8

Paramodulation and Theorem-proving in First-Order Theories with Equality

G. Robinson
Stanford Linear Accelerator Center
Stanford, California

and

L. Wos
Argonne National Laboratory
Argonne, Illinois

INTRODUCTION

A *term* is an individual constant or variable or an n-adic function letter followed by n terms. An *atomic formula* is an n-adic predicate letter followed by n terms. A *literal* is an atomic formula or the negation thereof. A *clause* is a set of literals and is thought of as representing the universally-quantified disjunction of its members. It will sometimes be notationally convenient[1] to distinguish between the empty clause $\square$, viewed as a clause, and 'other' empty sets such as the empty set of clauses, even though all these empty sets are the same set-theoretic object ϕ. A *ground* clause (term, literal) is one with no variables. A *clause* C' (literal, term) is an *instance* of another clause C (literal, term) if there is a uniform replacement of the variables in C by terms that transform C into C'.

The *Herbrand universe* H_S of a set S of clauses is the set of all terms that can be formed from the function letters and individual constants occurring in S (with the proviso that if S contains no individual constant, the constant a is used). An *interpretation* I of a set S of clauses is a set of ground literals such that for each atomic formula F that can be formed from an n-adic predicate letter occurring in S and n terms from H_S, exactly one of the literals F or $\bar{F}$ (the negation of F) is in I.

For any set J of literals, $\bar{J}$ is the set of negations of members of J. The set J *satisfies* a ground clause C if $J \cap C \neq \phi$ and *condemns* C if $C - \bar{J} = \phi$. J *satisfies* a non-ground clause C if it satisfies every instance of C and *condemns* C if it condemns some instance of C. A clause (possibly ground) that is neither

[1] Note, for example, that the empty set is a satisfiable set of clauses but at the same time is an unsatisfiable clause.

satisfied nor condemned by J is said to be *undefined* for J; otherwise it is *defined* for J. J *satisfies* a set S of clauses if it satisfies every clause in S and *condemns* S if it condemns some clause in S.

An *R-interpretation* of a set S of clauses is an interpretation I of S having the following properties: Let α, β, and γ be any terms in H_S and L any literal in I. Then

1. $(\alpha=\alpha) \in I$
2. If $(\alpha=\beta) \in I$ then $(\beta=\alpha) \in I$
3. If $(\alpha=\beta) \in I$ and $(\beta=\gamma) \in I$, then $(\alpha=\gamma) \in I$.
4. If L' is the result of replacing some one occurrence of α in L by β and $(\alpha=\beta) \in I$, then $L' \in I$.

An (R-)*model* of S is an (R-)interpretation of S that satisfies S.

A set S of clauses is (R-)*satisfiable* if there is an (R-)model of S; otherwise it is (R-)*unsatisfiable*.

If S is a set of clauses or a single clause and T is a set of clauses or a single clause, S(R-)*implies* T(abbreviation $S \models T$ or $S \models_R T$) if no (R-)model of S condemns T.

A deductive system W is (R-)*deduction-complete* if $S \vdash_W T$(T is deducible from S in the system W) whenever $S \models T$ ($S \models_R T$). W is (R-)*refutation-complete* if $S \vdash_W \square$ whenever S is (R-)unsatisfiable.

EQUALITY IN AUTOMATIC THEOREM-PROVING

The methods for dealing with the concept of equality in theorem-proving can be grouped roughly into three classes: (1) those which employ a set of first-order axioms for equality, for example, the following set (which we shall call $E(K)$, where K is the set of first-order sentences under study):

(i) $(x_1)(x_1=x_1)$

(ii) $(x_1)\ldots(x_n)(x_0)\,(x_j \neq x_0 \vee \bar{P}x_1\ldots x_j\ldots x_n \vee Px_1\ldots x_0\ldots x_n)$ $\quad (j=1,\ldots,n)$

(iii) $(x_1)\ldots(x_n)(x_0)\,(x_j \neq x_0 \vee f(x_1\ldots x_j\ldots x_n)=f(x_1\ldots x_0\ldots x_n))$ $\quad (j=1,\ldots,n)$

where n axioms of the form (ii) are included for each n-adic ($n>0$) predicate letter P occurring in K, and n axioms of the form (iii) are included for each n-adic ($n>0$) function letter in K[1]; (2) those which employ a smaller set of second-order axioms for equality; and (3) those which employ a substitution rule for equals as a rule of inference.

SOME DESIRABLE PROPERTIES FOR THEOREM-PROVING ALGORITHMS

In addition to the logical properties of soundness and completeness, two sets of somewhat more elusive properties are of interest in judging the usefulness of the inference apparatus for automatic theorem-proving.

[1] Note that an interpretation I of K is an R-interpretation of K iff it satisfies $E(K)$.

The first set, *efficiency*, *brevity*, and *naturalness*, are global properties in that they deal with the entire proof or proof-search, and are of interest in themselves. *Efficiency* refers to the ease or dispatch with which the search procedure locates a proof. *Brevity* refers to the lengths of proofs found. *Naturalness* refers to being in the spirit of what a human mathematician might write in a proof. Other factors being equal, a briefer proof might be considered more natural, but naturalness goes beyond this. For example, among proofs of roughly the same length, a unit resolution proof[1] might be considered more natural than a non-unit proof.

The second set, *immediacy*, *convergence*, and *generality*, are local properties in that they focus on only a small part of the proof or proof-search and are of interest primarily because they contribute to other properties such as efficiency.

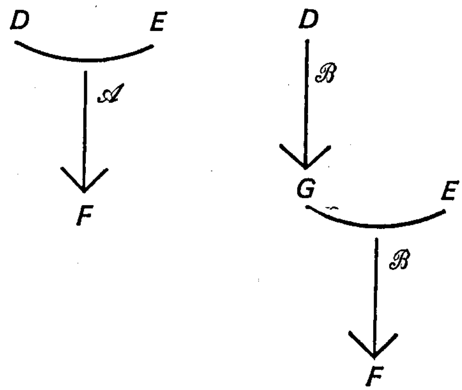

Figure 1

Immediacy is rather easily grasped. One inference apparatus $\mathscr{A}$ is said to be more immediate than another apparatus $\mathscr{B}$ (at least for the case in question) when $\mathscr{A}$ enables one to deduce a given conclusion from a given set of hypotheses in fewer steps than $\mathscr{B}$. For example (see figure 1), if to infer F from D and E by $\mathscr{B}$ one first had to infer G from D and only then infer F from E and G, while $\mathscr{A}$ allowed the inference of F directly from D and E in one step without recourse to G, then $\mathscr{A}$ would (for this case) be more immediate than $\mathscr{B}$.

