AUTOMATED REASONING: REAL USES AND POTENTIAL USES

L. Wo 8

Mathematics and Computer Science Division
Argonne National Laboratory
9700 South Cass Avenue
Argonne, IL 60439

ABSTRACT

An automated reasoning program has provided invaluable assistance in answering certain previously open questions in mathematics and in formal logic. These questions would not have been answered, at least by those who obtained the results, were it not for the program's contribution. Others have used such a program to design logic circuits, many of which proved superior (with respect to transistor count) to the existing designs, and to validate the design of other circuits. These successes establish the value of an automated reasoning program for research and suggest the value for practical applications. We thus conclude that the field of automated reasoning is on the verge of becoming one of the more significant branches of computer science. Further, we conclude that the field has already advanced from stage 1, that of potential usefulness, to stage 2, that of actual usefulness.

To pass to stage 3, that of wide acceptance and use, requires, among other things, easy access to an automated reasoning program and an understanding of the various aspects of automated reasoning. In fact, an automated reasoning program is available that is portable and can be run on relatively inexpensive machines. Moreover, a system exists for producing a reasoning program tailored to given specifications. As for the requirement of understanding the aspects of automated reasoning, much research remains—research aided by access to a reasoning program. A large obstacle has thus been removed, permitting many to experiment with and find uses for a computer program that can be relied upon as a most valuable automated reasoning assistant.

1. INTRODUCTION

Recent successes [3,4,20,23,26,35,24,27,28,6,29] resulting from heavy use of an automated reasoning program demonstrate that the field has advanced from conjectured usefulness to actual usefulness. By reviewing certain of the successes, we provide evidence for individuals from other fields to judge the progress that has occurred. Rather than surveying the field of automated reasoning as a whole, we concentrate on the work of an informally organized team of which the author is a member. Nevertheless, to give the minutest of hints about what is being achieved by others in the field, we cannot resist citing (here and in Section 4) some of the outstanding work of such researchers as R. Boyer and J Moore, W. W. Bledsoe, and J. Siekmann.

The first success realized by the aforementioned team was obtained with much assistance from the program AURA (for Automated Reasoning Assistant) [21]—a program developed jointly at Argonne National Laboratory and Northern Illinois University (ANL/NIU). The success was that of answering certain previously open questions in mathematics [23,26,27], followed by answering various open questions in formal logic [35,24], The results were judged of significant Interest that they were published respectively in a mathematics journal and in a logic journal. Such recognition by researchers having no direct interest in automated reasoning represents a significant breakthrough for the field of automated reasoning, and hence for automated theorem proving where it all began more than 20 years ago. Perhaps this occurrence is the first of many such occurrences In which various open questions will be answered by researchers using an automated reasoning program.

Must you master the various aspects of automated reasoning to answer open questions or, for that matter, to use such a program in an important way? After all, you might argue that, since those who answered the questions are active in automated reasoning, perhaps an automated reasoning program can be used only by those in the field. That this is not the case is proved by additional successes. Researchers having no previous exposure to automated reasoning used a reasoning program to design logic circuits [28] , many of which are superior (with respect to transistor count) to those previously known, and to validate the design of other circuits [29] already in existence. These researchers have provided us with powerful evidence for the curious and for the skeptical. To the former, we can say with confidence that patience and pain may be rewarded with the discovery of a new application for automated reasoning. To the latter, we can say with equal confidence that the years of research, development, implementation, and experimentation have culminated in the availability of a portable computer program [8,9,11,12] that can be used as a

This work was supported by the Applied Mathematical Sciences Research Program (KG-04-02) of the Office of Energy Research of the U.S. Department of Energy under Contract W-31-109-Eng-38.

most valuable automated reasoning assistant. Further, we can say that the field of automated reasoning as a whole has cause for genuine excitement and optimism, for we can now list various real problems that have been solved with an automated reasoning program.

The main objective of this paper is to inform individuals with diverse and unrelated interests of what has occurred and, equally, of what is likely to occur in the field of automated reasoning. We shall discuss the past and the future, and thus establish the position of this field relative to various other fields of computer science and of artificial Intelligence. Some who read this paper may even share our conclusion that the field of automated reasoning is on the verge of becoming one of the more significant branches of computer science. If not this, at least we show that the field has already advanced from stage 1, that of potential usefulness, to stage 2, that of actual usefulness. To pass to stage 3, that of wide acceptance and use, requires, among other things, easy access to an automated reasoning program and an understanding of the various aspects of automated reasoning. We shall discuss this transition as well.

Since many who may read this paper are not directly concerned with automated reasoning itself, we provide background material. Some of you may be surprised to find that an automated reasoning program is now or soon will be useful as an aid for research or for applications. The kinds of assistance that such a program can provide include finding proofs, generating models or counterexamples, suggesting and verifying conjectures, and testing hypotheses. The research and application areas in which a reasoning program can be used include mathematics, formal logic, program verification, logic circuit design, logic circuit validation, chemical synthesis, systems control, database inquiry, and robotics. How can one program—an automated reasoning program—provide so many kinds of assistance and in so many diverse areas? In fact, what is automated reasoning [32]?

2. BACKGROUND

Reasoning is the process of drawing conclusions from given facts or assumptions. The conclusions must follow inevitably and logically from the facts or assumptions. If the conclusions are false, then one of the facts or assumptions Is false. (In the main, we are not concerned with fuzzy logic or with probabilistic reasoning. Thus, conclusions that are likely to be true, or quite possibly true, are not what we have in mind. We shall touch briefly on that topic in the concluding section.) The goal of automated reasoning is the design and implementation of a computer program that provides assistance for that phase of problem solving requiring reasoning. Reasoning is applied after the problem has been identified and formulated, after an approach has been chosen to attack the problem, and only if the chosen approach dictates the need for such reasoning. As we are using the term, some problems do not require reasoning to obtain a solution, for computation suffices.

