CONF-790230--1

MASTER

GENERATION AND VERIFICATION OF FINITE MODELS AND COUNTEREXAMPLES USING AN AUTOMATED THEOREM PROVER ANSWERING TWO OPEN QUESTIONS

by

Steve Winker

Prepared for

Fourth Workshop on Automated Deduction

Austin, Texas

February 1-3, 1979

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

U of C-AUA-USDOE

ARGONNE NATIONAL LABORATORY, ARGONNE, ILLINOIS

Operated under Contract W-31-109-Eng-38 for the

U. S. DEPARTMENT OF ENERGY

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

The facilities of Argonne National Laboratory are owned by the United States Government. Under the terms of a contract (W-31-109-Eng-38) between the U. S. Department of Energy, Argonne Universities Association and The University of Chicago, the University employs the staff and operates the Laboratory in accordance with policies and programs formulated, approved and reviewed by the Association.

MEMBERS OF ARGONNE UNIVERSITIES ASSOCIATION

The University of Arizona	Kansas State University	The Ohio State University
Carnegie-Mellon University	The University of Kansas	Ohio University
Case Western Reserve University	Loyola University	The Pennsylvania State University
The University of Chicago	Marquette University	Purdue University
University of Cincinnati	Michigan State University	Saint Louis University
Illinois Institute of Technology	The University of Michigan	Southern Illinois University
University of Illinois	University of Minnesota	The University of Texas at Austin
Indiana University	University of Missouri	Washington University
Iowa State University	Northwestern University	Wayne State University
The University of Iowa	University of Notre Dame	The University of Wisconsin

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately-owned rights. Mention of commercial products, their manufacturers, or their suppliers in this publication does not imply or connote approval or disapproval of the product by Argonne National Laboratory or the U. S. Department of Energy.

For 77% Reduction

GENERATION AND VERIFICATION OF FINITE MODELS AND COUNTEREXAMPLES USING AN AUTOMATED THEOREM PROVER ANSWERING TWO OPEN QUESTIONS*

by

Steve Winker

Northern Illinois University

ABSTRACT

Two open questions in ternary Boolean algebras [1, 2,6] were answered with the aid of an existing automated theorem-proving program without recourse to any additional programming [6]. The new automated theorem-proving techniques developed in answering the open questions are presented in this paper; essentially the existing theorem prover is used in a nonstandard way to seek and verify small finite models and counterexamples for a first order axiom system. Exhibiting a model of an axiom system proves it consistent; this facility complements traditional theorem-proving methods which can only prove inconsistency.

1. INTRODUCTION

The solution of two open questions in mathematics with the aid of an existing automated theorem-proving program exhibits progress toward the long standing goal that automated theorem provers be of aid to mathematicians doing research. Traditionally automated theorem provers have only focused on the attempt to prove a theorem true, while neglecting the search for a counterexample. Techniques (within the context of cur existing theorem-proving program) are described in this paper for the construction of small finite counterexamples in particular and models in general. The user must make some decisions about what sort of model to seek, but much of the work involved in searching for models can be done automatically. To repeat, no reprogramming is required to further a given model search or to attack a new problem or new set of axioms.

The two open questions answered concern independence of axioms in an axiomatization of "ternary Boolean algebras" by A. A. Grau [2]; the results are described fully in Winker and Wos [6]. It is the purpose of this paper to present the new model-finding techniques in detail. No new programming was required to implement the model-finding techniques; rather, the use of certain special clauses (see Appendix II) enabled our existing theorem prover to do the desired operations. The techniques will be discussed in the context of the question to which they were first applied: the independence of axiom 2 of the Grau ternary Boolean algebra axioms (Appendix II) from the remaining axioms.

In the interest of clarity the automatic verification of a single completely defined model will be discussed first; the model search techniques, which are more involved, are deferred until section 3.

2. VERIFICATION OF A FINITE MODEL

This discussion will begin by considering theoretical aspects of model verification before proceeding with a detailed description of "model verification runs". First consider how a model may be specified. A finite model for Grau axioms[†] 1, 3, 4, and 5, and violating axiom 2, for example, may be specified by giving a set of elements and full tables of values for the functions F and G on those elements. One must then verify that

(1) these values are consistent with axioms 1, 3, 4, and 5, and

(2) that axiom 2 is violated.

