ASSOCIATION FOR AUTOMATED REASONING
NEWSLETTER

No. 23 June 1993

From the AAR President, Larry Wos...

This issue of the *AAR Newsletter* features—in addition to the usual announcements and calls for papers—four articles of particular note.

The first is from our AAR secretary, Bill McCune, requesting that AAR members help us save mailing costs by having the newsletter sent via e-mail. I most strongly encourage you to accept Bill's invitation to join the electronic age!

The second article of interest is by a young scientist from China, Jian Zhang, who presents a short discussion of the satisfiability problem.

The third contribution is from another Chinese reader, Li Dafa. He discusses what is believed to be the first automated solution of the halting problem obtained without manual conversion of the original formulation (as was reported in an earlier *AAR Newsletter*).

Finally, I have included an article I wrote a few years ago—one that, if the truth must be told, the *New York Times* (in effect) encouraged me to submit but then never published. I think AAR members will find the article thought-provoking.

Electronic AAR Newsletters

Bill McCune, mccune@mcs.anl.gov

To get information to members more quickly and to save some costs, we plan to give members the option of receiving the *AAR Newsletter* by e-mail instead of on paper. Also, starting with this issue, we will make AAR newsletters available by anonymous FTP on info.mcs.anl.gov in pub/automated.reasoning.

We assemble the newsletter with basic LaTeX, without any special style files or auxiliary PostScript files, and we plan to make the LaTeX source file and the corresponding PostScript file available. (We don't plan to prepare plain ASCII versions.)

If you are interested, send a note with subject line "AAR news by e-mail" to mccune@mcs.anl.gov, indicating one of the following:

(1) send LaTeX source for newsletters by email;
(2) send postscript for newsletters by email; or
(3) notify by e-mail when newsletters are available by FTP.

If you choose any of the above options, you will not automatically receive the newsletter by ordinary mail. Of course, we'll by happy to send a copy of a particular issue by ordinary mail if you request it.

(We will be especially grateful if members outside of the U.S.A. use the e-mail service, because airmail postage is quite expensive.)

Finding Finite Models of Equational Theories

Jian Zhang
Academia Sinica, Beijing, P.R. China

The satisfiability problem (and the construction of models) is very important both theoretically and practically. It also has a close relationship with automated deduction. In fact, semantic deduction methods often require the automatic generation of finite models [1]. On the other hand, Winker and Wos [2] have demonstrated the use of theorem provers in finding models and counterexamples.

We have written a program called Mod/E that can find small models of some given equational theory [3]. One may also specify additional properties of the model in a first-order logical language. The functionality of Mod/E thus is as follows:

> Given a set E of axioms, a positive integer n, and possibly some property P, find an n-element model of E such that P holds.

Our program views the problem of finding finite models as constraint satisfaction in finite domains, that is, in domains $\{0, 1, ..., n-1\}$. The algorithm used is essentially backtracking plus equational reasoning. To avoid the generation of redundant equations, the program first instantiates the input axioms before searching for a model. Mod/E was implemented in C on a Sun workstation. We give two examples below.

Example 1: Group theory. Find a 4-element group.

Input

```
# Functions (with arities)
    { i(1) }  { f(2) }
# Axioms
    [ f(0,x) = x ]
    [ f(x,0) = x ]
    [ f(g(x),s) = 0 ]
    [ f(x,g(x)) = 0 ]
    [ f(f(x,y),z) = f(x,f(y,z)) ]
# Size of the Model
    (4)
```

Output

```
i:           f:
  0 1 2 3      0 1 2 3
               1 0 3 2
               2 3 0 1
               3 2 1 0
```

Example 2: Combinatory logic [4]. Find a model satisfying L and Q such that the strong fixed point property does not hold. (This property says that there exists some y such that for any x, we have $a(y,x) = a(x, a(y,x))$.)