Convergence is a slightly subtler but, for automatic theorem-proving, perhaps more important property. Consider the clause G in the example above. Often such an intermediate result will seriously detract from proof-search efficiency by interacting with other clauses to produce unnecessary 'noise' in the proof-search space, either by generating successive generations of less than helpful clauses, or, somewhat less seriously, by requiring additional

[1] In effect one that is free from simultaneous case-analysis type reasoning and which prefers *modus ponens* to syllogism—formally, one in which non-unit clauses are never resolved against each other.

machine time to determine that no interesting clauses can be inferred from G. Freedom from this generation of 'side-effect' clauses we call *convergence*. Thus in the example, $\mathscr{A}$ is both more immediate and more convergent than $\mathscr{B}$.

Generality refers to choosing to infer a clause C rather than a proper instance of C, when either inference could be made from the premises without loss of soundness. For example, inferring from $f(xa)=g(x)$ and $Qf(xa)$ the conclusion $Qg(b)$, although sound, would be less general than inferring $Qg(x)$.

It is not difficult to see the advantage of inferring a clause rather than a proper instance of that clause, since the more general clause, being stronger, has greater potential for future inferences. Perhaps even easier to see is the problem of deciding which proper instance to select if a proper instance were to be preferred to the more general clause. Usually there is an infinite set of proper instances. For example, from $h(xyy)=g(x)$ and $Qh(zww)a$, we can infer $Qg(x)a$ by substitution. There is, however, an infinite set of proper instances of $Qg(x)a$ which could also be legitimately inferred. Among these are $Qg(a)a$, $Qg(g(a))a$, $Qg(g(g(a)))a$. . . . We shall apply the phrase *most general* to a clause (or term) C with respect to some given condition when C satisfies the condition and no clause (term) which satisfies the condition has C as a proper instance.

Of the approaches to equality described above, approach 1 has three obvious disadvantages. One has to do with length of deduction chains in the proof. In order to infer from

(1) Qa and
(2) $a=b$

the result

(3) Qb

one must first infer from the axiom

(4) $x\neq y \vee \bar{Q}x \vee Qy$

and, say (1), the intermediate result

(5) $a\neq y \vee Qy$,

before passing from (5) and (2) to (3). By contrast, approach 3 would allow us to go directly from (1) and (2) to (3) without ever inferring the intermediate result (5). Thus approach 3 contributes to *brevity of proofs*. More important for proof search, it contributes (by means of immediacy) to *brevity of deduction chains* within proofs.

A second, and perhaps more serious disadvantage of approach 1 as compared to approach 3, is that the intermediate debris such as step (5) tends to spawn increasingly larger generations of generally useless offspring, polluting the search space badly. We describe this difference by saying that approach 3 tends to be more *convergent* than approach 1. (Presence of various subsidiary strategies, such as set-of-support, may possibly mitigate the severity of such non-convergence effects.)

The third disadvantage of approach 1 is perhaps the least important, although superficially the most obvious: the equality axioms $E(K)$ must be present. The clerical chore of writing them all down could be eliminated merely by incorporating into the theorem-prover a program to generate them. Alternatively they may be specified by means of a schema (we shall call this variation approach 1b), or in approach 2 by means of a few second-order axioms. We feel that this third disadvantage is so superficial and trivial (since one can simply place $E(K)$ outside the set of support as is done in the standard set-of-support variant of approach 1) as to be quite spurious.

The method given by Darlington (1968), whether it be classed as approach 1b or as approach 2, can be taken as typical of methods which avoid the third disadvantage (greater number of explicit axioms) but fail to dent the first and second disadvantages (longer deduction chains and non-convergence). In effect Darlington infers (5) from (1) and

$$(4')\ x\neq y \vee \varphi(x) \vee \varphi(y),$$

which is thought of either as a schema defining a set of first-order axioms including (4), or as a single second-order axiom having (4) as an instance.

PARAMODULATION

Since our automatic theorem-proving environment consists exclusively of clauses, we should like our rule of inference for equality to operate on two clauses and yield a clause. Furthermore, we should like it to apply to units and non-units alike[1] and to yield a most general clause that can be R-soundly inferred. We shall now describe the inference rule for paramodulation, which is asserted to have these properties. Examples of paramodulation are given in figure 2.[2]

Paramodulation: Given clauses A and $\alpha'=\beta' \vee B$ (or $\beta'=\alpha' \vee B$) having no variable in common and such that A contains a term δ, with δ and α' having a most general common instance α identical to $\alpha'[s_i/u_i]$ and to $\delta[t_j/w_j]$, form A' by replacing in $A[t_j/w_j]$ some single occurrence of α (resulting from an occurrence of δ)[3] by $\beta'[s_i/u_i]$, and infer $A' \vee B[s_i/u_i]$.[4]

[1] Consider for example the set $S=\{c=d \vee \bar{Q}c, g(c)\neq g(d) \vee \bar{Q}c, a=b \vee Qc, g(a)\neq g(b) \vee Qc, x=x\}$. If the rule applied only to units, it would not be possible to this R-unsatisfiable set.

[2] These examples are primarily to give an intuitive idea of how paramodulation works. A comparison of the length and complexity of paramodulation proofs against resolution proofs can be obtained by considering the proofs of the theorem from group theory to the effect that $x^3=e$ implies $((x,y),y)=e$. The resolution proof is 136 steps long while the paramodulation proof is 47 steps long. These proofs appear in the appendix.

[3] Without this restriction one could infer from $a=b$ and $Qxa \vee Px$ the clause $Qab \vee Pa$ (a proper instance of the paramodulant $Qxb \vee Px$), resulting in a loss of generality.