An automated reasoning program can assist in so many different ways and for such diverse applications because of its generality. The most commonly used language—that of clauses [32]—for phrasing problems for an automated reasoning program is very general and powerful, although admittedly somewhat difficult to master. We were surprised to discover that we had underestimated its generality and power and that, with effort and inventiveness, the language of clauses could easily dispatch an even wider variety of tasks [28,35,24,36,25]. The various inference rules [18,19,17,13] employed to draw conclusions apply regardless of the domain from which the problem is selected. The strategies used to control the application of inference rules enable a reasoning program to avoid much fruitless inquiry [31] , and permit one to impose knowledge and intuition [16] on the program's search for a solution. The procedure of demodulation [33] allows a reasoning program to canonicalize and simplify information, resulting in an increased effectiveness. Finally, subsumption [18] is employed to remove duplicate information and, more importantly, to retain general facts and discard Instances of those facts. An automated reasoning program relies on an arsenal of weapons for problem solving, which is why it can be used for various purposes and in various studies.

Of course many problems remain to be solved in the three main areas, representation, inference rule, and strategy. Nevertheless, as will soon become evident, difficult questions have yielded to an attack relying on heavy use of an automated reasoning program. We shall now turn to some of those questions. The program used to answer the questions is resolution-based, but the other procedures cited above all play a vital role. What we clearly demonstrate is the usefulness of a resolution-based automated reasoning program both in research and for applications.

3. ANSWERING OPEN QUESTIONS WITH AN AUTOMATED REASONING PROGRAM

Answering open questions, regardless of the means, is considered a laudable activity in most circles. Answering any such question with an automated reasoning program might have been thought impossible. Nevertheless, such has occurred. Of course, this occurrence represents a breakthrough for automated reasoning, and hence for automated theorem proving. Roughly the first 18 years of automated theorem proving were spent in attempting to prove already proven theorems with an automated theorem-proving program, for knowing a proof permits the researcher to make judgments about the progress the reasoning program is making. Notwithstanding, one of the primary goals is that of using such a program to prove conjectured theorems—claims for which a proof is not yet known, or for which a proof may not even exist. To complement this activity, the need exists for disproving a conjectured theorem, hence finding a model or counterexample. The questions to be discussed address both aspects, that of finding a proof and that of generating a model or counterexample.

For the first question, concerned with the dependence or independence of axioms in a ternary Boolean algebra [23], the capacity of AURA (for Automated Reasoning Assistant) to generate models was relied upon. For the second question, that concerned with the possible existence of certain semigroups [26,27], the program's capacity to generate models as well as find proofs was employed. For the third question, that concerned with the adequacy or inadequacy of various formulas in equivalential calculus [35,24], AURA's capacity to find proofs came into play as well as its capacity to consider schemata under certain conditions. Since the details and methodology for solving the three questions can be found elsewhere [23,26,35,24,27], we are content merely to present each question with its solution.

Answering open questions—more precisely, the attempt to answer such questions—by using a reasoning program as an automated reasoning assistant often has a secondary benefit. The attempt may force one to formulate new methodology. As you will see, new methodology was needed. The existing features of the program were sufficient for the task, for the new methodology did not require implementation of new features tuned to the particular problem under attack.

The program AURA was indispensable for answering the open questions to be discussed. As evidence, we point out that the researchers who answered these questions have no expertise in any of the fields concerned. Their attempt to answer the first question was motivated solely by the attempt to demonstrate the value of using an automated reasoning program as an assistant. Their consideration of the second question was to show that somewhat difficult questions of interest to mathematicians could also be attacked with the assistance of a reasoning program. Finally, their interest in the third question was to show that the program was useful for a problem of a completely different type.

How hard are the open questions that were answered? While the first of the three questions could in no way be termed difficult, the second clearly presents serious problems. In particular, the number of semigroups of order 7 exceeds 800,000, and no smaller semigroup exists of the type to be discussed. The third question presents an even more awesome search space to be examined or circumvented. The number of formulas containing 27 significant symbols is greater than 100,000,000,000, and the completion of the study relied on a formula with 103 significant symbols. The solutions were obtained without examining the myriad of uninteresting semigroups or irrelevant formulas.

In view of the researchers' lack of expertise in any of the fields from which the diverse problems are taken, we have strong evidence that an automated reasoning program did play a vital role in answering the previously open questions.

3.1. Ternary Boolean Algebra

The first question concerns the possible independence of the elements of the usual set of five axioms given for a ternary Boolean algebra. Intuitively, the function f that occurs in each of the five axioms can be thought of as a 3-place product, and the function g as inverse.

```
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
```

The problem is to determine which, if any, of the five axioms is independent of the remaining four. If an axiom Is dependent, then a proof can be found of that fact. If an axiom Is independent, then a model exists that satisfies the other four but does not satisfy the independent axiom. Axioms 4 and 5 were known to be dependent even before proofs were obtained with two unrelated theorem-proving programs in 1969—that of Alan and Luckham [1] and that of G. Robinson and Wos (unpublished). Axioms 1, 2, and 3 are independent, and the required models were obtained with AURA.