Input

```
# Functions (with arities)
    { a(2) }
# Axioms ( 0 as Q, 1 as L )
    [ a(a(a(0,x),y),z) = a(y,a(x,z)) ]
    [ a(a(1,x),y) = a(x,a(y,y)) ]
# Size of the Model
    (4)
# Properties
    < Ay.Ex.(a(y,x) != a(x,a(y,x))) >
```

Output

```
a:
  0 0 2 2
  0 0 2 2
  1 1 3 3
  1 1 3 3
```

References

1. McCune, W., "Experiments with semantic paramodulation," *J. Automated Reasoning* 1 (1985) 231–261.

2. Winker, S., and Wos, L., "Automated Generation of Models and Counterexamples and Its Application to Open Questions in Ternary Boolean Algebra," in *Proceedings of the Eighth International Symposium on Multiple-Valued Logic,* IEEE and ACM, pp. 251–256 (May 1978).

3. Zhang, J., "Search for models of equational theories," *Proc. 3rd International Conf. for Young Computer Scientists* (ICYCS'93), Beijing, 1993.

4. Zhang, J., "Solution to an open question in combinatory logic," *AAR Newsletter* No. 21, September 1992, p. 12.

Editor's Note: John Slaney's program FINDER has similar functionality, is available by FTP, and is very fast. Contact jks@arp.anu.edu.au.

A Mechanical Proof of the Halting Problem in Natural Deduction Style

Li Dafa
Tsinghua University, Beijing, P.R. China

Introduction

We have implemented in Common Lisp an Automated Natural Deduction Proof (ANDP) system that can construct proofs of the first-order logic formulas with quantifiers in a natural deduction format. Natural deduction is taught in numerous logic texts (see, for example, [1, 5]); similarly, the rules of inference and the structures of the formulas used by ANDP can be found in books on logic.

The following table shows how to type formulat and display them on the screen.

Logic Formulas	Formulas Types	Output on the Screen
$A \vee B$	A or B	$A \vee B$
$A \wedge B$	A & B	A & B
$A \rightarrow B$	$A- > B$	$A- > B$
$A \leftrightarrow B$	A iff B	$A < - > B$
$\sim A$	$\sim A$	$\sim A$
$\forall x$	(Ax)	(Ax)
$\exists x$	(Ex)	(Ex)

What Is Natural Deduction Like?

A natural deduction system consists of axioms and rules of inference. A proof of a wff A is a sequence of wffs such that A is the last wff in the sequence and each wff in the sequence is an axiom or derived from previous wffs in the sequence by one of the rules of inference.

In our system a proof is a sequence of lines, each of which is of the following form [4,6]:

$$k.\{H1, ..., Hn\}| - A \qquad\qquad \text{Rule-name } n1, n2, ..., ni,$$

where k is a number that designates the formula as well as the line of derivation in which it occurs. $H1, ..., Hn$ show the hypotheses on which the formula A in the line depends. On the right, *Rule-name* represents the rule of inference, followed by a sequence of line numbers showing that A is an assertion obtained by applying the rule to line $n1, ..., ni$, or introduced by using the rule P (or HYP) [1,5].

Natural Deduction and Automated Theorem Proving

Many people have developed resolution-based proof systems, but the first-order logic formulas must be translated to clause form. In particular, the distributive law is often used to obtain clause formulas. For example, given $P \vee (Q \wedge R)$, its clauses are $(P \vee Q)$ and $(P \wedge R)$. The literal P occurs once in the former, twice in the latter. There are more occurrences of literals in the clause form and more clauses of a given formula.

Consider, for example, the problem known as the Andrews Challenge [6]:

$$[(\exists x)(\forall y)[Px \leftrightarrow Py] \leftrightarrow [(\exists x)Qx \leftrightarrow (\forall y)Qy]] \leftrightarrow [(\exists x)(\forall y)[Qx \leftrightarrow Qy] \leftrightarrow [(\exists x)Px \leftrightarrow (\forall y)Py]].$$

There are 67 symbols in the challenge, and it produces 1600 clauses. The problem is logically simple, but its size makes it too difficult to prove using resolution. In order to find a resolution refutation of a logic formula, it must be transformed to a prenex normal form. Then the existential quantifierehs in the prefix are eliminated by using Skolem functions. The resulting matrix is transformed to conjunctive normal form. Unfortunately, it is obvious that the original logic structure is destroyed, and resolution refutation is not readable. Some people therefore wish to transform their proofs in one style into natural deduction. Recently, many people have focused on the Gentzen system; the natural deduction system here was adapted from that system.

