule treatment of associativity (22). However, recall that if finite termination equations were all of a predictable form, say permutative equations, then having permutative unification algorithms available might make it possible to treat such equational theories by rewrite rules modulo permutative unification. Currently only commutative (29) and commutative-associative (30) unification algorithms have been developed.

4. Decision Algorithms

The study of decision algorithms for equational theories was perhaps initiated by Dehn (35) who gave a decision method for the theory of fundamental groups of two dimensional manifolds. Since then, word problem decision algorithms have been developed for a number of equational theories; for example, by Magnus (36,71) for group theory and one additional axiom, by Tartakovski (37,38) for certain enlargements of group theory. Mechanical theorem proving experiments have not used these classical kinds of methods.

Complete sets of reductions, discussed above (in 2. Rewrite Rules), appear to be a very powerful method for developing practical mechanical theorem provers in purely equational theories and some more general theories (using narrowing).

Certain well-known decision algorithms for several common mathematical theories, such as Abelian groups, rings, and Boolean algebras, have been used successfully. For example, the experiments of Bledsoe, et al. (21) and Ballantyne and Bledsoe (26) approximate field theory with a built-in canonical form for ring theory. Other common mathematical theories, like polynomials over a ring and modules over a ring, can be treated similarly. It has not been investigated how such decision algorithms can be combined with rewrite rule methods to form larger decision algorithms, semi-decision procedures, and so on.

The decision algorithms above apply to purely equational theories. There are also well-known decision algorithms for theories which contain non-unit clauses, or which contain predicates in addition to equality; for example, the elementary theory of Abelian groups (39); periodic, divisible, cyclic, and p-groups (40); ordered Abelian groups (41); algebraically closed fields (42,43,44); fields of fixed characteristic (42,43,44); the field of complex numbers (42,44); the field of real numbers (42,44); elementary algebra and geometry (45,65,72); elementary hyperbolic geometry (46); Boolean algebras (42,47); distributive lattices (47); Pressburger arithmetic, i.e., addition of natural numbers (48,67); addition of ordinals (49); addition of cardinals (50); free commutative semigroups (51); the theory of equations (52); a subclass of pressburger arithmetic (53,54); computable fields (55); pure equality of sets (56,67); unary predicates (57,58,59); transitive-symmetric predicates (60); unary functions (61); linear ordering (62); real and p-adic fields (63); real closed fields and algebraically closed fields with a predicate separating algebraic integers (65); certain free algebras (66); and two successor functions (70). For additional discussion and details see (67,68).

These methods, with the exception of a fragment of Pressburger arithmetic (53), have not been implemented. Some progress has been made by Collins (69) toward an implementation of elementary algebra. A problem with these approaches is that they are frequently too complex (69,73,74). However, the work of Bledsoe (53) on Pressburger arithmetic suggests that for practical applications a feasible approach is to implement fragments of a decision algorithm rather than an entire algorithm.

Another potential source of methods for treating equality are the ground decision procedures for zero order predicate calculi with equality (27). (The zero order predicate calculus is the "ground" or"variable-free" first order predicate calculus.) It is well-known that ground clauses without equality are decidable by resolution (1). And it follows immediately from the work of Knuth and Bendix (17) that ground unit equational theories are decidable by complete sets of reductions. So it is straightforward by the excess literal parameter method (93) to show that ground predicate calculi with equality are decidable by taking the closure of the given set of clauses under resolution and ordered (for example, by the lexical order (1)) paramodulation -- the given set of clauses is unsatisfiable iff the closure contains the empty clause or a clause of the form {NOT{t_1 = u_1}, ... , NOT{t_n = u_n}} (27). This ground decidability result is the basis of the refutation completeness results for rewrite rules reported by Lankford (19,22), but it has not been shown to lift when equations occur in non-unit clauses without the functional reflexive axioms. Overbeek (77) uses a similar ground decision algorithm as the basis of a general level theorem prover which semi-decides equality unsatisfiability by well-ordering the set of all ground terms and sending the ground decision algorithm successively deeper instantiations of the given set of clauses. These two methods can be regarded as the general level analogs of the Knuth and Bendix (17) vs. Guard, et al. (16) methods for purely equational theories. A variation of the former ground decision algorithm above has been proposed but not established beyond the ground unit case (78,79). The conjecture is that uniform replacement is a decision algoroithm for ground clauses. A theoretically attractive feature of the ground uniform replacement conjecture is that it would lift without the functional reflexive axioms. Perhaps the recent discoveries of Brand (7,8) could be aplied to this problem.