Before this study of the fascinating-to-few field of ternary Boolean algebra, the methodology for generating models and counterexamples with an automated reasoning program did not exist. We credit S. Winker [63] for this contribution. He also is chiefly responsible for answering the questions under discussion, but his methodology is the more significant achievement.

3.2. Finite Semigroups

The second question concerns the possible existence of a finite semigroup in which one type of mapping is present while another type is absent. Specifically, does there exist a finite semigroup admitting a nontrivial antiautomorphism but admitting no nontrivial involutions? A nontrivial antiautomorphism is a one-to-one, onto mapping H such that H(xy) = H(y)H(x) and such that H is not the identity mapping. A nontrivial involution is a nontrivial antiautomorphism whose square is the identity mapping. Whichever way the question is settled, commutative semigroups are excluded from consideration, for they are not of interest by assumption.

This question is more interesting and much harder than the previously discussed question for ternary Boolean algebra. I. Kaplansky, the famous algebraist at the University of Chicago, suggested this question because of its relation to a similar question in division algebras. If the presence of a nontrivial antiautomorphism always implies the presence of a nontrivial involution, then the proof-finding capacity of an automated reasoning program comes into play. If a semigroup exists for which the implication does not hold, then the model-gene ration capacity is the key. The latter case presents certain problems since the number of noni8omorphic semigroups Increases rapidly with the order—more than 800,000 exist of order 7—and thus the search for the desired semigroup is potentially lengthy. Thus, to answer the question, we must either find a proof or generate a model. The model generation capacity settled the issue by finding a semigroup of the desired type. The first found has order 83 [26].

With this information in hand, the natural question to ask is what is the smallest such semigroup. Just as the previous study led to the methodology for generating models and counterexamples, this study led to new methodology. To find the minimal semigroup with the given properties [27] required AURA to cope with problems of isomorphism. A way was found to have AURA apply various other techniques such as the well-known without loss of generality argument that mathematicians use. The result was a proof that the smallest semigroup of interest has order 7 [27] and that four such nonisomorphic ones exist. This study was conducted jointly with Winker.

The discussion of this question is not complete without a brief reference to the true story [34] behind its solution—the story of what occurred before finding the semigroup of order 83. Before discovering that semigroup, we had a prior solution. That solution to the open problem was thought to be correct for some 18 months, before it was discovered that the wrong problem had been solved. What was missing was the correct definition of an involution. In the first attempt at solving the problem, an involution was not assumed to be an antiautomorphism, but simply a one-to-one, onto mapping J whose square is the identity and such that J(xy) - J(x)J(y). With that definition of J, there exist semigroups of order 4 that suffice, but the corresponding problem is not of much interest. When the correct problem was identified, three hours of the researchers' time sufficed to extend the methodology to enable AURA to solve the problem in a sequence of runs requiring two minutes of time on an IBM 3033. The method used to solve the uninteresting problem proved pertinent to the interesting one, so the effort was not wasted after all. We thus have an illustration of how a class of problems can be treated by a single methodology developed for one of them—even if the wrong one. This story is told when we are asked if one must start anew when submitting each distinct problem to an automated reasoning program.

3.3. Equivalential Calculus

The third question concerns the adequacy or inadequacy of various formulas to serve as shortest single axioms for the equivalential calculus. Intuitively, the equivalential calculus is concerned with the abstraction of "equivalent". The calculus consists of the formulas that can be composed from variables x, y, z, ..., and the two-place function E. In addition, the inference rule of condensed detachment [35] is employed to yield formulas from pairs of formulas. The calculus can be axiomatized with a single formula, a formula from which all theorems can be deduced by repeated application of condensed detachment. The shortest single axiom must contain five occurrences of the function E. Before this area of formal logic was studied with the assistance of AURA, 11 shortest single axioms were known. When the study commenced, seven of the 630 formulas in which the function E occurs exactly five times remained unclassified—the adequacy or inadequacy for being a single axiom was unknown.

The model-generation method was deemed inadequate for the study. What was desired was a characterization of the entire set of theorems deducible from each of the seven formulas under investigation. While answering the open question for ternary Boolean algebra led to the methodology for generating models, this study led to a methodology to achieve the goal of obtaining a complete characterization [35,24] of the set of deducible theorems. The approach centers on the use of schemata, forms that describe collections of formulas sharing certain global syntactic properties. The needed schemata were found by relying heavily on an automated reasoning program. The use of such a program, however, was not limited to schemata discovery. Other uses [36] include suggesting conjectures, finding notation, conducting case analyses, proving and refuting conjectures, and of course finding proofs. The completion of this study required many computer runs, much examination of the output, and approximately 30 minutes of CPU time on an IBM 3033.

With invaluable assistance from AURA, five of the seven previously unclassified formulas were proved too weak to be a shortest single axiom [35,24]. What was surprising was that two of the seven were proved to be new shortest single axioms [24], contrary to the conjecture that no more would be found. The two new slfortest single axioms are denoted by XHK and XHN. The study of equivalential calculus, especially the proof that two additional shortest single axioms exist, proved to be the most intricate and complex use of an automated reasoning program known to us. The proof that XHK is a single axiom includes 84 applications of condensed detachment and Involves formulas as long as 71 symbols, not counting grouping symbols and commas. The proof that XHN is a single axiom consists of 162 steps, one of which involves a formula of length 103. The number of formulas of length 27 exceeds 100,000,000,000.