Our natural deduction system ANDF enables valid wffs to be deduced in a very “natural” way. It keeps the original structures of logic formulas, and it need not transform a formula into other normal form. It can give fairly natural and well-structured proofs, and can express theforms of arguments that arise in mathematical practice. Therefore, the proofs constructed in natural deduction style are reasonably readable and comprehensible. In Robinson’s unification algorithm only the substitution rule is used in the algorithm. Therefore, it cannot be used to handle formulas with quantifiers.

Our unification algorithm also can treat any wffs with quantifiers. Not only the substition rule but also the rules for quantifiers UG, US, EG, and ES are used in our unification algorithm; that is, given two effs, the algorithm tries to apply the rules for quantifiers to them so that they become equal. The automatic natural deduction proving system based on our algorithm has been implemented.

Halting Problem

The halting problem is an important theorem in computer science. It was presented by L. Burkholder in [3] as the 76th automated theorem proving problem. We note that there is a mistake in premiss 2 of the formulation in [2]: after we tried to prove it several times, we found that the two free occurrences of x should be the bounded occurrence of x. We replace $H2xy$ with Pxy, and $H3xyz$ with $Qxyz$, respectively. (Ex), (Ax) stand for existential quantifier and universal quantifier, respectively. The English statement for the halting problem is omitted; see [2]. The following formulas come from [2]. The notation is as follows:

Ax – x is an algorithm
Cx – x is a computer program in some programming language
$Dxyz$ – x is able to decide whether y halts given input z
Pxy ($H2xy$ is used in [2]) – x halts on given input y
$Qxyz$ ($H3xyz$ is used in [2]) – x halts on given input $< y, z >$
Oxy – x outputs y

Four premisses are given:

$P1$ (for premiss 1):
$(Ex)[Ax \wedge (Ay)[Cy \rightarrow (Az)Dxyz]] \rightarrow (Ew)[Cw \wedge (Ay)[Cy \rightarrow (Ax)Dwyz]]$
$P2$ (for premiss 2):
$(Aw)[[Cw \wedge (Au)[Cu \rightarrow (Av)Dwuv]] \rightarrow (Ay)(Az)[[[Cy \wedge Pyz] \rightarrow [Qwyz \wedge Owg]] \wedge$
$[[Cy \wedge \sim Pyz] \rightarrow [Qwyz \wedge Owb]]]]$
$P3$ (for premiss 3):
$(Ew)[Cw \wedge (Ay)[[[Cy \wedge Pyy] \rightarrow [Qwyy \wedge Owg]] \wedge [[Cy \wedge \sim Pyy] \rightarrow [Qwyy \wedge Owb]]]] \rightarrow$
$(Ev)[Cv \wedge (Ay)[[[Cy \wedge Pyy] \rightarrow [Pvy \wedge Ovg]] \wedge [[Cy \wedge \sim Pyy] \rightarrow [Pvy \wedge Ovb]]]]$
$P4$ (for premiss 4):
$(Ev)[Cv \wedge (Ay)[[[Cy \wedge Pyy] \rightarrow [Pvy \wedge Ovg]] \wedge [[Cy \sim Pyy] \rightarrow [Pvy \wedge Ovb]]]] \rightarrow$
$(Eu)[Cu \wedge (Ay)[[[Cy \wedge Pyy] \rightarrow \sim Puy] \wedge [[Cy \sim Pyy] \rightarrow [Puy \wedge Oub]]]]$

The conclusion is that an algorithm to solve the halting problem does not exist:

$$G : \sim (Ex)[Ax \wedge (Ay)[Cy \rightarrow (Az)Dxyz]]$$

Failure to Prove the Halting Problem by Resolution

Massimo Bruschi reported the following facts in [2]: "No mention of [automated solutions of] this problem has been made in subsequent issues of the Newsletter." He also noted that he "was unable to obtain a mechanical proof of the theorem simply by applying ENprover" and added that applications of OTTER (Argonne's theorem prover) were also unsuccessful. Bruschi felt that the cause of the problem was "the number and the length of the clauses derived from the transformation of the given input: 84 from the axiom and 2 from the negation of the thesis, some of them with 7 literals."