Axiom 2 is an equality. In order for it to be violated, the two sides must be unequal for some values (X,Y) and for this it is required that two elements of the model be demonstrably unequal. This is not a trivial matter to arrange. In particular, A¬=B cannot be derived from a set of equalities in A and B; equality of all elements is consistent with *any* set of positive equalities. Instead of deriving A¬=B one must prove A¬=B to be consistent with the equalities. Consistency cannot be proven by the refutation techniques of traditional automated theorem proving, and so new techniques are needed.

The method of "complete sets of reductions" [3] enables one to prove consistency of, for example, A¬=B with certain sets of equalities, and thus provides a starting point for the new techniques. Essentially, if a set of demodulators [8] form a complete set of reductions and do not demodulate A and B to the same term, then A¬=B plus those demodulators form a satisfiable set of clauses. The procedures given in [3] for generating and

* This work was supported in part by NSF grant MCS77-02703, and in part was performed under the auspices of the U.S. Department of Energy while in residency at Argonne National Laboratory.

† Grau Ternary Boolean Algebra Axioms (see also Appendix I):

1: F(V,W,F(X,Y,Z))=F(F(V,W,X),Y,F(V,W,Z))
2: F(Y,X,X)=X
3: F(X,Y,G(Y))=X
4: F(X,X,Y)=X
5: F(G(Y),Y,X)=X

Campus Printing

testing complete sets of reductions are easily performed using the standard paramodulation [7] and demodulation [8] features of our theorem-proving program.

Unfortunately not every system of equalities yields a finite complete set of reductions (by undecidability of the word problem) and even a finite set may be unmanageably large. Indeed in the ternary Boolean algebra problems under consideration, application of the standard procedure for generation of a complete set of reductions from the axioms [3] yielded a set of equalities which exceeded time and memory limitations. This difficulty was overcome as follows: set up another, in some sense simpler set of equalities to define a finite model, prove that the simpler equalities form a complete set of reductions, and finally prove that the original axioms are necessarily satisfied in the model so defined. Two questions then arise: first, how does one choose the simpler set of equalities; second, how does one then verify the axioms? Given a specific model, a simpler set of equalities could be obtained by removing complex equalities (e.g. axiom 1†) and adding simple ground equalities (e.g. those of Appendix IB) to fill in for the removed complex equalities in defining the function tables. Actually the models were found by the methods of section 3, which yield simple equalities anyway.

Verification of the axioms is done automatically in a "model verification run"; model verification runs will now be discussed in detail. To specify a model of an equational system, one must specify a set of elements (considered to be distinct) and functions on those elements corresponding to the functions of the system. The following requirements must be satisfied:

(1) Each function must be well-defined.

(2) Each function must be closed.

(3) Each axiom of the system must be satisfied in each instance.

A function on a finite set of elements may be specified by simply tabulating its values. The function tables are supplied to our program in the form of a set of "function defining equalities"; for an example see the model given in Appendix IB. The value of F(t1,...tn)), where t1,...tn are model elements, is then defined to be the result of demodulating F(t1,...tn), using the set of "function defining equalities" as demodulators. For example, in the model of Appendix IB, F(A,G(A),G(A)) receives the value G(A); G(G(G(A))) receives the value G(A); G(G(A)) (G applied to the element G(A)) receives the value G(G(A)) because G(G(A)) does not demodulate. In this way evaluation of functions of model elements is done using our existing demodulation routine. Observe that one equality may stand for several function table entries.

†See Appendix I

Closure may be tested by forming, for each function F and each possible n-tuple t1...tn of model elements, the term F(t1...tn), and then demodulating each such term. If only model elements are obtained, F is closed. Clauses for testing closure are given in Appendix IIB.