4. OTHER USERS, OTHER PROGRAMS

The primary goal of automated reasoning, and hence of automated theorem proving, is the design and implementation of a computer program. The primary use of the program is to assist in the reasoning phase of problem solving. If that program can be used only by an expert in automated reasoning, then we have failed. More precisely, if we never provide an automated reasoning program that can easily be used by researchers from other fields and by individuals interested in applications outside of our field, then we have not fulfilled our charter. Some progress in this direction has occurred.

Both students and faculty from the Illinois Institute of Technology have used and are using an automated reasoning program for their research. Two of the students, W. Wojciechowski and W. Kabat, have received a Ph.D. for work on logic circuit design. Each relied heavily on an automated reasoning program to design circuits with given characteristics. Many of the circuits obtained by Wojciechowski [28] were pronounced by others In the field of design as superior (with respect to transistor count) to those previously known. The technology employed is that of T-gates, the analogue in A-valued logic of the multiplexor in binary logic, as the basic building block or component. The methodology was then extended by Kabat [6] to the use of arbitrary building blocks, for either multi-valued or binary logic design. In addition, A. Wojcik, their thesis advisor, is using a reasoning program for the validation of existing designs [29]. For example, he has obtained a 297-step proof validating the design of a 16-bit adder. The proof was obtained In 20 seconds of CPU time on an IBM 3033. The research in these areas relies on the use of AURA—Automated Reasoning Assistant.

Also at IIT, M. Evens and her student T. Wang are studying the possible use of an automated reasoning program for chemical synthesis [5]. The problem focuses on a large database of simple compounds and a set of reaction rules that combine them to produce complex compounds. At the University of Illinois at Urbana, L. Hanes is using an automated reasoning program to prove the equivalence of symbolic representations purported to represent the same logic circuit. The two studies are being conducted with the assistance of a reasoning program ITP [12] produced from the LMA (Logic Machine Architecture) system. The LMA system is a set of procedures or subroutines for tailoring an automated reasoning program to given specifications. The tailoring is not an automated process. The LMA system [8,9] was designed and implemented by E. Lusk, W. McCune, and R. Overbeek.

Occasionally, various diverse areas of research meet in a fortuitous fashion. The study of the equivalential calculus and the development of LMA provide an example of such an occurrence. In the fall of 1982, J. Kalman from the University of Auckland came to Argonne to study and use our techniques in automated reasoning. He is interested in equivalential calculus and is the person who pointed us to the previously discussed open questions in that field. During his visit, we asked him about the possibility of axlomatizing the calculus with three obvious axioms that might be chosen in view of the fact that the calculus studies the abstraction of the notion of equivalent. Specifically, do the following three formulas axiomatize the calculus?

```
1) E(x,x)
2) E(E(x,y),E(y,x))
3) E(E(x,y),E(E(y,z),E(x,z)))
```

These three formulas can be thought of as reflexivity, symmetry, and transitivity. Kalman answered our question in the affirmative [7] by using a program produced from LMA. We thus have another example of actual use of an automated reasoning program.

One other group of users must be mentioned. A workshop was given at Argonne National Laboratory in May of 1982. The workshop was a tutorial in automated reasoning, demonstrating existing and possible uses of both AURA and programs produced from LMA. Most of the attendees were from various industries, including IBM, DEC, Motorola, INTEL, AMOCO, Silicon Systems, APTEC, NCR, and Lockheed. They came expressing some doubt about the potential usefulness of automated reasoning, but hoping to gain an understanding of its basic aspects. They left expressing much enthusiasm and some optimism, and with an understanding of the basics. Perhaps more important, some of them are now experimenting with an automated reasoning program, produced from LMA, for various applications. Although no industry has yet committed itself to a solid attempt to apply automated reasoning, the change in attitude from but a few years ago is marked.

We now come, as promised in the introduction, to the briefest of accounts of the work of others in the field of automated reasoning. The successes of other researchers provides still more evidence of the progress that has occurred. These successes resulted directly from the use of an automated reasoning program. Obviously other powerful automated reasoning or automated theorem-proving programs exist, besides those designed and implemented by the ANL/NIU group. Independent development of this kind is essential for the field. The availability of other programs provides valuable opportunities to compare performance, features, implementation details, and special-purpose versus general-purpose approaches to solving a problem.

Perhaps the most outstanding of these programs, measured by the problems solved with it, is that of Boyer and Moore. Their system is a special-purpose program aimed at program verification. Among their recent successes with their program they proved the recursive unsolvability of the halting problem [3] and also proved the invertibility of the RSA encryption algorithm [4] used for message transmission. The encryption algorithm is not a classroom exercise, nor a problem purely of theoretical interest, for the algorithm is in actual use. Boyer and Moore are to be congratulated for their achievements. Besides being a beautiful piece of work, the success demonstrates the potential value of using such a program. As we do, they also use their program in the assistant mode, occasionally suggesting to it lemmas that might be of value when and if the program succeeds in proving them.

Two other efforts must be mentioned, that of Bledsoe and that of Siekmann. Bledsoe's work [2] is oriented toward intermediate analysis, and he has proved with his system some fairly difficult and fundamental theorems in that domain. The proofs were obtained with an automated theorem-proving program employing a number of his techniques. Especially from the viewpoint of automated reasoning, the area of intermediate analysis presents significant challenges. Rather different from Bledsoe's approach, Siekraann has developed a large integrated reasoning program. As one example of the type of problem that can be solved with his program, Siekmann has recently announced [20] that a proof of an interesting theorem from relevance logic has been obtained with his system.