Bruschi had to introduce new predicate symbols by hand to simplify the structure of the formulation. These reduced the number of the clauses and literals occurring in the clauses, hence reducing the complexity of the problem. ENprover was able to get a proof of the reformulation after this manual conversion of the original formulation. Bruschi concluded that no automated theorem prover can make the automatic transformation.

The Mechanical Proof in Natural Deduction Style

We have run the ANDP system interpretively on a Sun workstation at the AI laboratory of Tsinghua University. The natural deduction proof of the problem has 173 steps; after the useless lines are removed, the proof has 93 steps. The total CPU time required for solving the problem is 1.5 seconds. (We estimate that if ANDP were compiled, the CPU time would be $1.5/20 \approx 0.01$ second.)

The natural deduction proof of the problem is direct, not a refutation from the negation of the conclusion. It uses the following rules of inference in 93 steps.

Rules of inference for propositional logic and times used

MP	LDS	SIMP	DeMorgan
1	14	20	5
CP	IMP	CASES	$\sim$ ELIMINATION
2	1	4	4

Rules of inference for predicate logic and times used

US	UG	EE (or ES)
8	1	5

ANDP uses a strategy for minimal scopes of quantifiers. That is, given a formula to be proved, ANDP will find a formula with minimal scopes of quantifiers being equivalent to the formula. For the latter, there are more chances to apply to it the rules of inference for propositional logic. Thus, it reduces the complexity of the problem to be proved.

Let us look at the outline of the proof of the problem. First, $P1$ is divided into two cases by the rule CASES:

Case 1 is $\sim (Ex)[Ax \wedge (Ay)[Cy \rightarrow (Az)Dxyz]]$.
Case 2 is $(Ew)[Cw \wedge (Ay)[Cy \rightarrow (Az)Dwyz]]$.

If {Case 1 of $P1$, $P2, P3, P4$} $\vdash G$ and {Case 2 of $P1, P2, P3, P4$} $\vdash G$ are proved, then $\{P1, P2, P3, P4\} \vdash G$ is proved.

It is easy to prove {Case 1 of $P1, P2, P3, P4$} $\vdash G$ because Case 1 of $P1$ is the same as G. To prove {Case 2 of $P1, P2, P3, P4$} $\vdash G$, $P3$ is divided into two cases by the rule CASES:

Case 1 is $\sim (Ew)[Cw \wedge (Ay)[[[Cy \wedge Pyy] \rightarrow [Qwyy \wedge Owg]] \wedge [[Cy \wedge \sim Pyy] \rightarrow [Qwyy \wedge Owb]]]]$.
Case 2 is $(Ev)[Cv \wedge (Ay)[[[Cy \wedge Pyy] \rightarrow [Pvy \wedge Ovg]] \wedge [[Cy \wedge \sim Pyy] \rightarrow [Pyy \wedge Ovb]]]]$.

If {Case 2 of $P1, P2, P3$, Case 1 of $P3, P4$} $\vdash G$ and {Case 2 of $P1, P2$, Case 2 of $P3, P4$} $\vdash G$ are proved, then {Case 2 of $P1, P2, P3, P4$} $\vdash G$ is proved.

It uses 14 steps to prove {Case 2 of $P1, P2$, Case 2 of $P3, P4$} $\vdash G$. The final rule of inference is the rule $\sim$ ELIMINATION. This means that it infers a contradiction from Case 2 of $P1, P2$ and Case 2 of $P3, P4$ and that {Case 2 of $P1, P2$, Case 2 of $P3, P4$} is contradictory.

To prove {Case 2 of $P1, P2$, Case 1 of $P3, P4$} $\vdash G$, $P4$ is divided into two cases by the rule CASES:

Case 1 is $\sim (Ev)[Cu \wedge (Ay)[[[Cy \wedge Pyy] \rightarrow [Pvy \wedge Ovg]] \wedge [[Cy \wedge \sim Pyy] \rightarrow [Pyy \wedge Ovb]]]]$.
Case 2 is $(Eu)[Cu \wedge (Ay)[[[Cy \wedge Pyy] \rightarrow Puy] \wedge [[Cy \wedge \sim Pyy] \rightarrow [Puy \wedge Oub]]]]$.