Well-definedness must hold for functions of a model: any function of given model elements must be given a unique value. Equivalently, demodulation of a term F(t1,...tn) where the arguments are model elements must yield a unique result no matter how the demodulation is done. The simplest way to guarantee a unique result ("unique termination") is to verify that the "function defining equalities" form a "complete set of reductions". This verification may be done by a paramodulation-and-demodulation procedure as described in [3]. An alternative method for verifying unique termination is outlined in Appendix V but has not been needed for the models examined so far. Note that even though the axiom system being modeled may not yield a complete set of reductions, the function defining equalities for a particular model of that system can still form a complete set of reductions.

The condition that each axiom is true in each instance is checked by forming each instance, demodulating both sides of the equality, and checking that the two sides become identical in each case. For example, substituting A/V, G(G(A))/W, G(A)/X, A/Y, G(G(A))/Z in Grau Axiom 1 yields, after demodulation, A=A, verifying axiom 1 in this instance. A method for generating all the instances is given in Appendix IIA. Our theorem prover demodulates each automatically, then tests each for subsumtion by X=X (equivalently, for identity of the two sides). Checking satisfaction of Axiom 1 formed the bulk of the work of verifying each Grau model; 3 to the 5th power = 243 instances must be checked for a three-element model. (The amount of checking can in some cases be reduced if needed by symmetry and other considerations; see Appendix III).

Note: If the function defining equalities form a complete set of reductions, no axiom included among those equalities (for example axioms 3-5 in the model of Appendix IB) needs to be checked for satisfaction in all instances. Proof: The check is trivial. Apply the demodulator s=t to the axiom instance su=tu giving tu=tu.

This "trivial check" is only valid when the axiom is one of the function defining equalities and the function defining equalities form a complete set of reductions. In checking axiom 1 in the model of Appendix IB, for example, axiom 1 must not be used as a demodulator, because axiom 1 is not one of the function defining equalities. As a simple example of an invalid "trivial check" consider a model with elements A and B and equalities J(J(J(X)))=X for all X, J(A)=B, and J(B)=A, with the latter two forming the complete set of reductions. J(J(J(X)))=X seems to be satisfied when the "trivial check" is applied; however when the latter two equalities are applied to J(J(J(A))), B

Campus Printing

For 77% Reduction

is obtained, not A. In doing the "trivial check" the implicit assumption is made that demodulating using J(J(J(X)))=X gives the same result as demodulating using the other equalities; this is not true however because the three equalities together do not form a complete set of reductions.

3. SEARCHING FOR FINITE MODELS

The NIUTP automatic theorem prover [4,5,7,8] was used extensively in searching for the models we found as well as in verifying them. The model searches proceeded in three stages: preliminary paramodulation runs, partial-model runs, and final model-verification run.

Preliminary paramodulation runs can be used to derive consequences of the axioms being modeled, to suggest what equalities do not follow from the axioms, and to test the effect of adding various function-defining or other equalities to the axiom system. For example:

1) The equality F(X,Y,X)=X was derived by paramodulation from axioms 1, 3, 4, and 5 of Appendix IA, filling in part of the function table for F (in any model of axioms 1, 3, 4, and 5).

2) The failure to derive G(G(X))=X from the above system (axioms 1, 3, 4, and 5) suggested that models be sought in which G(G(A))¬=A for some A (see Section 4, Example of a Model Search).

3) Addition of the axiom F(X,Y,Z)=F(X,Z,Y) to the above system was rejected because paramodulation then derived G(G(X))=X (G(G(A))¬=A for some A was desired).

4) Addition of G(G(G(X)))=G(X) to the above system yielded no undesirable paramodulants; this system thus seemed promising and was studied further by means of "partial-model runs".

Partial-model runs are begun when a model has been partially specified: that is, when the set of model elements (e.g. (A, G(A), G(G(A)))) has been selected, and most but not all entries in the function tables have been filled in.

A partial-model run is similar to a model verification run. The similarities:

A set of model elements is input.

A set of function-defining equalities is input.

Closure, well-definedness, and satisfaction of the axioms in all instances are tested as in model verification.

The differences:

The function defining equalities do not completely define the functions. Rather, the values for some terms (called "undecided terms") are left unspecified.

The closure test will not indicate closure, but rather will yield a list of the undecided terms.

In a partial-model run, the check for satisfaction of the axioms in all instances may yield any or all of the following:

1) Equality of a model element to itself -- indicating satisfaction of an axiom in the particular instance tested.