5. Non-standard Model Theory

Recently it was learned (75) that real number theory could be formally axiomatized using the notion of infinitesimal as suggested by Liebnitz and Newton. Ballantyne (76) and Ballantyne and Bledsoe (26) noticed that non-standard analysis methods provide more efficient implementations than the classical epsilon-delta approach for much of the differential calculus. A close inspection of these methods reveals that many concepts, including convergence, uniform convergence, continuity, uniform continuity, differentiability, and Cauchy sequence, can be expressed as formulas about the equivalence relation # which is defined by x # y iff x and y are in the same monad (26). This equivalence relation is also a congruence relation on the ring of finite real numbers, which means the relation # may be treated as if it were equality on the ring of finite real numbers. In particular, the infinitesimal implementation (26) derives much of its power from representing many formulas of the differential calculus as equational formulas which are then treated by rewrite rule methods. Since Ballantyne and Bledsoe (26) use mostly ad hoc rewrite rule methods, completion methods (17,18,19,20,22) might significantly enhance the nonstandard approach. These non-standard experiments (26,76) also suggest the need for theoretical and experimental knowledge about how to combine two or more decision algorithms, sets of rewrite rules, congruence relations, and so on. Of particular interest is the case when congruence relations occur within a larger theory of equality.

6. Conclusions And Remarks

There is curremtly no objective method for evaluating mechanical theorem proving methods. A subjective evaluation of an implementation can be obtained from the list of theorems proved, but the value of such subjective evaluations is debatable for intricate ad hoc implementations, especially those which include combinations of methods. With considerable reservations based on the above remarks, the approaches to equality presented in this article are evaluated as follows.

The most useful brute force method is the modification method (7,8), but it is practical only for moderately difficult problems.

For large or difficult theories, decision algorithm methods seem to be the only approaches that offer much hope for practical theorem provers in theories which include equality. Consequently, the decision algorithm methods -- and especially the rewrite rule methods -- should be studied further, both theoretically and experimentally. A number of theoretical and experimental research problems have been presented in this article, and the interested reader can easily find others.

The above mentioned areas of application deal mainly with traditional mathematical theories. Another area of growing interest concerns the kinds of theorems which arise from program verification, as illustrated by the work of Deutsch (88), Good, et al. (89), King (90), Levitt and Waldinger (91), and others. Judging by the kinds of theorems they encounter, completion and other decision algorithms for equality should be useful in these kinds of program verification systems.

The equality methods discussed in this article apply directly to resolution, paramodulation, and ad hoc natural deductive systems. Other deductive systems are also possible bases, including Gentzen systems (78,81) and inverse systems (82,83,84,85,86,87), but the details have not yet been worked out.

Acknowledgments

I would like to thank Mark Moriconi for carefully reading the first draft of this article and for making numerous suggestions about style and presentation.

References

1. J. A. Robinson A machine-oriented logic based on the resolution principle. **JACM** 12, 1 (Jan. 1965), 23-41.

2. G. Robinson and L. Wos Paramodulation and theorem proving in first-order theories with equality. **Machine Intelligence 4**, eds. B. Meltzer and D. Michie, Edinburgh University Press, 1969, 135-150.

3. L. Wos and G. A. Robinson Paramodulation and set of support. **Lecture Notes In Mathematics** 125, Springer-Verlag, 1968, 276-310.