[4] Since every non-trivial immediate modulant (see Wos *et al.*, 1967b) of a clause is a paramodulant, any clause obtained by demodulation can be obtained by repeated paramodulation.

Example 1	*Example* 2	*Example* 3	*Example* 4
1. $a=b$	1. $a=b$	1. $a=b$	1. $a=b$
2. Qa	2. Qx	2. $Qx \vee Px$	2. $Qx \vee Px$
3. $\therefore Qb$	3. $\therefore Qb$	3. $\therefore Qb \vee Pa$	3. $\therefore Qa \vee Pb$

Example 5	*Example* 6	*Example* 7
1. $x=h(x)$	1. $a=b$	1. $f(xg(x))=e$
2. $Qg(y)$	2. $Qf(g(h(j(a))))$	2. $Pyf(g(y)z)z$
3. $\therefore Qh(g(y))$	3. $\therefore Qf(g(h(j(b))))$	3. $\therefore Pyeg(g(y))$

Example 8. If $x^2=e$ for all x in a group, the group is commutative.

1.	$f(ex)=x$	
2.	$f(xe)=x$	
3.	$f(xf(yz))=f(f(xy)z)$	
4.	$f(xx)=e$	
5.	$f(ab)=c$	
6.	$c \neq f(ba)$	
7.	$f(xe)=f(f(xy)y)$	4 into 3 with $\delta: f(yz)$
8.	$x=f(f(xy)y)$	2 into 7 on $f(xe)$
9.	$a=f(cb)$	5 into 8 on $f(xy)$
10.	$f(yf(yz))=f(ez)$	4 into 3 on $f(xy)$
11.	$f(yf(yz))=z$	1 into 10 on $f(ez)$
12.	$f(ca)=b$	9 into 11 on $f(yz)$
13.	$c=f(ba)$	12 into 8 on $f(xy)$
14.	$\square$	13 resolved with 6

Figure 2

From a superficial point of view, paramodulation might be described as 'a substitution rule for equality'. Indeed, the motivation given above for studying the rule has dwelt principally on that aspect of paramodulation. But to consider it as only substitution of equals for equals would be to make a mistake analogous to characterizing resolution as merely syllogistic inference akin to that employed by Davis and Putnam (1960). The property of maximum generality provided by paramodulation must not be overlooked if the process is to be fully understood. Consider the following example:

From $f(xg(x))=e \vee Qx$ and $Pyf(g(y)z)z \vee Wz$ one can infer $Pyeg(g(y)) \vee Qg(y) \vee Wg(g(y))$ by paramodulating with $f(xg(x))$ as α' and $f(g(y)z)$ as δ.

COMPLETENESS OF PARAMODULATION FOR BASIC GROUP THEORY

Consider the following clauses from the first-order theory of groups:

A1	$Pxyf(xy)$	closure
A2	$Pexx$	left identity
A3	$Pg(x)xe$	left inverse
A4	$\bar{P}xyu \vee \bar{P}yzv \vee \bar{P}uzw \vee Pxvw$	associativity (case 1)

A5	$\bar{P}xyz \vee \bar{P}xyu \vee z=u$	uniqueness of product
A6	$z \neq u \vee \bar{P}xyz \vee Pxyu$	substitution (3rd position)
A7	$z \neq u \vee \bar{P}xzy \vee Pxuy$	substitution (2nd position)
A8	$z \neq u \vee \bar{P}zxy \vee Puxy$	substitution (1st position)
A9	$x=x$	reflexivity
A10	$x \neq y \vee y=x$	symmetry
A11	$x \neq y \vee y \neq z \vee x=z$	transitivity
A12	$x \neq y \vee f(xz)=f(yz)$	f-substitution (1st position)
A13	$x \neq y \vee f(zx)=f(zy)$	f-substitution (2nd position)
A14	$x \neq y \vee g(x)=g(y)$	g-substitution

Let us define a *basic* set S of clauses of group theory to be a set over the vocabulary of A1–A14 and such that $S \vdash \{A1, \ldots, A5\}$. We then have the following completeness result for the special case of basic sets.

Theorem: If S is a satisfiable, fully paramodulated, fully factored, basic set of clauses of group theory, then S is R-satisfiable.

Proof: Let M be a maximal model[1] of S. Suppose that $\alpha=\beta$ and $P\gamma\delta\alpha$ are both in M. By the maximality of M, there must be clauses A and B in S having instances $A'{:}\alpha=\beta \vee K$ and $B'{:}P\gamma\delta\alpha \vee L$ with $K \cap M=\phi=L \cap M$. Then factors of A and B can be paramodulated on the arguments corresponding to α to give a clause in S having $P\gamma\delta\beta \vee K \vee L$ as an instance. Since M satisfies S, $(P\gamma\delta\beta \vee K \vee L) \cap M \neq \phi$. But $(K \vee L) \cap M=\phi$. Hence $P\gamma\delta\beta \in M$. Thus M satisfies A6. It can be shown[2] that A1–A6$\vdash$A7–A14. Hence M satisfies A6–A14 and is therefore an R-model of S.

This result is generalized to the case of what will be called functionally-reflexive systems in the next section.

COMPLETENESS OF PARAMODULATION FOR FUNCTIONALLY-REFLEXIVE SYSTEMS

Paramodulation is intended to be utilized, along with resolution, for theorem-proving in first-order theories with equality.[3]

We first give an algorithm for generating a refutation (of a finite set of clauses) employing paramodulation and resolution if such a refutation exists.

Full Search Algorithm (*FSA*): Let S_0 be the set of all factors of the given set S of clauses[4]. For odd $i>0$ let S_i be formed from S_{i-1} by adding all clauses

[1] The concept of maximal model is defined and the pertinent existence theorem proved in Wos and Robinson (1968a). For the present purpose a maximal model of S may be thought of as a model M such that for each positive literal x in M there is an instance C' of some C in S with $C' \cap M=\{x\}$.

[2] Robinson and Wos (1967c).

[3] The earliest formulations of paramodulation were designed to operate without resolution and could be shown to subsume resolution as a special case. It is felt, however, that the processes can be better understood if the inference apparatus not involving equality is isolated from the apparatus for equality, even if this means that some of the completeness theorems cannot be stated in quite as pat a fashion.