2) Equality of two model elements -- indicating that the partially defined functions already do not satisfy the axiom tested.

3) Equality of an undecided term to a model element -- any such equality is input as a function defining equality in subsequent partial-model and model-verification runs.

4) More complex ground equalities involving undecided terms -- may be used to eliminate some possible values for the undecided terms.

Example of a partial-model run: Axioms 1, 3, 4, and 5 of Appendix I, plus the equality G(G(G(X)))=G(X), yield the first seven equalities in Appendix IB; these were used as defining equalities for a partial-model run. The model elements A, G(A), and G(G(A)) were also input. The undecided terms were F(A,G(A),G(A)), F(A,G(G(A)),G(G(A))), and F(G(G(A)),A,A). The check for satisfaction of axiom 1 in all instances yielded (among others) the equalities

F(A,G(G(A)),G(G(A)))=A
F(G(G(A)),A,A)=G(G(A))
F(F(A,G(A),G(A)),G(G(A)),A)=A

The first two were input as function defining equalities in subsequent runs. The last equality eliminates the possibility F(A,G(A),G(A))=G(G(A)) (as this and axiom 4 would demodulate the last equality to G(G(A))=A indicating violation of axiom 1). Each of the other two possibilities, F(A,G(A),G(A))=A and F(A,G(A),G(A))=G(A), led to a valid model. This ends the example.

Finally, when enough information has been gained from paramodulation and partial-model runs to specify a likely model completely, that model is verified in a model-verification run (as described in section 2).

Notes on searching for models:

1) When making a partial-model run, it appears desirable to instantiate axioms using only the elements which are expected to be in the final model. Instantiation using "undecided terms" would yield more axiom instances to check and more complicated ground equalities.

2) A partial-model run with too many undecided terms may yield an unmanageable quantity of complex ground equalities. In this case paramodulation runs might be used to find the consequences of proposed function-defining equalities.

Campus Printing

When more function-defining equalities have been tested, found seemingly acceptable, and added to the partial model, partial-model runs may again be attempted.

3) a partial-model run may be considered "promising" if no equality between distinct model elements is derived. Each "promising" partial-model run in the author's brief experience has led to a valid model. However the author doubts that a "promising" partial-model run guarantees that there is a valid model: it might not be possible to complete the function tables consistently with the axioms being modeled.

4) It is possible in modeling certain axiom systems (e.g. semigroups, having associativity as the only axiom) to include all the axioms, plus the function defining equalities, in a complete set of reductions. In this case none of the axioms need to be tested for satisfaction in each instance (see the note at the end of Section 2) and neither partial-model nor model-verification runs are required; the model search may be conducted using paramodulation runs only.

4. AN EXAMPLE OF A MODEL SEARCH

Some of the automated theorem prover runs made in searching for the model of Appendix IB are listed below to indicate the degree of our reliance on the computer. As might be expected, the search includes tests which appear inconclusive or unnecessary in retrospect.

1) Paramodulation runs proved Grau axioms 4 and 5 (see Appendix I) from axioms 1, 2, and 3, and incidentally derived G(G(X))=X from axioms 1, 2, and 3.

2) A paramodulation run attempting to prove axiom 2 from axioms 1, 3, 4, and 5, proved neither axiom 2 nor G(G(X))=X. Incidentally, F(X,Y,X)=X was derived.

The failure to prove axiom 2 motivated the search for a counterexample. The fact that G(G(X))=X was not derived suggested that a model violating G(G(X))=X be attempted. Such a model would necessarily violate axiom 2. It could be based on one generator (an A for which G(G(A)¬=A) rather than two (as A and B for which F(B,A,A)¬=A), possibly requiring fewer elements, fewer defining relations, and less computer time for verification.

3) Paramodulation run seeking consequences of axioms 1, 3, 4, and 5, in conjunction with G(G(G(X)))=G(X), G(F(X,Y,Z))=F(G(X),G(Y),G(Z)), and F(X,Y,Z)=F(X,Z,Y). Axiom 2 was not proved, but the last of these equalities together with axioms 3 and 5 yielded G(G(X))=X.