Then {Case 2 of $P1, P2$, Case 1 of $P3$, Case 1 of $P4$} $\vdash G$ and {Case 2 of $P1, P2$, Case 1 of $P3$, Case 2 of $P4$} $\vdash G$ have to be proved. It uses 13 steps to prove Case 2 of $P1, P2$, Case 1 of $P3$, Case 2 of $P4$} $\vdash G$. Finally, it uses the rule $\sim$ ELIMINATION to infer G. Further, it means that it infers a contradiction from Case 2 of $P1, P2$, Case 1 of $P3$ and Case 2 of $P4$ and that {Case 2 of $P1, P2$, Case 1 of $P3$, Case 2 of $P4$} is contradictory.

To prove {Case 2 of $P1, P2$, Case 1 of $P3$, Case 1 of $P4$} $\vdash G$, it infers 26 steps, using the rule CASES once more and the rule $\sim$ ELIMINATION in each subcase.

Other Information That the Natural Deduction Proof Gives

It is clear that if $\{A\} \cup W$ and $\{B\} \cup W$ are contradictory, then $\{A \vee B\} \cup W$ is also contradictory, where W is a set of wffs, and A and B are wffs. By the reason above, {Case 2 of $P1, P2$, Case 1 of $P3$, Case 1 of $P4$} is contradictory. Because {Case 2 of $P1, P2$, Case 1 of $P3$, Case 2 of $P4$} is contradictory, {Case 2 of $P1, P2$, Case 1 of $P3, P4$} is contradictory. Also, because {Case 2 of $P1, P2$, Case 2 of $P3, P4$} is contradictory, {Case 2 of $P1, P2, P3, P4$} is contradictory.

We have four premisses for the halting problem, $P1 \wedge P2 \wedge P3 \wedge P4 =$ (Case 1 of $P1 \vee$ Case 2 of $P1$) $\wedge P2 \wedge P3 \wedge P4 =$ (Case 1 of $P1$) $\wedge P2 \wedge P3 \wedge P4 \vee$(Case 2 of $P1$) $\wedge P2 \wedge P3 \wedge P4 =$ (Case 1 of $P1$) $\wedge P2 \wedge P3 \wedge P4$ (because (Case 2 of $P1$) $\wedge P2 \wedge P3 \wedge P4$ is a contradition) $\Rightarrow$ (Case 1 of $P1$) $= \sim (Ex)[Ax \wedge (Ay)[Cy \rightarrow (Az)Dxyz]]$ (that is the conclusion).

Question: Can we impose some restrictions on the four premisses of the halting problem such that $\{$(Case 2 of $P1$), $P2, P3, P4\}$ is satisfiable and (Case 1 of $P1$) is not the conclusion? If so, then we can get a stronger theorem than the halting problem.

References

1. Andrews, P. B., *An Introduction to Mathematical Logic and Type Theory: To Truth through Proof,* Orlando, Academic Press, 1986.

2. Bruschi, M., The halting problem, *AAR Newsletter* 17, March 1991.

3. Burkholder, L., A 76th automated theorem proving problem, *AAR Newsletter* 8, April 1987.

4. Li Dafa, Unification algorithm with quantifiers and applications to automatic natural deduction, *Proceedings of the 4th Florida Artificial Intelligence Reserch Symposium,* Florida, April 2–5, 1991, pp. 196–200.

5. Manna, Z., *Mathematiccal Theory of Computation,* McGraw-Hill, New York., 1974.

6. Pelletier, J., Seventy-five problems for testing automatic theorem provers, *J. Automated Reasoning* 2 (1986) 191–216.

Editor's Note: A paper copy of the 93-step proof of the halting problem is available. If you wish a copy, send e-mail to pieper@mcs.anl.gov. We also have a paper copy of some rules of inference in the ANDP system.

Computers Do Produce Beautiful Mathematics

Larry Wos
Argonne National Laboratory

In an article entitled "Computers Still Can't Do Beautiful Mathematics," which appeared in the Week in Review Section E of the *New York Times* on July 14, 1991, Gina Kolata argued that computers are replacing reasoning by calculation. Kolata claimed that "instead of converging on an answer through a long chain of logical deduction, mathematicians now often use computers to systematically test and discard a vast number of possible answers until they hit on the right one." Calling the increasing number of computer-aided proofs "a festering problem," she implied that such proofs are mathematically unsatisfying principally because "a lone human sitting in a room [cannot] go through [them] line by line and understand."