5. FROM STAGE 1 TO STAGE 2; WHAT CAUSED THE PROGRESS?

Fields like automated reasoning usually begin in stage I, in which conjectures are made about potential value, possible uses, and expected power. If all goes well, such a field advances to stage 2, in which some of the potential has been realized, various uses exist, and successes have been achieved that might not have occurred were it not for the power that has been provided. Of course the goal is to reach stage 3, in which the field is widely accepted and in which its products are widely used. Automated reasoning is at stage 2—which is what Sections 3 and 4 are about. Of course, stage 2 is not a point on the entire spectrum, but rather an interval in which we find ourselves with many problems still to solve. The question here is, How did automated reasoning advance from stage 1 to stage 2?

One of the prime factors was much experimentation with an existing automated reasoning or automated theorem-proving program. Various concepts that are considered central to the field came directly from such experimentation, concepts such as the unit preference strategy [30] , the set of support strategy [31], demodulation [33], and paramodulation [17]. Various uses were added because of experimentation, uses such as question-answering, program verification, database inquiry, and the design and validation of logic circuits. For experimentation, the ideal case is that in which the program presents to the researcher an array of choices for representation, inference rule, strategy, and the use of auxiliary processes. Since the only available form offered by AURA for representing information was that of clauses, we were not in the ideal case. In the other areas, however, we did have access to a wide array of choices. A large and integrated system such as AURA provides unusual opportunities for learning and, in fact, no substitute can be found.

A second important factor is the work of Overbeek [15,10,16,13,14], especially his design and implementation of various programs. Of these programs, AURA is by far the most powerful and versatile automated reasoning program, and is the one that was used to answer the open questions. AURA extends and incorporates the design and implementation found in Overbeek's earlier work, and Includes vital features due to Smith, Winker, Lusk, and Wos. Without AURA, the open questions presented here would not even have been considered, at least by the researchers who solved them. The questions even might have resisted solution by any means were it not for the existence of this program with its many features. The combination of the particular implementation, well-chosen inference rules, and carefully selected strategies was mainly responsible for the successes.

How does the particular implementation have such an important effect? A poor implementation can obscure the value of a new idea. For example, if a new and powerful strategy is tested with a poor implementation, any attempt at obtaining a proof might take essentially forever. On the other hand, a poor implementation can equally mislead the researcher into evaluating a bad idea as good. If, for example, a new inference rule is tested with a program that requires a great deal of time to obtain a proof for even the simplest of problems, a sharp reduction of that time might be meaningless. The researcher could, however, mistakenly conclude that the new inference rule was quite promising. A good implementation, when presented with the new rule, might have produced no important change in the experiments, and thus the erroneous conclusion would have been avoided. The choice of implementation, therefore, can have serious consequences.

A third important factor that advanced automated reasoning from stage 1 to stage 2 is the work of Winker, who, by the way, does not totally agree with the views expressed in this paper. His methodology for generating models and counterexamples provides the needed complement to finding proofs. But perhaps more important is his establishment of the usefulness of clause representation. His mastery of problem representation and the techniques resulting from it have led to the successful submission of diverse and unrelated problems to an automated reasoning program.

Before turning to the final factor that caused the advance to stage 2, we note that many researchers have made contributions both to the theory and to the application of automated reasoning. We are content here to concentrate on the work that has led to our contribution, and not focus on the many and important aspects of others' work.

The final factor promoting the advance of automated reasoning centers on answering open questions with the assistance of an automated reasoning program. This success has caught the interest of members of other fields, such as mathematics and formal logic. We have clearly demonstrated that an automated reasoning program can provide invaluable assistance in research. We must recognize that, for example, proving a known theorem with a reasoning program often does not impress those outside of the field. For one thing, real difficulty exists in ascertaining how unbiased the experiment is. An analysis of how much the "dice were loaded" is most complex. In contrast, answering an open question by relying on an automated reasoning program produces many fewer questions about "loading the dice". One may certainly wonder how much credit should be given to the ingenuity of the researchers and to their intimate knowledge of the program that was used. However, when one takes into account the fact that the researchers knew and still know very little about the fields from which the open questions were selected, the case for the program's contribution is strengthened markedly. An even stronger case can be made when one takes into account the successes of other researchers, especially when those researchers have virtually no knowledge of the underlying reasoning program. When the success with open questions was followed by the success of members from other fields using an automated reasoning program for other applications, the field had arrived at stage 2.

6. FROM STAGE 2 TO STAGE 3: HOW DO WE GET THERE

Much experimentation with existing automated reasoning programs will be required to get from stage 2 to stage 3. Certainly more theory is needed in the three major areas of representation, inference rule, and strategy. However, to the disappointment of some, too much emphasis is placed on theory without evidence to support the resulting claims. When a particular approach has been explored for many years and still cannot be employed even to solve problems whose solution is known, that approach must be viewed with great skepticism. Perhaps some of the effort devoted to the theoretical questions might better be placed in bridging the gap between the researchers in automated reasoning and those who might use an automated reasoning program.

In particular, we must find a way to inform various individuals of how a reasoning program works and how to "think" the way it does, and make clear that, with patience and pain, communication with such a program is possible and available on various levels. For example, one argument for using such a program is the fact that it can, upon request, present the precise derivation information permitting the user to ascertain what may be missing from the input, or why the program is struggling. If the user doubts the results obtained with an automated reasoning program, the user can examine each step of the reasoning process. Acquainting those who might use an automated reasoning program that such advantages exist and that various successes have occurred will help. In addition, we must seriously consider certain current practices.