[4] Every clause is a factor of itself as in G.Robinson *et al.* (1964b). For further definitions of factoring and resolution see Wos *et al.* (1964a) and J.Robinson (1965).

that can be obtained by paramodulating two clauses in S_{i-1}. For even $i>0$ let S_i be formed from S_{i-1} by adding all factors of clauses that can be obtained by resolving two clauses in S_{i-1}. Since each deduction from S is contained in S_n for some n, each refutation of S must be contained in S_n for some n. Each S_j is finite. If S_j contains $\square$, a refutation has been found, so stop. Otherwise form S_{j+1}.

Now, to prove that paramodulation and resolution are complete for theorem-proving in first-order theories with equality, we would like to show that *FSA* is a semi-decision procedure for *R*-unsatisfiability. The difficult part is to show that, for *R*-unsatisfiable sets of clauses, there *exists* a refutation, namely, that paramodulation plus resolution is *R*-refutation complete. It will suffice to show that an unsatisfiable set can be deduced from an *R*-unsatisfiable set, since (due to the refutation-completeness of resolution) *FSA* will generate a refutation if it ever generates an unsatisfiable set.

A functionally-reflexive system S is defined as one for which $S \vdash x_1 = x_1$ and $S \vdash f(x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$ for every function letter f occurring in the vocabulary of S, n being the degree of f. There are $h+1$ such unit clauses, where h is the number of function letters in the vocabulary of S. For such systems refutation-completeness is proved in Wos and Robinson (1968c).[1] From that result one can obtain the following corollary: If S is a finite functionally-reflexive set of clauses, *FSA* is a semidecision procedure for *R*-unsatisfiability.

Even for theories that do not happen to be functionally reflexive, this result shows that adding the $h+1$ functional-reflexivity unit clauses before applying *FSA* gives a general semi-decision procedure for *R*-unsatisfiability.

FURTHER COMPLETENESS RESULTS FOR PARAMODULATION

Since first-order theories are not usually functionally-reflexive when the only rules are resolution and paramodulation, and since adding the functional-reflexivity units to the theory may detract somewhat from proof-search efficiency, one would wish to show that some weaker assumption than functional-reflexivity will suffice for completeness. It seems that at least $S \vdash x = x$ will be needed. (Consider the case where S consists of $\{a \neq a\}$. S is *R*-unsatisfiable but cannot be refuted without some sort of help from reflexivity.) This is not surprising, since the standard texts on logic that use the substitution rule or schema approach to equality consistently supply a separate reflexivity axiom.[2]

But is simple reflexivity ($x=x$) enough? We think so,[3] although a proof of this is not yet available.

[1] A weaker version of this result was given in the earlier (1968b) paper.
[2] See, e.g., Church (1956) or Quine (1963).
[3] In the two years that paramodulation has been under study, no counterexample has been found to the *R*-refutation completeness of paramodulation and resolution for simply-reflexive systems.

To see where the difficulty lies in generalizing the proof given in Wos and Robinson (1968c) beyond the functionally-reflexive case, we examine the relation between deductions and refutations based on a given set S and those based on proper instances of clauses from S.

Capturing lemma[1]: Let S be a fully paramodulated and fully resolved set of clauses such that $S \vdash x=x$, and let A' and B' be instances of clauses A and B in S and let C' be the result of paramodulating from a term α' in A' into an occurrence δ_0 of a term in B'. Then

Strong subterm form: There is a clause C in S with C' as an instance.
Restricted subterm form: If B has a term in the same position as that of δ_0 in B', then there is a clause C in S with C' as an instance.
(Occurrences of terms in two literals are said to be in the *same position* if each is the i_1-st argument of the i_2-nd argument of ... of the i_n-th argument of its literal.)
Argument form: If δ is an argument of B' (as opposed to a proper subterm of an argument), then there is a clause C in S with C' as an instance.

When the strong subterm form of the capturing lemma holds and $S \vdash x=x$, every maximal model (with respect to positive literals) of S is an R-model, and since every satisfiable set S has a maximal model, it follows that either $\square \in S$ or S is R-satisfiable. Thus the strong subterm form of the capturing lemma and simple reflexivity imply R-refutation-completeness. The line of proof given for R-refutation-completeness in functionally-reflexive systems in (1968c) depends (at least indirectly) on the strong subterm form, which happens to hold in such systems.[2] The following example will suffice to show however that the strong subterm form is not universally true:

$$S\colon \{x=x,\ a=b,\ b=a,\ a=a,\ b=b,\ Qxg(x),\ Qag(a),\ Qbg(b),\ Qag(b),\ Qbg(a)\}$$

A: $a=b$
A': $a=b$
B: $Qxg(x)$
B': $Qg(a)g(g(a))$
C': $Qg(b)g(g(a))$

S is fully paramodulated and (vacuously) fully resolved. A' and B' paramodulate on a into the first occurrence of a in B' to give C'. But C' is an instance of no clause in S. (The restricted subterm form of the lemma is not violated since B has no term in the same position as the first occurrence of a in B'. Neither is the argument form of the lemma, since a is not an argument

[1] The analogue of this capturing lemma for resolution alone plays a basic role in proving the refutation-completeness of resolution (see J. Robinson, 1965 and Slagle, 1967) and of set-of-support (Wos *et al.*, 1965).
[2] Alternatively, one can view the difficulty as resulting from the fact that it is not always possible to satisfy the hypotheses of the restricted subterm form.

of B'.) Functional-reflexivity of S, if present, would dispose of the difficulty, since, if $g(x)=g(x)$ were in S, so would $g(a)=g(b)$ be in S if it were fully paramodulated; and hence the result $Qg(b)g(g(a))$ of paramodulating $g(a)=g(b)$ and $Qxg(x)$ would be in S and serve as C.

Weakening the strong subterm capturing lemma in a different fashion leads to the

Refutation capturing lemma: If there exists a refutation of a set of instances of clauses in a set S by means of paramodulation and resolution, then there exists a refutation of S itself by means of paramodulation and resolution.