Because a model with G(G(A))¬=A was being sought, the last equality was not used for subsequent models. The possibility that G(G(G(X)))=G(X) might imply G(G(X))=X was not tested at this time. (The extra equalities were added in an attempt to add enough structure to help get a model, but without satisfying axiom 2. Equalities known to be true in ternary Boolean algebras were added so that the known structure would be approached; for example, G(G(G(X)))=G(X) is a weakening of G(G(X))=X.)

4) A paramodulation run deriving consequences of axioms 1, 3, 4, and 5, in conjunction with G(F(X,Y,Z))=F(G(X),G(Y),G(Z)) and F(A,X,G(G(A)))=A, derived no undesirable consequences. The latter equality, when X=A, is an instance of axiom 4; when X=G(A) it is an instance of axiom 3; the author hoped the generalization to X=G(G(A)) as well, violating axiom 2, would lead to a model violating axiom 2.

5) Partial-model run, using the first seven function defining equalities of Appendix IB. The next two equalities were derived (in checking satisfaction of axiom 1).

6) Partial-model run, using all but the eighth equality of Appendix IB. The eighth equality was derived.

The ninth equality was suggested, before runs 5 and 6 were made, by an examination of the proof of G(G(A))=A from axioms 1, 2, and 3. This proof used the instance F(A,G(G(A)),G(G(A)))=G(G(A)) of axiom 2; if this term had the value A instead, G(G(A))=A would not be proven.

7) Model-verification run verifying the model of Appendix IB.

One goal of future work is to automate more of the interactive process illustrated here, conceivably following the general plan of Fig. 1.

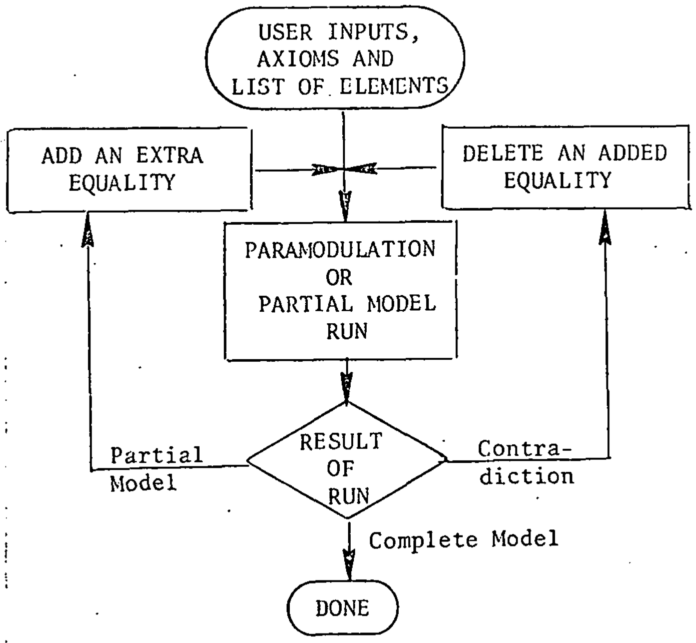

Fig. 1. Flowchart for Model Search

5. THE ROLE OF THE AUTOMATED THEOREM PROVER

The theorem prover served as a "logical calculator", rapidly performing calculations which would have been laborious if done by hand. The calculations were of two types: first, given a set of equalities, obtain a list of consequences; second, test the validity of a model or partial model.

The program helped the author to operate effectively with an unfamiliar axiom system.

These benefits were obtained without recourse to new programming, attesting to the generality and flexibility of the existing theorem proving program and techniques.

The program did not decide what sort of model to seek; this was up to the user, who made the intelligent decisions. Symmetry and other arguments had to be made by the user (see Appendix III).

The automated theorem prover is limited in the number of instances of an axiom which can be verified. An axiom containing m variables, will when there are n distinct elements have n to the m-th power instances to be checked; when n to the m-th power exceeds 500 to 5000 the cost of computer time becomes high. Presumably various methods of checking many instances at once can be developed to deal with larger models (see Appendix III); one would hope that some of these will be general-purpose rather than problem-dependent.