Among the most dangerous is that of writing about advances in automated reasoning but with no evidence. Of course a paper can be of substantial value even though no experimental data can be quoted. However, actual problems that are solved are, In the final analysis, the only real sign of progress. The problems need not be open questions. Proving a well-known theorem, even if trivial, counts. Verifying a small computer program counts. Solving a puzzle counts. What must be viewed with grave question is the paper that has the appearance of a mathematics paper, but whose only examples are either propositional or purely syntactic.

With few exceptions, propositional examples or purely syntactic examples taken from first-order predicate calculus are valuable only for expository purposes. An illustration, for example, that consists of a set of clauses containing variables, functions, and predicates, but for which no meaningful interpretation can be given, must be viewed with skepticism when it is offered as evidence of the value of a new concept. The usefulness of an inference rule or strategy cannot be demonstrated by such an illustration. In fact, the conjectured value is highly suspect, for such a conjecture should at least appeal to intuition. Such an appeal in most cases should in turn be based on some real example, some problem selected from the existing literature, or some problem that has meaning to individuals outside of automated reasoning. When the value of a new idea can be indicated only by a syntactic example, or when the weakness of an Idea can be illustrated only by a syntactic example, the corresponding claim is suspect. We therefore recommend very strongly that papers include examples that correspond to real problems. Since the literature suffers from a paucity of such problems, you can turn to puzzles for that domain provides a wealth of examples.

Some puzzles correspond to problems that might be faced by some industry, at least in miniature. As one example, consider the problem In which you are asked to design a circuit and are required to use only NAND gates. The circuit has two inputs, x and y, and two outputs, y and x. The input x is above the input y, but the reverse is true for the outputs. Finally, you are required to build this circuit with no crossing wires. This puzzle has been solved with the program ITP produced from LMA.

For a more complex example, you are asked to build a circuit with inputs a, b, and c, and with outputs not a, not b, and not c. You can think of the inputs as each taking on values 0 or 1. You are permitted to use as many OR gates and as many AND gates as you wish, but no more than two NOT gates. (Incidentally, this problem was brought to our attention by E. Snow of INTEL, an attendee at the 1982 Workshop on Automated Reasoning, which illustrates the value of such tutorial/dialogues.) The problem can be phrased equivalently as a problem of moving the contents of three locations to three other locations in memory, using "move", "or", "and", and "complementation" instructions. You are of course limited to the use of no more than two "complementation" instructions. This puzzle has also been solved with the program ITP, but substantial time was required as compared with the previous circuit design puzzle.

By including actual problems as part of the illustrations of the potential value of a new inference rule or a new strategy, the writer of the paper presents information by which the new contribution can perhaps be judged. In addition, if the problem has not been presented before, the writer has provided a needed addition to the pool of problems. The preferable case, of course, is that where actual results from a computer run can be quoted. When those results reflect that another previously open question was solved with the assistance of an automated reasoning program, they take on added significance. Perhaps the most significant of all is the case in which an automated reasoning program is successfully used in some industrial or commercial application.

When the suggestion was made that experiments using an actual automated reasoning program were needed, a typical response was that no program was easily available and writing one required too much time. That void has now been filled. In fact, a portable automated reasoning program ITP can now be obtained. Moreover, a system LMA (for Logic Machine Architecture) consisting of a set of procedures or subroutines for producing a reasoning program satisfying given specifications is also available. Both ITP [12] and LMA [11] are due to Lusk, McCune, and Overbeek. The system LMA has been ported to a number of machines, including rather inexpensive ones. It offers access to a large integrated reasoning program, thus permitting the user the simultaneous use of various inference rules and strategies that in turn permit comparisons to be made. Equally, the structure permits the removal of unwanted procedures. The program is written in PASCAL, and therefore the AI community might wonder how accessible it is for integration with existing software. LMA is designed to interface to LISP and to programs like MACSYMA, and is even prepared for multiprocessing. The LMA system is structured to permit a researcher in automated reasoning to adopt a language other than clause input, to add to the array of inference rules, and to make other similar modifications. Since a number of the central concepts of the field have resulted from experimentation, we recommend that, if no alternative exists, you obtain a copy of LMA and experiment freely. The lack of a suitable reasoning program on which to test ideas Is no longer an obstacle.

The foregoing remarks do not mean that we endorse an ad hoc approach. Rigor and care in both the development of the theoretical foundations and the design and implementation are mandatory. In almost all instances, for example, soundness is required. Experimentation that includes a careful comparison with the work of others is most laudable. What the foregoing does say is that those who considered automated reasoning, and hence automated theorem proving, at best an ivory tower discipline were and are in error. Practical applications may occur sooner than anyone might have guessed, if the progress of the last few years is an indication.

As mentioned earlier, we must convey to various potential users what automated reasoning is about. The fact that a reasoning program can cope with problems that require moving cannibals, playing dominos on a modified checkerboard, or weighing billiard balls comes as a surprise to many. The natural reaction to such a claim is skepticism, even sometimes by members of the field itself. What is necessary is the usual: namely, to educate people about what can be done. Were an Interface available permitting a user to communicate to an automated reasoning program in the user's own language, the gap and resistance would narrow rapidly. The design and implementation of such a translator is truly a worthy project. Although this useful interface is still missing, the present state of automated reasoning, as a whole is most encouraging.

7. THE FUTPRE

The future is far more promising than many might imagine. In fact, exciting opportunities now exist, both for research and for applications. Many basic research questions in automated reasoning remain unanswered, questions that are challenging in themselves but also have substantial relevance to solving problems in other areas. For example, can the power of the set of support strategy that is most evident at level 1 be recursively extended to higher levels in the clause space? What strategies can be formulated to tightly control pararaodulation or comparable inference rules for building in equality? A number of applications appear quite tractable. They include the design and validation of circuits and chips, the chemical synthesis of complex compounds, robotics, program verification, and database Inquiry.