4. L. Wos and G. Robinson The maximal model theorem. Spring 1968 meeting of Assn. for Symb. Logic.

5. L. Wos and G. Robinson Maximal models and refutation completeness: semi-decision procedures in automatic theorem proving. **Studies In Logic And The Foundations Of Mathematics**, Word problems,; decision problems and the Burnside problem in group theory, Vol. 71, eds. W. Boone, F.

Cannonito and R. Lyndon, North-Holland Pub. Co., 1973, 609-693.

6. R. Kowalski The case for using equality axioms in automatic demonstration. **Lecture Notes in Mathematics** 125, Springer-Verlag, 1968, 112-127.

7. D. Brand Proving theorems with the modification method. **SIAM J. Comput**. 4, 4 (Dec. 1975), 412-430.

8. D. Brand Resolution and equality in theorem proving. Tech. Rept. No. 58, Dept. of Comp. Sci., Univ. of Toronto, Nov. 1973.

9. J. B. Morris E-resolution: extensions of resolution to include the equality relation. **Proc. IJCAI** 1, eds. D. Walker and L. Norton, MITRE Corp., Bedford, MA, 1969, 287-294.

10. R. Anderson Completeness results for E-resolution. **AFIPS Spring Joint Computer Conference** 36 (1970), 653-656.

11. R. Anderson Some theoretical aspects of automatic theorem proving. Ph.D. dissertation, Univ. of Texas, Austin, TX, Aug. 1970.

12. C. L. Chang Renamable paramodulation for automatic theorem proving with equality. **Artif. Intell. J.** 1, 4 (Dec. 1970), 247-256.

13. C. L. Chang and J. R. Slagle Linear refutation for theories with equality. **JACM** 18, 1 (Jan. 1971), 126-136.

14. R. Anderson Completeness of the locking restriction for paramodulation. **SIGART Newsletter** 30 (Oct. 1971), 4.

15. D. S. Lankford Equality atom term locking. Ph.D. dissertation, Univ. of Texas, Austin, TX, Dec. 1972.

16. J. Guard, F. Oglesby, J. Bennett, and L. Settle Semi-automated mathematics. **JACM** 16, 1 (Jan. 1969), 49-62.

17. D. E. Knuth and P. B. Bendix Simple word problems in universal algebras. **Computational Problems in Abstract Algebras**, ed. J. Leech, Pergamon Press, 1970, 263-297.

18. J. R. Slagle Automated theorem proving for theories with simplifiers, commutativity, and associativity. **JACM** 21, 4 (Oct. 1974), 622-642.

19. D. S. Lankford Canonical algebraic simplification. Automatic Theorem Proving Project, Depts. Math. and Comp. Sci., Univ. of Texas, Austin, TX, rpt. ATP-25, May 1975.

20. A. J. Nevins A human oriented logic for automatic theorem proving. **JACM** 21, 4 (Oct. 1974), 606-621.

21. W. W. Bledsoe, R. S. Boyer and W. H. Henneman Computer proofs of limit theorems. **Artif. Intell. J.** 3 (1972), 27-60.

22. D. S. Lankford Canonical inference. Automatic Theorem Proving Project, Depts. Math. and Comp. Sci., Univ. of Texas, Austin, TX, rpt. ATP-32, Dec. 1975.

23. G. P. Huet Experiments with an interactive theorem prover for logic with equality. Jennings Computing Center, Case Western Reserve Univ., Cleveland, OH, rpt. No. 1106, 1972.

24. D. E. Knuth Notes on central groupoids. **J. of Comb. Theory** 8 (1970), 376-390.

25. W. W. Bledsoe Splitting and reduction heuristics in automatic theorem proving. **Artif. Intell. J.** 2 (1971), 57-78.