For functionally-reflexive S, this lemma may be proved by noting that the refutability of a set of instances of S and R-soundness of paramodulation and resolution yield the R-unsatisfiability of S; so that the refutation-completeness of paramodulation and resolution for functionally-reflexive systems establishes the refutability of S itself.

Given the refutation capturing lemma one could prove the following:

General refutation-completeness: If S is a fully paramodulated and fully resolved R-unsatisfiable set and if $S \vdash x=x$, then $\square \in S$.

Corollary: *FSA* is a semi-decision procedure for R-unsatisfiability for finite sets S of clauses such that $S \vdash x=x$.

Conversely, given general refutation-completeness, one can prove the refutation capturing lemma (at least for systems S such that $S \vdash x=x$). In view of this equivalence, proof of the refutation capturing lemma can be considered the most pressing unsolved problem in the theory of paramodulation. Alternatively, one might seek a proof of general refutation-completeness based on the restricted subterm form of the capturing lemma, which holds even when the assumption of functional reflexivity is suppressed.

Acknowledgement

This work was supported by the US Atomic Energy Commission.

REFERENCES

Church, A. (1956) *Introduction to mathematical logic* I. Princeton.

Darlington, J.L. (1968) Automatic theorem proving with equality substitutions and mathematical induction. *Machine Intelligence 3*, pp. 113–27 (ed. Michie, D.). Edinburgh: Edinburgh University Press.

Davis, M. & Putnam, H. (1960) A computing procedure for quantification theory. *J. Assn. Comput. Mach.*, **7**, 201–15.

Quine, W.V.O. (1963) *Set theory and its logic*. Cambridge, Mass: Harvard University Press.

Robinson, J.A. (1965) A machine-oriented logic based on the resolution principle. *J. Assn. Comput. Mach.*, **12**, 23–41.

Slagle, J.R. (1967) Automatic theorem proving with renamable and semantic resolution. *J. Assn. Comput. Mach.*, **14**, pp. 687–97.

Wos, L., Carson, D. & Robinson, G. (1964a) The unit-preference strategy in theorem proving. *AFIPS* Conference Proceedings, **26**, Washington D.C.: Spartan Books, 615–21.

Robinson, G.A., Wos, L.T. & Carson, D. (1964b) Some theorem-proving strategies and their implementation. *AMD* Tech. Memo No. 72, Argonne National Laboratory.
Wos, L., Robinson, G.A. & Carson, D.F. (1965) Efficiency and completeness of the set of support strategy in theorem proving. *J. Assn. Comput. Mach.*, **12**, 536–41.
Robinson, G.A., Wos, L. & Shalla, L. (1967a) Two inference rules for first-order predicate calculus with equality. *AMD* Tech. Memo No. 142, Argonne National Laboratory.
Wos, L., Robinson, G.A., Carson, D.F. & Shalla, L. (1967b) The concept of demodulation in theorem proving. *J. Assn. Comput. Mach.*, **14**, 698–709.
Robinson, G. & Wos, L. (1967c) Dependence of equality axioms in elementary group theory. Comp. Group Tech. Memo No. 53, Stanford Linear Accelerator Center.
Wos, L. & Robinson, G. (1968a), The maximal model theorem. Spring 1968 meeting of Assn. for Symbolic Logic. Abstract to appear in *J. Symb. Logic.*
Robinson, G. & Wos. L. (1968b) Completeness of paramodulation. Spring 1968 meeting of Assn. for Symbolic Logic. Abstract to appear in *J. Symb. Logic.*
Wos, L. & Robinson, G. (1968c) Maximal models and refutation completeness (unpublished).

APPENDIX

Paramodulation versus resolution

Problem: $x^3=e$ implies $((x,y),y)=e$ where $(x,y)=xyx^{-1}y^{-1}$
Reference: *Group Theory* by Marshall Hall, page 322, 18.2.8.

Refutation by Paramodulation

1. $f(ex)=x$
2. $f(xe)=x$
3. $f(g(x)x)=e$
4. $f(xg(x))=e$
5. $f(xf(yz))=f(f(xy)z)$
6. $x=x$
7. $f(f(xx)x)=e$
8. $h(xy)=f(f(f(xy)g(x))g(y))$
9. $h(h(ab)b)\neq e$
10. $f(xe)=f(f(xy)g(y))$, $f(xg(x))$ of 4 into $f(yz)$ of 5
11. $x=f(f(xy)g(y))$, $f(xe)$ of 2 into $f(xe)$ of 10
12. $x=f(eg(g(x)))$, $f(xg(x))$ of 4 into $f(xy)$ of 11
13. $x=g(g(x))$, $f(ex)$ of 1 into $f(eg(g(x)))$ of 12
14. $f(f(xx)f(xz))=f(ez)$, $f(f(xx)x)$ of 7 into $f(xy)$ of 5
15. $f(f(xx)f(xz))=z$, $f(ex)$ of 1 into $f(ez)$ of 14
16. $f(f(xx)e)=g(x)$, $f(xg(x))$ of 4 into $f(xz)$ of 15
17. $f(xx)=g(x)$, $f(xe)$ of 2 into $f(f(xx)e)$ of 16
18. $f(f(xy)f(g(y)z))=f(xz)$, $f(f(xy)g(y))$ of 11 into $f(xy)$ of 5
19. $f(f(xy)f(g(y)g(x)))=e$, $f(xg(x))$ of 4 into $f(xz)$ of 18
20. $f(we)=f(f(wf(xy))f(g(y)g(x)))$, $f(f(xy)f(g(y)g(x)))$ of 19 into $f(yz)$ of 5
21. $w=f(f(wf(xy)f(g(y)g(x)))$, $f(xe)$ of 2 into $f(we)$ of 20