6. ACKNOWLEDGEMENTS

The author wishes to thank Dr. L. Wos for the encouragement, his enthusiasm in using the techniques of this paper to seek more models [6], our discussions concerning refinements of these methods (Appendix III), and his advice concerning this paper; and to thank Dr. Ross Overbeek and many others for the man years of work devoted to designing, writing, maintaining, and improving the "NIUTP" theorem proving program used in this work.

APPENDIX I. GRAU TERNARY BOOLEAN ALGEBRAS AND MODELS

A. AXIOMS FOR A GRAU TERNARY BOOLEAN ALGEBRA

The ternary boolean algebra discussed herein was first presented in [2]. The axioms are:

Axiom 1:	F(V,W,F(X,Y,Z))=F(F(V,W,X),Y,F(V,W,Z))
Axiom 2:	F(Y,X,X)=X
Axiom 3:	F(X,Y,G(Y))=X
Axiom 4:	F(X,X,Y)=X
Axiom 5:	F(G(Y),Y,X)=X

Axioms 1, 2, and 3 imply 4 and 5 [1,6]. The techniques of this paper were used to establish that Axiom 2 is independent of axioms 1, 3, 4, and 5 and that Axiom 3 is independent of Axioms 1, 2, 4, and 5 [6]. Other results, concerning the above axioms, and discovered using the techniques of this paper, appear in [6].

B. A MODEL FOR AXIOMS 1, 3, 4, AND 5 VIOLATING AXIOM 2

The following form a complete set of reductions, defining closed functions F and G on three elements A, G(A), and G(G(A)). All variables are universally quantified.

F(X,Y,G(Y)) = X	(Axiom 3)
F(X,X,Y) = X	(Axiom 4)
F(G(Y),Y,X) = X	(Axiom 5)
F(X,Y,X) = X	(Consequence of Axioms 1 and 4)
G(G(G(X))) = G(X)	Special hypothesis
F(X,G(G(Z)),G(Z)) = X	Implied equality
F(G(Z),G(G(Z)),X) = X	Implied equality
F(A,G(G(A)),G(G(A))) = A	Implied equality
F(G(G(A)),A,A) = G(G(A))	Implied equality
F(A,G(A),G(A)) = G(A)	Special hypothesis

The functions thus defined satisfy axiom 1 in all instances, as was demonstrated using our automated theorem-proving program. The last three equalities violate axiom 2; the model thus shows axiom 2 to be independent of axioms 1, 3, 4, and 5.

C. A MODEL FOR AXIOMS 1, 3, 4, AND 5, VIOLATING AXIOM 2, BUT SATISFYING G(G(X)) = X

Elements: A, G(A), C, G(C)

Function defining equalities:

Unit clauses	F(X,Y,G(Y)) = X;	F(X,X,Y) = X;
F(G(Y),Y,X) = X;	F(X,Y,X) = X;	G(G(X)) = X;
F(X,G(Z),Z) = X;	F(Z,G(Z),X) = X;	
F(A,C,C) = A	and seven variants;	
F(A,C,G(A)) = A	and seven variants.	

The model has 8-fold symmetry given by the permutations (A G(A)), (C G(C)), and (A C)(G(A) G(C)). The variants of the last two equalities are obtained by applying these symmetries; thus F(G(A),C,C) = G(A). The equality F(A,C,C) = A and its variants violate axiom 2.

APPENDIX II. CLAUSES USED FOR MODEL CHECKING RUNS

A. GENERATION OF AXIOM INSTANCES

All instances of axiom 1 (distributive axiom) which may be formed by substituting the elements A, G(A), and G(G(A)) for the variables, may be generated by forming all hyperresolvents of the following clauses:

Units Q(A); Q(G(A)); Q(G(G(A))); nucleus
¬Q(V) ¬Q(W) ¬Q(X) ¬Q(Y) ¬Q(Z) or
EQUAL (F(V,W,F(X,Y,Z)), F(F(V,W,X),Y,F(V,W,Z)))

The general principles are: Write a clause Q(e) for each element e in the proposed model. These clauses serve as hyperresolution electrons. Write the hyperresolution nucleus as the disjunction of the equality to be instantiated and, for each variable v appearing in the equality, the literal ¬Q(v). Hyperresolution will generate n to the m-th power instances, where n is the number of model elements and m is the number of variables appearing in the equality being checked.

