Automated Reasoning and Enumerative Search, with Applications to Mathematics*

Jian Zhang, Institute of Software, Academia Sinica, P.O. Box 8718, Beijing 100080, P.R. China

Larry Wos, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois 60439-4801, U.S.A.

JAN 31 1992

ABSTRACT

JAN 31 1992

More and more mathematical problems are being solved with the aid of computers. In this paper, we examine the applications of reasoning and search programs to mathematics. It is also shown that the combination of these two techniques can solve mathematical problems more effectively.

INTRODUCTION

Reasoning and search are two of the most important activities involved in problem solving. In this paper, we first focus on the value of using an automated reasoning program as a research assistant in attacking questions from mathematics. Indeed, a number of open questions have been answered with the use of such a program, including questions from finite semigroups posed by Kaplansky, from equivalential calculus posed by Kalman, from combinatory logic posed by Smullyan, and from sentential calculus posed by Scott. In answering the various questions, an automated reasoning program was asked to find algebraic constructs, to consider mappings in combination, to provide a complete characterization of an infinite set of deducible theorems, to refute conjectures, to construct needed objects, to find shorter proofs, and to discover a new axiom system.

Also central to this paper is the case we make (in Section 4) for the symbiotic relation that can exist between enumerative search (ES) and automated reasoning (AR). Indeed, an enumerative search can profitably be used as a search strategy by an automated reasoning program, and conversely an automated reasoning program can be used to sharply reduce the size of an enumerative search.

AUTOMATED REASONING

From a 1955 perspective, the attempt to use a single computer program to answer questions from various areas of mathematics and logic by applying reasoning would be viewed, at best, as audacious and, at worse, as absurd. After all, how could one program cope with concepts that include group, ring, topological space, homomorphism, and combinator? Further, how could this mythical program be instructed to apply the logical reasoning required to prove theorems from essentially unrelated fields? Perhaps most challenging, what means could be found to adequately direct the program's search for the needed steps of a proof and to effectively restrict its search to the consideration of a reasonable subset of the usually vast number of possible conclusions?

As it turns out, the 1955 evaluation of audacious or even absurd--although understandable--was incorrect. Instead, from the early programs that offered essentially no promise of proving interesting theorems, there evolved various general-purpose, automated reasoning programs whose use resulted in answering open questions from a wide variety of essentially unrelated fields.

For example, Kaplansky's question concerning the possible existence of a finite *semigroup* in which one type of mapping is present and another type is absent was answered when an automated reasoning program found such a semigroup of order 83 (consisting of 83 elements). That same program then proved that the smallest such semigroup has order 7 and that, of the more than 800,000 semigroups of that order, only four possess the desired properties.

Kalman's questions concerning the possibility of using any of six formulas by itself as a complete axiom

MASTER

EB

system for *equivalential calculus* were answered when an reasoning program helped obtain a total characterization of the infinite set of deducible theorems for four of the formulas and a lengthy proof for each of the other two. The characterization showed that each of the four formulas is too weak to provide a complete axiomatization, and the two proofs established that the remaining two formulas are each strong enough. The proofs, respectively of length 84 and 159 steps and of complexity 71 and 103 if measured in terms of the longest included formula, refuted the conjecture that each was too weak.

Smullyan's questions focusing on the presence or absence of the strong fixed point property for various subsets (called fragments) of *combinatory logic* were answered by constructing (where appropriate) a fixed point combinator and, elsewhere, providing the key information to show that the property could not hold.

Scott's question regarding the existence of a sharply different proof for the completeness of the Lukasiewicz axiom system for *sentential calculus* was answered in the affirmative when an automated reasoning program found a 30-step proof of the completeness, contrasting nicely with the original 46-step proof. In addition, the study led to the discovery of a new complete axiom system for sentential calculus.

Single programs now exist that offer sufficient power to answer each of the cited questions--and far more! The most powerful general-purpose reasoning program currently available is called **OTTER**, designed and implemented by Dr. William McCune of the Mathematics and Computer Science Division at Argonne National Laboratory. The program consists of approximately 20,000 lines of C code. It is portable, running well on workstations, personal computers, and the Macintosh. On a Sparcstation, per CPU second, OTTER continually deduces approximately 500 conclusions, and can easily apply more than 4,000 rewrite rules for canonicalization and simplification. This program derives its power in part from the use of the clause language paradigm [Wos 84] [Wos 87] and the use of diverse inference rules and strategies.

ENUMERATIVE SEARCH

The method of enumerative search can be summarized in the following way. Given an abstract *space* (or a set) which consists of a number of *objects*, we wish to find an object x that has a specific property P . Let us call x the *target*. To achieve our intention, we simply examine the objects in some order and check each object to see whether it satisfies P .

In practice, we often need not check all the objects. Instead, some *heuristics* may be employed to reduce the search space. If we know in advance that none of the objects in a subspace satisfies the property P , then this subspace may be skipped. Usually such a subspace is characterized by a certain condition, which will represent all the objects in it. For example, if we are seeking a prime, or a prime having some additional properties, and we know that all primes (except 2) are odd numbers, then the numbers in the set $\{n > 2 \text{ and } n \text{ is even}\}$ may be skipped without missing the target. Such heuristics are essential for improving the efficiency of the search, and make the ES method more practical.