Both those already active in automated reasoning and those who may wish to use an automated reasoning program should experience great excitement. The probability that such programs will be commonplace in but five years has increased from essentially zero to one half. Among the factors that have caused this change are the progress that has occurred in just the last four years and the trend in the cost and in the type of computer that is available. Within two years, a computer with one megabyte of memory may be available for less than 2,000 dollars. The user can then port an LMA-based system, or an alternative, to such an inexpensive machine. Experimentation, formulation, development, and implementation will be accessible to many and diverse Individuals. Comparisons of representations, of inference rules, and of strategies will be possible, and on real and varied problems. The evidence gathered in the past few years, demonstrating the value and usefulness of a resolution-based reasoning program with its attendant features, may cause many computer scientists and others to share the fascination of automated reasoning. But even more is to come.

8. A FORECAST OF WHAT IS TO COME

At home, you have a set of discs; and at the office, you have a copy of each. On your home computer resides a reasoning program; on the office computer, a duplicate resides. By selecting the appropriate disc, you can have a dialogue with the reasoning program on a variety of topics. The topics Include scheduling the day's activities so that conflicts are avoided, analyzing the consequences of a proposed decision, and prioritizing tasks to be accomplished. You carry a few of these discs with you so that they can be used with your reasoning program on your pocket computer, should some important problem arise.

Your personal reasoning program can learn from previous experiments. When asked to solve a problem, it can determine that the given strategy must be replaced by another. Given the problem, your reasoning program can choose a representation that has the best chance of permitting it to solve the problem. It can decide that more unit clauses are needed, or that more equality clauses would help, and then set about to find them. Finally, your reasoning program can examine its output, presenting to you only the more interesting results. In fact, it can find significant theorems, make conjectures, and suggest concepts that might be worth studying.

You have become aware of four new areas of research with commercial possibilities. Their respective subjects of investigation are: systems of axioms or assumptions that permit a reasoning program to attack problems in new areas; inference rules that are tailored to specific problem areas; strategies that enable a reasoning program to be very intelligent in its attack on a given problem; and, finally, reasoning programs themselves. Should you spend any of your research time on any of these areas? Your personal automated reasoning assistant may have the answer. Such a scenario may sound like science fiction, but it may well be scientific fact in the near future.

9. CONCLUSIONS

The field of automated reasoning, and hence automated theorem proving, has progressed rapidly in the previous four years. Open questions in mathematics and in formal logic have been answered with the assistance of an automated reasoning program. Logic circuits have been designed and validated, and by researchers with little or no background in automated reasoning. Complex algorithms have been verified using a theorem-proving program. Representatives from various industries are expressing both interest and curiosity about the potential of the field. One advantage of using a reasoning program is the information contained in its output. All derivations can be given explicitly and completely, and can be checked algorithmically. Examination of results from early runs often shows what is missing from the input or shows why the program is struggling. The results can be trusted, for they are obtained with inference rules that are sound, that yield conclusions that follow inevitably and logically from the assumptions. Even when probabilistic reasoning is desired, as in certain expert systems, a means exists for having an automated reasoning program obtain the results.

Admittedly, using an automated reasoning program is often difficult, but the rigor and unambiguity often more than compensate for the required effort. An interface permitting the user to communicate in the user 's chosen language would be of obvious and great value for it would sharply reduce the difficulty of using a reasoning program. What we have shown is that, even without this interface, a resolution-based, automated reasoning program such as AURA provides invaluable assistance in problem solving. The power of such a program is derived not only from its array of Inference rules, but also from procedures such as demodulation, subsumption, and the strategies that it employs.

What is needed now is much experimentation. Experimentation has led to the formulation of a number of important concepts of the field. Where previously such experimentation was made difficult by the lack of a suitable program, now such a program is available. Moreover, a system LMA is available and portable, and provides the researcher with the opportunity of tailoring an automated reasoning program to given specifications. Only by using a reasoning program to attack problems both in research and in application can we gain the needed information to solve certain basic problems of the field. Such usage has resulted in the advance of automated reasoning from conjectured usefulness to actual usefulness, and will result in advancing the field to the stage of wide acceptance and use.

Much is still unknown about representation. More powerful inference rules are needed. More effective strategies must be found, for without strategy even the simplest of problems becomes hard. Nevertheless, with all that remains to be accomplished, automated reasoning has made important advances, resulting in more powerful and versatile reasoning programs. Perhaps soon such a program will be used by many In both research and for applications, and will function as an automated reasoning assistant.

REFERENCES

[1] Allen, J. and Luckham, D., "An Interactive theorem-proving program," Machine Intelligence, Vol. 5(1970), Meltzer and Michie (eds), American Elsevier, New York, 321-336.

[2] Bledsoe, W. W. "Some Automatic Proofs in Analysis", December 1982, University of Texas, Math Dept Memo ATP-71.

[3] Boyer, R. and Moore, J, "A Mechanical Proof of the Unsolvability of the Halting Problem," accepted for publication by J. ACM.

[4] Boyer, R. and Moore, J, "Proof Checking the RSA Public Key Encryption Algorithm," technical report 33, The Institute for Computing Science, University of Texas at Austin 78712.

[5] Evens, M. Private Communication (1982).