Note: It would appear that generation of instances could be done almost equally well by P1 deduction, or by forward chaining using an appropriate "nucleus" like

Q(X) -> (Q(Y) -> ... -> EQUAL(...,...)...)) .

The demodulation (simplification, reduction) of the instances could then be achieved by any general system of algebraic simplification. The author would appreciate hearing from other workers who repeat these experiments with their programs.

B. TESTING OF CLOSURE

To test closure of a ternary function F, on elements A, G(A), and G(G(A)), form all hyperresolvents of the following clauses:

```
Q(A);  Q(G(A));  Q(G(G(A)));  and nucleus
¬Q(X)  ¬Q(Y)  ¬Q(Z)  or  Q(F(X,Y,Z)) .
```

Demodulate each derived clause using the equalities defining F. If each resulting clause is identical to Q(e) for some element e in the model, then F is closed. Otherwise F is not closed: for example if Q(F(A,G(A),G(A)) is derived and does not simplify, F(A,G(A),G(A)) has not been defined. The general principles are: Write a clause Q(e) for each element e of the proposed model. Write the nucleus (for an n-ary function F) as:

¬Q(V1)... ¬Q(Vn) or Q(F(V1,...,Vn)) .

APPENDIX III. METHODS FOR REDUCING RUN TIME OF PARTIAL-MODEL AND MODEL-VERIFICATION RUNS

If a fairly large model is being verified, the number of instances of axioms to be verified may be reduced by symmetry and other considerations.

For example, the four element model of Grau axioms 1, 3, 4, and 5, described in Appendix IC, has a set of symmetries mapping any element into any other. Thus in checking the instances of axiom 1, only instances in which A is substituted for V need be checked; instances in which G(A), C, or G(C) is substituted for V will behave analogously. The clauses used to generate the restricted set of instances were:

```
Q(A);  Q(G(A));  Q(C);  Q(G(C));
Q2(A);  and nucleus
¬Q2(V)  ¬Q(W)  ¬Q(X)  ¬Q(Y)  ¬Q(Z)  or
EQUAL (F(V,W,F(X,Y,Z)), F(F(V,W,X),Y,F(V,W,Z))).
```

The use of ¬Q2(V) and Q2(A) causes only A to be substituted for V. The set of instances to be checked may be further reduced by examining what happens when V=W or V=G(W). When V=W,

EQUAL (F(V,V,X),Y,F(V,V,Z)), F(V,V,F(X,Y,Z)))

reduces to EQUAL(F(V,Y,V),V) by Axiom 4, and thence to EQUAL(V,V) by F(X,Y,X)=X. Thus (assuming the defining equalities form a complete set of reductions) instances where V and W are given the same value need not be checked. Similarly when V=G(W),

```
EQUAL (F(F(G(W),W,X),Y,F(G(W),W,Z)),
       F(G(W),W,F(X,Y,Z)))
```

reduces to EQUAL(F(X,Y,Z),F(X,Y,Z)) by Axiom 5; again instances satisfying V=G(W) (e.g. G(A)/V, A/W; A/V, G(A)/W if G(G(A))=A) need not be checked individually. Thus in checking the four element model, only A/V with C/W or G(C)/W need be checked in full; this may be done by generating all hyperresolvents of the following clauses:

```
Q(A);  Q(G(A));  Q(C);  Q(G(C));
Q2(A);
Q3(C);  Q3(G(C));  and nucleus
¬Q2(V)  ¬Q3(W)  ¬Q(X)  ¬Q(Y)  ¬Q(Z)  or
EQUAL (F(V,W,F(X,Y,Z)), F(F(V,W,X),Y,F(V,W,Z)))
```

It is hoped that such tricks can be incorporated in a general-purpose, fully automated package, but at present the desired tricks are discovered and set up by the user. Tricks of various kinds will presumably become very important in verifying large models, as the numbers of instances of certain axioms become enormous.