22. $g(f(xy))=f(ef(g(y)g(x))), f(g(x)x)$ of 3 into $f(wf(xy))$ of 21
23. $g(f(xy))=f(g(y)g(x)), f(ex)$ of 1 into $f(ef(g(y)g(x)))$ of 22
24. $g(h(xy))=f(g(g(y))g(f(f(xy)g(x)))), f(f(f(xy)g(x))g(y))$ of 8 into $f(xy)$ of 23
25. $g(h(xy))=f(yg(f(f(xy)g(x)))), g(g(x))$ of 13 into $g(g(y))$ of 24
26. $g(h(xy))=f(yf(g(g(x))g(f(xy)))), g(f(xy))$ of 23 into $g(f(f(xy)g(x)))$ of 25
27. $g(h(xy))=f(yf(xg(f(xy)))), g(g(x))$ of 13 into $g(g(x))$ of 26
28. $g(h(xy))=f(yf(xf(g(y)g(x)))), g(f(xy))$ of 23 into $g(f(xy))$ of 27
29. $f(f(f(h(ab)b)g(h(ab)))g(b))\neq e, h(xy)$ of 8 into $h(h(ab)b)$ of 9
30. $f(f(f(f(f(f(ab)g(a))g(b))b)g(h(ab)))g(b))\neq e, h(xy)$ of 8 into $h(ab)$ of 29
31. $f(f(f(f(f(ab)g(a))f(g(b)b))g(h(ab)))g(b))\neq e, f(f(xy)z)$ of 5 into $f(f(f(f(ab)g(a))g(b))b)$ of 30
32. $f(f(f(f(f(ab)g(a))e)g(h(ab)))g(b))\neq e, f(g(x)x)$ of 3 into $f(g(b)b)$ of 31
33. $f(f(f(f(ab)g(a))g(h(ab)))g(b))\neq e, f(xe)$ of 2 into $f(f(f(ab)g(a))e)$ of 32
34. $f(f(f(f(ab)g(a))f(bf(af(g(b)g(a)))))g(b))\neq e, g(h(xy))$ of 28 into $g(h(ab))$ of 33
35. $f(f(f(f(ab)f(aa))f(bf(af(g(b)g(a)))))g(b))\neq e, g(x)$ of 17 into $g(a)$ of 34
36. $f(f(f(f(f(ab)f(aa))b)f(af(g(b)g(a))))g(b))\neq e, f(xf(yz))$ of 5 into $f(f(f(ab)f(aa))f(bf(af(g(b)g(a)))))$ of 35
37. $f(f(f(f(f(f(ab)f(aa))b)a)f(g(b)g(a)))g(b))\neq e, f(xf(yz))$ of 5 into $f(f(f(f(ab)f(aa))b)f(af(g(b)g(a))))$ of 36
38. $f(f(f(f(f(f(f(ab)a)a)b)a)f(g(b)g(a)))g(b))\neq e, f(xf(yz))$ of 5 into $f(f(ab)f(aa))$ of 37
39. $f(f(f(f(f(f(ab)a)f(ab))a)f(g(b)g(a)))g(b))\neq e, f(f(xy))z)$ of 5 into $f(f(f(f(ab)a)a)b)$ of 38
40. $f(f(f(f(f(ab)a)f(f(ab)a))f(g(b)g(a)))g(b))\neq e, f(f(xy)z)$ of 5 into $f(f(f(f(ab)a)f(ab))a)$ of 39
41. $f(f(f(f(ab)a)f(f(ab)a))f(f(g(b)g(a))g(b)))\neq e, f(f(xy)z)$ of 5 into $f(f(f(f(f(ab)a)f(f(ab)a))f(g(b)g(a)))g(b))$ of 40
42. $f(f(f(f(ab)a)f(f(ab)a))f(g(f(ab)))g(b))\neq e, f(g(y)g(x))$ of 23 into $f(g(b)g(a))$ of 41
43. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(ab)f(ab))g(b)))\neq e, g(x)$ of 17 into $g(f(ab))$ of 42
44. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(f(ab)a)b)g(b)))\neq e, f(xf(yz))$ of 5 into $f(f(ab)f(ab))$ of 43
45. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(ab)a)f(bg(b))))\neq e, f(f(xy)z)$ of 5 into $f(f(f(f(ab)a)b)g(b))$ of 44 .
46. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(ab)a)e))\neq e, f(xg(x))$ of 4 into $f(bg(b))$ of 45

47. $f(f(f(f(ab)a)f(f(ab)a))f(f(ab)a)) \neq e, f(xe)$ of 2 into
$f(f(f(f(ab)a)f(f(ab)a))f(f(f(ab)a)e))$ of 46

7 contradicts 47.

Paramodulation versus resolution

Problem: $x^3=e$ implies $((x,y),y)=e$.

Refutation by Resolution

1. $f(ex)=x$
2. $f(xe)=x$
3. $f(e(x)x)=e$
4. $f(xg(x))=e$
5. $f(xf(yz))=f(f(xy)z)$
6. $x=x$
7. $x \neq y \quad y=x$
8. $x \neq y \quad y \neq z \quad x=z$
9. $u \neq w \quad f(ux)=f(wx)$
10. $u \neq w \quad f(xu)=f(xw)$
11. $u \neq w \quad g(u)=g(w)$
12. $f(f(xx)x)=e$
13. $h(xy)=f(f(f(xy)g(x))g(y))$
14. $h(h(ab)b) \neq e$
15. $x \neq f(ew) \quad x=w$, 1 and 8_2
16. $f(f(xg(x))w)=f(ew)$, 4 and 9_1
17. $f(f(xy)z) \neq w \quad f(xf(yz))=w$, 5 and 8_1
18. $f(xf(g(x)z))=f(ez)$, 16 and 17_1
19. $f(xf(g(x)z))=z$, 18 and 15_1
20. $f(uf(yg(y)))=f(ue)$, 4 and 10_1
21. $f(ue)=f(uf(yg(y)))$, 20 and 7_1
22. $f(uf(yg(y))) \neq z \quad f(ue)=z$, 21 and 8_1
23. $f(xe)=g(g(x))$, 19 and 22_1
24. $x=f(xe)$, 2 and 7_1
25. $f(xe) \neq z \quad x=z$, 24 and 8_1
26. $x=g(g(x))$, 23 and 25_1
27. $f(f(f(uu)u)y)=f(ey)$, 12 and 9_1
28. $f(f(f(uu)u)y)=y$, 27 and 15_1
29. $f(f(xx)f(xy))=y$, 28 and 17_1
30. $f(f(xx)e)=g(x)$, 29 and 22_1
31. $f(xx)=g(x)$, 30 and 25_1
32. $f(xe)=f(f(xy)g(y))$, 5 and 22_1
33. $x=f(f(xy)g(y))$, 32 and 25_1
34. $f(xz)=f(f(f(xy)g(y))z)$, 33 and 9_1
35. $f(f(f(xy)g(y))z)=f(xz)$, 34 and 7_1
36. $f(f(xy)f(g(y)z))=f(xz)$, 35 and 17_1