26. A. M. Ballantyne and W. W. Bledsoe Automatic proofs of theorems in analysis using non-standard techniques. Automatic Theorem Proving Project, Depts. Math. and Comp. Sci., Univ. of Texas, Austin, TX, rpt. ATP-23, July 1973.

27. D. S. Lankford On mechanizing zero and first order equality calculi. Unpublished manuscript, spring 1977.

28. J. V. Matijasevic Simple examples of undecidable associative calculi. **Soviet Math. Dokl.** 8, 2 (1967), 555-557.

29. D. S. Lankford and A. M. Ballantyne Commutative reduction. Unpublished notes, spring 1977.

30. M. E. Stickel A complete unification algoeithm for associative-commutative functions. Unpublished manuscript, tutorial given at **IJCAI** 4, Tibilisi, USSR, Sept. 1975.

31. A. M. Ballantyne and D. S. Lankford An implementation of derived reduction. Local memo, Depts. Math. and Comp. Sci., Univ. of Texas, Austin, TX, Aug. 1975.

32. R. S. Boyer and J. S. Moore Proving theorems about LISP functions. **JACM** 22, 1 (Jan. 1975), 129-144.

33. R. Milner Logic for computable functions, description of a machine implementation. Stanford AI Project, Comp. Sci. Dept., Stanford Univ., Stanford, CA, Memo AIM-169, May 1962.

34. G. Plotkin Building-in equational theories. **Machine Intelligence 7**, eds. B. Meltzer and D. Michie, John Wiley & Sons, 1972, 73-89.

35. M. Dehn Uber unendliche diskontinuierliche Gruppen. **Math. Ann.** 71 (1911), 116-144.

36. W. Magnus Das Identitatsproblem fur Gruppen mit einer definierenden Relation. **Math. Ann.** 106 (1932), 295-307.

37. **Amer. Math. Soc. Translation No. 60**, Translation of Three Russian Papers: The sieve method in group theory. Mat. Sbornik N. S. 25 (67), 1949, 3-50. Application of the siev method to the word problem for certain types of groups. Mat. Sbornik N. S. 25 (67), 1949, 251-274. Solution of the word problem for groups with a k-reduced basis k > 6. Izvestiya Akad. Nauk. SSSR Ser. Mat. 13, 1949, 483-494.

38. M. Hall, Jr. Generators and relations in groups - the Burnside problem. **Lectures On Modern Mathematics**, Vol. II, ed. T. Saaty, John Wiley & Sons, 1964, 42-92.

39. W. Szmielew Elementary properties of Abelian groups. **Fund. Math.** 41 (1954), 203-271.

40. Y. L. Ershov Decidability of elementary theories of some classes of Abelian groups. **Algebra i Logika Seminar** 6 (1963), 37-41.

41. Y. L. Gurevich Elementary properties of ordered Abelian groups. **Algebra i Logika Seminar** 3, 1 (1964), 5-39, **MR** 28 # 5004.

42. A. Tarski Arithmetic classes and types of mathematical systems, mathematical aspects of arithmetical classes and types, arithmetical classes and types of Boolean algebras, arithmetical classes and types of algebraically closed and real-closed fields. **Bull. Amer. Math. Soc.** 55 (1949), 63-64.

43. R. Vaught Application of the Lowenheim-Skolem-Tarski theorem to problems of completeness and decidability. **Indag. Math.** 16 (1954), 467-472.

44. A. Robinson **Complete Theories**. North-Holland Pub. Co., 1956.

45. A. Tarski **A Decision Method For Elementary Algebra And Geometry**. Univ. of California Press, Berkeley, 1951.

46. W. Schwabhauser Entscheidbarkeit und Vollstandigkeit der elementaren hyperbolischen Geometrie. **Z. Math. Logik Grundlagen Math.** 5 (1959), 132-205.

47. Y. L. Ershov Decidability of the theory of relatively complemented distributive lattices and the theory of filters. **Algebra i Logika Seminar** 3, 3 (1964), 17-38.