[6] Rabat, W. and Wojcik, A., "Automated synthesis of combinational logic using theorem proving techniques," Proceedings of the Twelfth International Symposium on Multiple-Valued Logic, Paris, France, May 1982, IEEE, 178-199.

[7] Kalman, J., Private Communication (1981).

[8] Lusk, E., McCune, W. and Overbeek, R., "Logic machine architecture: kernel functions," 6th Conference on Automated Deduction, Vol. 138, Lecture Notes in Computer Science, ed. D. W. Loveland, Springer-Verlag, Berlin, Heidelberg, New York, 1982, 70-84.

[9] Lusk, E., McCune, W. and Overbeek, R., "Logic machine architecture: inference mechanisms," 6th Conference on Automated Deduction, Vol. 138, Lecture Notes in Computer Science, ed. D. W. Loveland, Springer-Verlag, Berlin, Heidelberg, New York, 1982, 85-108.

[10] Lusk, E. and Overbeek, R., "Experiments with resolution-based theorem-proving algorithms," Computers and Mathematics with Applications 8 (1982), 141-152.

fill Lusk, E. and Overbeek, R., "Logic Machine Architecture Inference Mechanisms - Layer 2 User Reference Manual", Argonne National Laboratory Technical Report ANL-82-84.

[12] Lusk, E. and Overbeek, R., "An LMA-Based Theorem Prover," Argonne National Laboratory Technical Report ANL-82-75.

[13] McCharen, J., Overbeek, R. and Wos, L., "Problems and experiments for and with automated theorem proving programs," IEEE Transactions on Computers, Vol. C-25(1976), 773-782.

[14] Overbeek, R., "A New Class of Automated Theorem-proving Algorithms", J. ACM, Vol. 21(1974), 191-200.

[15] Overbeek, R., "An implementation of hyper-resolution," Computers and Mathematics with Applications, Vol. 1(1975), 201-214.

[16] Overbeek, R., McCharen, J. and Wos, L., "Complexity and related enhancements for automated theorem-proving programs," Computers and Mathematics with Applications, Vol. 2(1976), 1-16.

[17] Robinson, G. and Wos, L., "Paramodulation and theorem-proving in first-order theories with equality," Machine Intelligence, Vol. 4(1969), Meltzer and Michie (eds), American Elsevier, New York, 135-150.

[18] Robinson, J., "A machine-oriented logic based on the resolution principle," J. ACM, Vol. 12(1965), 23-41.

[19] Robinson, J., "Automatic deduction with hyper-resolution," International Journal of Computer Mathematics, Vol. 1(1965), 227-234.

[20] Siekmann, J., Private Communication (1982).

[21] Smith, B., "Reference manual for the environmental theorem prover, An Incarnation of AURA," to be published as an Argonne National Laboratory technical report.

[22] Winker, S., "Generation and verification of finite models and counterexamples using an automated theorem prover answering two open questions," J. ACM, Vol. 29 (1982), 273-284.

[23] Winker, S. and Wos, L., "Automated generation of models and counterexamples and its application to open questions in ternary Boolean algebra," Proc. of the Eighth International Symposium on Multiple-valued Logic, Rosemont, Illinois, 1978, IEEE and ACM Publ., 251-256.

[24] Winker, S. and Wos, L., "New shortest single axioms for the equivalential calculus," in preparation.

[25] Winker, S. and Wos, L., "Procedure implementation through demodulation and related tricks," 6th Conference on Automated Deduction, Vol. 138, Lecture Notes in Computer Science, ed. D. W. Loveland, Springer-Verlag, Berlin, Heidelberg, New York, 1982, 109-131.

[26] Winker, S., Wos, L. and Lusk, E., "Semigroups, antiautomorphisms, and involutions: a computer solution to an open problem, I," Mathematics of Computation, Vol. 37 (1981), 533-545.

[27] Winker, S., Wos, L. and Lusk, E., "Semigroups, antiautomorphisms, and involutions: a computer solution to an open problem of Minimality,-II," in preparation.

[28] Wojciechowski, W. and Wojcik, A., "Automated design of multiple-valued logic circuits by automatic theorem proving techniques," to appear in IEEE Transactions on Computers.

[29] Wojcik, A., "Formal design verification of digital systems, Proceedings of the 20th Design Automation Conference, June 1983.

[30] Wos, L., Carson, D. and Robinson, G., "The unit preference strategy in theorem proving," Proc. AFIPS 1964 Fall Joint Computer Conference, Vol. 26, Part II, 615-621 (Spartan Books, Washington, D.C.).

[31] Wos, L., Carson, D. and Robinson, G., "Efficiency and completeness of the set of support strategy in theorem proving," J. ACM, Vol. 12(1965), 536-541.

[32] Wos, L., Overbeek, R., Lusk, E., and Boyle, J., "Automated Reasoning - Introduction and Applications," Book in preparation.

[33] Wos, L., Robinson, G., Carson, D. and Shalla, L., "The concept of demodulation in theorem provvng," J. ACM, Vol. 14(1967), 698-709.

[34] Wos, L. and Winker, S., "Open Questions Solved with the Assistance of AURA," to be published.

[35] Wos, L., Winker, S., Veroff, R., Smith, B. and Henschen, L., "Questions concerning possible shortest single axioms for the equivalential calculus: an application of automated theorem proving to infinite domains," accepted for publication by Notre Dame Journal of Formal Logic.

[36] Wos, L., Winker, S., Veroff, R., Smith, B. and Henschen, L., "A new use of an automated reasoning assistant: open questions in equivalential calculus and the study of infinite domains," submitted for publication.