37. $x \neq f(ug(u)) \quad x=e$, 4 and 8_2
38. $f(f(xy)f(g(y)g(x)))=e$, 36 and 37_1
39. $e=f(f(xy)f(g(y)g(x)))$, 38 and 7_1
40. $f(we)=f(wf(f(xy)f(g(y)g(x))))$, 39 and 10_1
41. $u \neq f(xf(yz)) \quad u=f(f(xy)z)$, 5 and 8_2
42. $f(ue)=f(f(uf(xy))f(g(y)g(x)))$, 40 and 41_1
43. $u=f(f(uf(xy))f(g(y)g(x)))$, 42 and 25_1
44. $f(f(g(x)x)u)=f(eu)$, 3 and 9_1
45. $z \neq f(f(g(x)x)u) \quad z=f(eu)$, 44 and 8_2
46. $g(f(xy))=f(ef(g(y)g(x)))$, 43 and 45_1
47. $g(f(xy))=f(g(y)g(x))$, 46 and 15_1
48. $g(h(xy))=g(f(f(f(xy)g(x))g(y)))$, 13 and 11_1
49. $u \neq g(f(xy)) \quad u=f(g(y)g(x))$, 47 and 8_2
50. $g(h(xy))=f(g(g(y))g(f(f(xy)g(x))))$, 48 and 49_1
51. $g(g(x))=x$, 26 and 7_1
52. $f(g(g(u))z)=f(uz)$, 51 and 9_1
53. $x \neq f(g(g(u))z)=f(uz)$, 52 and 8_2
54. $g(h(xy))=f(yg(f(f(xy)g(x))))$, 50 and 53_1
55. $f(zg(f(xy)))=f(zf(g(y)g(x)))$, 47 and 9_1
56. $u \neq f(zg(f(xy))) \quad u=f(zf(g(y)g(x)))$, 55 and 8_2
57. $g(h(xy))=f(yf(g(g(x))g(f(xy))))$, 54 and 56_1
58. $f(yf(g(g(u))z))=f(yf(uz))$, 52 and 10_1
59. $x \neq f(yf(g(g(u))z)) \quad x=f(yf(uz))$, 58 and 8_2
60. $g(h(xy))=f(yf(xg(f(xy))))$, 57 and 59_1
61. $f(uf(zg(f(xy))))=f(uf(zf(g(y)g(x))))$, 55 and 10_1
62. $w \neq f(uf(zg(f(xy)))) \quad w=f(uf(zf(g(y)g(x))))$, 61 and 8_2
63. $g(h(xy))=f(yf(xf(g(y)g(x))))$, 60 and 62_1
64. $f(zg(h(xy)))=f(zf(yf(xf(g(y)g(x)))))$, 60 and 62_1
65. $f(wf(zg(h(xy))))=f(wf(zf(yf(xf(g(y)g(x))))))$, 64 and 10_1
66. $f(uf(wf(zg(h(xy)))))=f(uf(wf(zf(yf(xf(g(y)g(x)))))))$, 65 and 10_1
67. $f(uf(wf(zg(h(xy)))))=f(f(uw)f(zf(yf(xf(g(y)g(x))))))$, 66 and 41_1
68. $f(uf(wf(zg(h(xy)))))=f(f(f(uw)z)f(yf(xf(g(y)g(x)))))$, 67 and 41_1
69. $f(f(xy)z)=f(xf(yz))$, 5 and 7_1
70. $f(xf(yz)) \neq u \quad f(f(xy)z)=u$, 69 and 8_1
71. $f(f(uw)f(zg(h(xy))))=f(f(f(uw)z)f(yf(xf(g(y)g(x)))))$, 68 and 70_1
72. $f(f(f(uw)z)g(h(xy)))=f(f(f(uw)z)f(yf(xf(g(y)g(x)))))$, 71 and 70_1
73. $f(f(f(f(xy)z)g(h(xy)))u)=f(f(f(f(xy)z)f(yf(xf(g(y)g(x)))))u)$, 72 and 9_1
74. $f(h(xy)z)=f(f(f(f(xy)g(x))g(y))z)$, 13 and 9_1
75. $u \neq f(f(xy)z) \quad u=f(xf(yz))$, 69 and 8_2
76. $f(h(xy)z)=f(f(f(xy)g(x))f(g(y)z))$, 74 and 75_1
77. $f(uf(g(x)x))=f(ue)$, 3 and 10_1
78. $z \neq f(uf(g(x)x)) \quad z=f(ue)$, 77 and 8_2