48. M. Pressburger Uber de Vollstandigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem de Addition als einzige Operation hervortritt. **Comtes Rendus du Ier Congres des Mathematiciens des payes Slaves**, Warszawa, 1927, 92-101,395.

49. A Ehrenfeucht Applications of games to the completeness problem for formalized theories. **Fund. Math.** 49 (1961), 129-141.

50. A. Tarski **Ordinal Algebras**. North-Holland Pub. Co., 1956.

51. A Mostowski On direct products of theories. **JSL** 17 (1952), 1-31.

52. A. Robinson On the mechanization of the theory of equations. **Bulletin Of The Research Council Of Israel**, Vol. 9F (1960), 47-70.

53. W. W. Bledsoe The sup-inf method in Pressburger arithmetic. Automatic Theorem Proving Project, Depts. Math. and Comp. Sci., Univ. of Texas, Austin, TX, rpt. ATP-18, Dec. 1974.

54. R. E. Shostak On the sup-inf method for proving Pressburger formulas. Local memo, Stanford Research Institute, Menlo Park, CA, 1976.

55. M. Rabin Computable algebra, general theory and theory of conputable fields. **Trans. Amer. Math. Soc.** 95 (Apr.-June 1960), 341-360.

56. L. Lowenheim Uber Moglichkeiten im Relativkalkul. **Math. Ann.** 76 (1915), 447-470.

57. W. Ackermann **Solvable Cases Of The Decision Problem**. North-Holland Pub. Co., 1954.

58. P. S. Novikov **Elements Of Mathematical Logic**. Oliver & Boyd, 1964.

59. A. Church **Introduction To Mathematical Logic**. Vol. I. Princeton Univ. Press, Princeton, 1956.

60. H. Rogers Certain logical reduction and decision problems. **Ann. of Math.** 64 (1956), 264-284.

61. A. Ehrenfeucht Decidability of the theory of one function. **Notices Amer. Math. Soc.** 6 (1959), 268, no. 556-37.

62. A. Ehrenfeucht Decidability of the theory of linear ordering relation. **Notices Amer. Math. Soc.** 6 (1959), 286, no. 556-38.

63. P. J. Cohen Decision procedures for real and p-adic fields. **Comm. Pure Appl. Math.** 22 (1969), 131-151.

64. A Seidenberg A new decision method for elementary algebra. **Ann. of Math.** 60, 2 (Sept. 1964), 365-374.

65. A. Robinson Solution of a problem of Tarski. **Fund. Math.** 47 (1959), 179-214.

66. M. A. Taitslin Decidability of certain elementary theories. **Algebra i Logika Seminar** 3, 3 (1964), 5-12.

67. J. D. Monk **Mathematical Logic**. Springer-Verlag, 1976, 233-242.

68. Y. Ershov, I. Lavrov and A. Taimanov Elementary theories. **Russian Math. Surveys** 20 (1965), 35-105.

69. G. E. Collins Quantifier elimination for real closed fields by cylindric algebraic decomposition. **Lecture Notes In Computer Science** 33, Proc. 2nd GI Conference On Automata And Formal Languages, Kaiserslauten, Springer-Verlag, 1975, 134-183.

70. M. Rabin Decidability of second order theories and automata on infinite trees. **Trans. Amer. Math. Soc.** 141 (1961), 1-35.

71. W. Magnus, A. Karrass and D. Solitar **Combinatorial Group Theory**. New Interscience Publishers, 1966.

72. N. Jacobson **Basic Algebra, I**. Freeman & Co., 1976, 290-318.

73. L. Stockmeyer and A. Meyer Word problems requiring exponential time. **Fifth Annual ACM Symp. on Theory of Comput.**, 1973,

74. M. Fischer and M. Rabin Super-exponential complexity of Pressnurger arithmetic. **SIAM-AMS Proc.** 7 (1974), 27-41.

75. A. Robinson **Nonstandard Analysis**. North-Holland Pub. Co., 1966.

76. A. M. Ballantyne Non-standard analysis, part II: The meta theorist. Automatic Theorem Proving Project, Depts.

Math. and Comp. Sci., Univ. of Texas, Austin, TX, rpt. ATP-30, Jan. 1977.