APPENDIX IV. CLAUSES FOR GENERATING A LIST OF ELEMENTS AND A FUNCTION TABLE

To generate a list of elements of a model, generate all hyperresolvents of the following clauses and their hyperresolvent consequences (demodulating each using the function defining equalities):

```
¬Q(X)  or  Q(G(X))
¬Q(X)  ¬Q(Y)  ¬Q(Z)  or  Q(F(X,Y,Z))
...(one such clause for each function, as in
   closure test)
and units  Q(A);  Q(B); ...
(one such clause for each generator of the
 model).
```

This is useful when a model has been found using paramodulation runs only (Section 3, Note 4).

To generate a function table, generate all hyperresolvents of the following clauses (demodulating each using the function defining equalities):

```
¬Q(X)  ¬Q(Y)  ¬Q(Z)  or  PF(X,Y,Z,F(X,Y,Z))
(for a ternary function F)
and units  Q(A);  Q(G(A)); ...
(one such clause for each element of the model).
```

The derived "PF" clauses give the function table entries for "F".

Campus Printing

For 77% Reduction

APPENDIX V. AN ALTERNATIVE METHOD OF CHECKING WELL-DEFINEDNESS IN A MODEL-VERIFICATION RUN

If the function defining equalities for a proposed model do not form a complete set of reductions, the following test for well-definedness may be used instead. (The reader is urged to contact the author for details of this method if desired; space considerations do not permit a full presentation here.) First, design the demodulation algorithm to satisfy two criteria:

1) In demodulating a given term, fully demodulate each subterm before applying demodulators to the full term. For example, in demodulating F(G(G(G(A))),A,A), demodulate G(G(G(A))) to G(A) before applying a demodulator to the term whose major symbol is F.

2) Demodulate a given ground term in the same way each time it appears. If two demodulators apply, choose the same one each time. These criteria insure that demodulation will act as a well-defined function.

Then check the function defining equalities for satisfaction in each instance, using the method applied to Grau axiom 1 in Appendix IIA with the above demodulation algorithm.

APPENDIX VI. APPLICATION TO MODELS OF FIRST ORDER NON-EQUATIONAL SYSTEMS

The techniques of this paper might be applied to non-equational systems by (1) rewriting each predicate, "OR", and "NOT" as functions; (2) supplying function defining equalities for the above which yield "T" and "F" as values, e.g. LT(1,2)=T, OR(T,F)=T, NOT(T)=F; (3) seeking and verifying models as in the preceding part of the paper, omitting T and F from the list of elements. The author has not tried this technique and makes no claim for its practical utility.

REFERENCES

1. Chinthayamma, Sets of independent axioms for a ternary Boolean algebra (preliminary report), Notices of the American Math. Soc., Vol. 16, No. 4, p. 654, June 1969.

2. A. A. Grau, Ternary Boolean Algebra, Bulletin of the American Math. Soc., Vol. 53, No. 6, pp. 567-572, June 1947.

3. D. E. Knuth and P. B. Bendix, Simple Word Problems in Universal Algebras, Computational Problems in Abstract Algebra, pp. 263-297, (Edited by J. Leech), Pergamon Press, Oxford (1970).

4. J. D. McCharen, R. A. Overbeek, and L. Wos, Problems and Experiments for and with Automated Theorem-Proving Programs, IEEE Transactions on Computers, Vol. C-25, No. 8, August 1976, pp. 773-782.

5. R. A. Overbeek, "An implementation of hyper-resolution", Comput. Math. Appl., Vol. 1, pp. 201-214, 1975.

6. Steve Winker and L. Wos, Automated Generation of Models and Counterexamples and its Application to Open Questions in Ternary Boolean Algebra, Proc. Eighth Int. Symposium on Multiple-Valued Logic, pp. 251-256. Rosemont, Illinois (1978); IEEE (1978).

7. L. Wos and G. A. Robinson, Paramodulation and Set of Support, Proc. IRIA Symposium on Automatic Demonstration, pp. 276-310. Versailles, France (1968); Springer-Verlag (1970).

8. L. Wos, G. A. Robinson and D. F. Carson, The Concept of Demodulation in Theorem Proving, J. Assn. Comput. Mach. 14, 4 687-709 (October 1967).

Campus Printing

CMU