79. $f(h(xy)y)=f(f(f(xy)g(x))e)$, 76 and 78_1
80. $u\neq f(xe) \quad u=x$, 2 and 8_2
81. $f(h(xy)y)=f(f(xy)g(x))$, 79 and 80_1
82. $f(f(h(xy)y)z)=f(f(f(xy)g(x))z)$, 81 and 9_1
83. $f(f(f(h(xy)y)z)w)=f(f(f(f(xy)g(x))z)w)$, 82 and 9_1
84. $h(h(ab)b)\neq y \quad y\neq e$, 14 and 8_2
85. $f(f(f(h(ab)b)g(h(ab)))g(b))\neq e$, 13 and 84_1
86. $f(f(f(h(ab)b)g(h(ab)))g(b))\neq y \quad y\neq e$, 85 and 8_3
87. $f(f(f(f(ab)g(a))g(h(ab)))g(b))\neq e$, 83 and 86_1
88. $f(f(f(f(ab)b(a))b(h(ab)))g(b))\neq y \quad y\neq e$, 87 and 8_3
89. $f(f(f(f(ab)g(a))f(bf(af(g(b)g(a))))g(b))\neq e$, 73 and 88_1
90. $g(x)=f(xx)$, 31 and 7_1
91. $f(wg(x))=f(wf(xx))$, 90 and 10_1
92. $f(uf(wg(x)))=f(uf(wf(xx)))$, 91 and 10_1
93. $f(uf(wg(x)))=f(f(uw)f(xx))$, 92 and 41_1
94. $f(f(uw)g(x))=f(f(uw)f(xx))$, 93 and 70_1
95. $f(f(f(uw)g(x))y)=f(f(f(uw)f(xx))y)$, 94 and 9_1
96. $f(f(f(f(uw)g(x))y)z)=f(f(f(f(uw)f(xx))y)z)$, 95 and 9_1
97. $f(f(f(f(ab)g(a))f(bf(af(g(b)g(a)))))g(b))\neq y \quad y\neq e$, 89 and 8_3
98. $f(f(f(f(ab)f(aa))f(bf(af(g(b)g(a)))))g(b))\neq e$, 96 and 97_1
99. $f(f(f(f(ab)f(aa))f(bf(af(g(b)g(a))))))\neq y \quad y\neq e$, 98 and 8_3
100. $f(f(xf(yz))u)=f(f(f(xy)z)u)$, 5 and 9_1
101. $f(f(f(f(f(ab)f(aa))b)f(af(g(b)g(a))))g(b))\neq e$, 100 and 99_1
102. $f(f(f(f(f(ab)f(aa))b)f(af(g(b)g(a))))g(b))\neq y \quad y\neq e$, 101 and 8_3
103. $f(f(f(f(f(f(ab)f(aa))b)a)f(g(b)g(a)))g(b))\neq e$, 100 and 102_1
104. $f(f(f(xf(yz))u)v)=f(f(f(f(xy)z)u)v)$, 100 and 9_1
105. $f(f(f(f(xf(yz))u)v)w)=f(f(f(f(f(xy)z)u)v)w)$, 104 and 9_1
106. $f(f(f(f(f(xf(yz))u)v)w)t)=f(f(f(f(f(f(xy)z)u)v)w)t)$, 105 and 9_1
107. $f(f(f(f(f(f(ab)f(aa))b)a)f(g(b)g(a)))g(b))\neq y \quad y\neq e$, 103 and 8_3
108. $f(f(f(f(f(f(f(ab)a)a)b)a)f(g(b)g(a)))g(b))\neq e$, 106 and 107_1
109. $f(f(f(f(f(xy)z)u)v)w)=f(f(f(f(xf(yz))u)v)w)$, 105 and 7_1
110. $f(f(f(f(f(f(f(ab)a)a)b)a)f(g(b)g(a)))g(b))\neq y \quad y\neq e$, 108 and 8_3
111. $f(f(f(f(f(f(ab)a)f(ab))a)f(g(b)g(a)))g(b))\neq e$, 109 and 110_1
112. $f(f(f(f(xy)z)u)v)=f(f(f(xf(yz))u)v)$, 104 and 7_1
113. $f(f(f(f(f(f(ab)a)f(ab))a)f(g(b)g(a)))g(b))\neq y \quad y\neq e$, 111 and 8_3
114. $f(f(f(f(f(ab)a)f(f(ab)a))f(g(b)g(a)))g(b))\neq e$, 112 and 113_1
115. $f(f(f(f(ab)a)f(f(ab)a))f(f(g(b)g(a))g(b))\neq e$, 114 and 70_2
116. $f(f(f(f(ab)a)f(f(ab)a))f(f(g(b)g(a))g(b))\neq y \quad y\neq e$, 115 and 8_3
117. $f(g(y)g(x))g(f(xy))$, 47 and 7_1
118. $f(f(g(y)g(x))z)=f(g(f(xy))z)$, 117 and 9_1
119. $f(uf(f(g(y)g(x))z))=f(uf(g(f(xy))z))$, 118 and 10_1
120. $f(f(f(f(ab)a)f(f(ab)a))f(g(f(ab))g(b)))\neq e$, 119 and 116_1
121. $f(g(x)z)=f(f(xx)z)$, 90 and 9_1
122. $f(uf(g(x)z))=f(uf(f(xx)z))$, 121 and 10_1

123. $f(f(f(f(ab)a)f(f(ab)a))f(g(f(ab))g(b))) \neq y \quad y \neq e$, 120 and 8_3
124. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(ab)f(ab))g(b))) \neq e$, 122 and 123_1
125. $f(wf(f(xf(yz))u)) = f(wf(f(f(xy)z)u))$, 100 and 10_1
126. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(ab)f(ab))g(b))) \neq y \quad y \neq e$, 124 and 8_3
127. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(f(ab)a)b)g(b))) \neq e$, 125 and 126_1
128. $f(uf(f(xy)z)) = f(uf(xf(yz)))$, 69 and 10_1
129. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(f(ab)a)b)g(b))) \neq y \quad y \neq e$, 127 and 8_3
130. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(ab)a)f(bg(b)))) \neq e$, 128 and 129_1
131. $f(zf(uf(yg(y)))) = f(zf(ue))$, 20 and 10_1
132. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(ab)a)f(bg(b)))) \neq y \quad y \neq e$, 130 and 8_3
133. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(a)b)a)e)) \neq e$, 131 and 132_1
134. $f(uf(xe)) = f(ux)$, 2 and 10_1
135. $f(f(f(f(ab)a)f(f(ab)a))f(f(f(a)b)a)e)) \neq y \quad y \neq e$, 133 and 8_3
136. $f(f(f(f(ab)a)f(f(ab)a))f(f(ab)a)) \neq e$, 134 and 135_1

12 contradicts 136

Notes added in proof

1. In this paper we intend *fully resolved* sets to be fully factored also.
2. The reader may wish to note that in subsequent work we reserve the term *general* for clauses or terms and use *conservative* instead of *general* for inference systems, in order to avoid possible confusion arising from some misleading connotations of *general* when used in connection with inference systems.
3. A critical difference between functional-reflexive systems defined here as well as in Wos and Robinson (1968c) and those treated in Robinson and Wos (1968b), is that only $h+1$ functional-reflexivity unit clauses are required, where h is the number of function letters in the vocabulary of S; whereas arbitrarily many instances of reflexivity may be required to satisfy the earlier, weaker completeness result.