77. R. Overbeek A new class of automatic theorem proving algorithms. **JACM** 21, 2 (Apr. 1974), 191-200.

78. M. Richter Resolution, paramodulation, and Gentzen-systems. Schriften zur Informatik und Angewandten Mathematik, Rheinisch-Westfalische Technische Hochschule, 51 Aachen, Templergraben 55, West Germany, Bericht Nr. 23, 1976.

79. M. Richter A note on paramodulation and the functional reflexive problem. Unpublished notes, 1975.

80. J. V. Matijasevic Simple examples of unsolvable canonical calculi. **Proc. Stek. Inst. Math.** 93 (1967), Problems in the Constructive Trend in Mathematics, IV, eds. V. Orevkov and M. Sanin, Amer. Math. Soc., Providence, 1970, 61-110.

81. V. A. Lifsic Specialization of the form of deduction in the predicate calculus with equality and function symbols, I. **Proc. Stek. Inst. Math.** 98 (1968), The Calculi of Symbolic Logic, I, ed. V. Orevkov, Amer. Math. Soc., Providence, 1971, 1-23.

82. S. J. Maslov The inverse method for establishing decidability for logical calculi. **Proc. Stek. Inst. Math.** 98 (1968), 25-96, **MR** 40, # 5516, **MR** 43, # 4620.

83. S. J. Maslov The inverse method and tatics for establishing decidability for a calculus with function symbols. Proc. Stek. Inst. Math. 121 (1972), Logical and Logico-Mathematical Calculi, II, ed. V. O. Orevkov, Amer. Math. Soc., Providence, 1974 11-59.

84. S. J. Maslov The connection between tatics of the inverse method and the resolution method. **Sem. Math. V. A. Stek. Inst. Leningrad** 16 (1969), 69-73, **MR** 42, # 1645.

85. S. J. Maslov An inverse method for establishing deducibility of nonprenex formulas of the predicate calculus. **Soviet Math. Dokl.** 8 (1967), 16-19, **MR** 35, # 19.

86. G. S. Ceitin The complexity of deduction in the propositional predicate calculus. **Sem. Math. V. A. Stek. Math. Inst. Leningrad** 8 (1968), 115-125, **MR** 43, # 6098.

87. D. G. Kuehner A note on the relation between resolution and Maslov's inverse method. **Machine Intelligence 6**, eds. B. Meltzer and D. Michie, American Elsevier Pub. Co. Inc., 1971, 73-76.

88. L. P. Deutsch An interactive program verifier. Ph.D. dissertation, Univ. of California, Berkeley, 1973; also Xerox Palo Alto Research Center, rpt. CSL-73-1, May 1973.

89. D. Good, R. London, and W. Bledsoe An interactive program verification system. **IEEE Trans Soft. Eng.** 1 (1975) 59-67.

90. J. C. King A program verifier Ph.D. dissertation, Carnegie-Mellon Univ., Pittsburg, PA, 1969.

91. F. von Henke and D. Luckham A methodology for verifying programs. **In Proc. Int. Cong. Reliable Soft.**, 1975.

92. R. Waldinger and N. Levitt Reasoning about programs. **Artif. Intel. J.** 5, 3 (Fall 1974), 235-316; also **Conf. Rec. ACM Symp. Princ. Prog. Lang.** (1973), 169-182.

93. R. Anderson and W. Bledsoe A linear format for resolution with merging and a new technique for establishing completeness. **JACM** 17 (July 1970), 525-534.

94. M. E. Stickel The inadequacy of primitive recursive complexity measures for determining finite termination of sets of reductions. Local memo, Dept. Comp. Sci., Univ. Arizona, Tucson, AZ, Dec. 1976.