For a problem to be solved completely by a program applying ES, the search space must be finite. With such a goal, we must work in discrete (not continuous) domains. Appropriately constrained to guarantee finiteness, the search space may be a set of natural numbers, a set of 0-1 matrices, or a set of finite groups. For example, we may search for a prime number in some finite set N of natural numbers, or search for a pair of primes p and q in some finite set $N \times N$ such that $m^{\text{FR}} - n = 2$ (such numbers are called *twin primes*). Normally, the search space should admit some ordering so that we can proceed systematically, moving from "smaller" objects to "larger" objects. For example, we might move from 1 to 2, to 3, ... or move from $(0, 0, 0)$ to $(0, 0, 1)$, to $(0, 1, 0)$ to $(0, 1, 1)$. In this way, we can visit all the objects and see whether a target has been found. Also important to ES is that the property P be readily verifiable. Moreover, the property should belong to a single object only, not to a set of objects.

Appropriate candidates for a program applying ES include problems from number theory, combinatorics, algebra, logic, and other similar branches of mathematics. Among the successes obtained with ES are

the classification of all graphs with 10 vertices and the proof that no projective plane exists of order 10.

AR and ES COMBINED

That AR and ES complement each other in many respects can be seen by focusing on ternary Boolean algebra. A *ternary Boolean algebra* [Grau 47] (TBA) is a set S with one ternary function f from $S \times S \times S$ to S and one unary function g from S to S , where the following five axioms hold (for any v, w, x, y, z in S):

- (1) $f(f(v, w, x), y, f(v, w, z)) = f(v, w, f(x, y, z))$
- (2) $f(y, x, x) = x$
- (3) $f(x, y, g(y)) = x$
- (4) $f(x, x, y) = x$
- (5) $f(g(y), y, x) = x$

A ternary Boolean algebra can be transformed into a Boolean algebra, and conversely.

Now suppose we wish to find a TBA that has 4 elements. Without loss of generality, the elements may be denoted as $\{0, 1, 2, 3\}$. The task is to determine the values $g(0), g(1), g(2), g(3)$ and $f(i, j, k)$ for each 3-tuple (i, j, k) , where $0 \leq i, j, k \leq 3$. The total number of these unknowns is $4 + 4^3 = 68$, and each unknown may take the value 0, 1, 2, or 3. With ES, we have a maximum of 4^{68} different choices. However, axioms (2) and (4) can be used to "calculate" some values directly: $f(0, 0, 1) = 0, f(0, 1, 1) = 1, \dots$, reducing the number of unknowns to 40, still too large a number.

We can next use an AR program to derive additional equations to further reduce the number of unknowns, obtaining the following [Winker 78]:

- (6) $f(x, y, x) = x$
- (7) $g(g(x)) = x$
- (8) $f(g(y), x, y) = x$
- (9) $f(x, y, z) = f(y, x, z)$
- (10) $f(x, y, z) = f(x, z, y)$

We know from (7) that the elements are divided into pairs: (a_1, b_1) and (a_2, b_2) , with $g(a_1) = b_1, g(b_1) = a_1, g(a_2) = b_2, g(b_2) = a_2$. Without loss of generality, we may choose $0 = a_1, 1 = b_1, 2 = a_2$, and $3 = b_2$, and the function g is completely determined. Then, the equations (2), (4), (6), (3), (5), and (8) can be used to calculate some values of the function f . At this point, there exist 24 unknowns, some equivalent to others in the sense of equations (9) and (10). The elimination of redundant unknowns reduces their number to 4, and we see how AR can benefit ES.

Conversely, AR can benefit from ES by, for example, using an exhaustive search to find key lemmas. For a second example, ES may play a key role in finding a shorter proof. We thus see how AR and ES can sometimes complement each other.

CONCLUSIONS

The use of automated reasoning and of enumerative search can play a key role in answering some question or solving some problem that has remained open. The combination of the two sometimes offers even greater power. Indeed, the application of an enumerative search by an automated reasoning program led to finding far shorter proofs of two important theorems in equivalential calculus and to finding an unexpectedly elegant proof establishing the completeness of a new axiom system for sentential calculus.

REFERENCES

[Grau 47] Grau, A. "Ternary Boolean algebra," *Bulletin of the Amer. Math. Soc.* 53 (1947) pp. 567-572.

[Winker 78] Winker, S., and Wos, L., "Automated generation of models and counterexamples and its application to open questions in ternary Boolean algebra," in *Proc. 8th Inter. Symp. on Multiple Valued Logic*, IEEE & ACM (1978) pp. 251-256.

[Wos 84] Wos, L., Overbeek, R., Lusk, E., and Boyle, J., *Automated Reasoning: Introduction and Applications*, Prentice-Hall, Englewood Cliffs, N.J. (1984).

[Wos 87] Wos, L., *Automated Reasoning: 33 Basic Research Problems*, Prentice-Hall, Englewood Cliffs, N.J. (1988).

* This work was supported by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U.S. Department of Energy, under Contract W-31-109-Eng-38.

END

**DATE
FILMED**

3 / 11 / 92