I disagreed vehemently with Kolata's arguments and conclusions. I called the *New York Times* and was encouraged to write a response. I did so, but that response was never published. Nevertheless, through word of mouth, many people have learned of my reply and have asked that I print it. The following is the article I sent to *New York Times* in 1991. I still agree with the points I made then; indeed, I am more convinced than ever!

Computers do produce beautiful mathematics. In drawing the opposite conclusion, the July 14 article published in the "Week in Review" section overlooked the entire field of automated reasoning. The science and art so evident in the research of mathematicians are now complemented by the assistance provided by a new type of computer program that reasons logically. The field whose research has culminated in the design of such programs is called automated reasoning. Computers do produce beautiful mathematics. In drawing the opposite conclusion, Kolata's article overlooked the entire field of automated reasoning.

Rather than simply calculating what is needed to examine a myriad of possibilities, automated reasoning programs now produce proofs that are sequences of logical steps, each step drawn from earlier steps. Indeed, what makes the results provided by a computer so satisfying is that they include the history of the steps of which the result consists. Thus a mathematician can sit alone in an office and enjoy results that share the magic often found in publications in which computers play no role. Further, the mathematician can quickly learn how the proof works, often generalize the approach taken, sometimes encountering deep mathematical truths, and without the need to trust the program that produced the proof.

In addition to exhibiting logical reasoning of the type found in mathematics, reasoning programs produce results that are startling and elegant. Dr. J. Lukasiewicz was well recognized for his contributions to areas of logic, and yet the program OTTER recently found a proof far shorter and more elegant than that produced by this eminent researcher, and the program used the same notation and style of reasoning. Mathematicians and logicians find elegance in shorter proofs.

Indeed, mathematicians of note are more than interested in what might be done with an automated reasoning program. Dr. I. J. Kaplansky of the University of California at Berkeley once suggested an open question for study by the computer. That question was answered, and answered by reasoning and not by calculation. Dr. John Kalman of the University of Auckland posed a number of questions concerning possible axioms for a field in logic. Contrary to the conjecture that each of the formulas under study was too weak, a computer program provided a complicated and lengthy proof showing that two of them offer the needed and sought-after power. The proofs are precisely in the style that Kalman used in some of his published papers, offering a line of logical thought that could be checked for accuracy.

Knowing that a reasoning program can produce such proofs, a scientist might wish to have a personal copy of such a program. That wish is easy to satisfy. Some reasoning programs run well on workstations, on personal computers, and on the Macintosh. Evidence of how well such programs run is provided by the following occurrence.

In August of 1990, Dr. Dana Scott of Carnegie Mellon University attended a workshop at Argonne National Laboratory. There he learned of OTTER and some of its uses and successes. Upon returning to his university, Dr. Scott's curiosity prompted him to suggest (via electronic mail) 68 theorems for consideration by the computer. His curiosity was almost immediately satisfied, for the sought- after 68 proofs were returned with the comment that all were obtained in a single computer run with the program–and in less than 16 CPU minutes on a Sun 4 workstation. Dr. Scott now uses his own copy of OTTER on his Macintosh.

The type of program that has produced these results and that we expect will continue to do so is in no way a replacement for the mathematician or logician. Nor will it ever be. Instead, this type of program provides an automated reasoning assistant whose work and methods complement those of the scientist. Reasoning assistants like OTTER are not designed in the classical notion of artificial intelligence; these programs do not take an approach that a person would take. Quite the reverse, for automated reasoning programs rely on techniques and procedures that a person would prefer not to use. Nevertheless, their use has resulted in answers to questions that were and are of interest to experts in various fields.

Dr. R. Smullyan of the University of Indiana showed great pleasure and surprise at learning of some of the successes achieved by an automated reasoning program. As evidence of his interest, he posed a number of questions, receiving in turn the answers to all but one of them—a question that is still open.

Without the following features, so evident in the program OTTER, far less beauty and elegance would have been produced by a program reasoning logically. This newest reasoning program, designed by Dr. William McCune of Argonne National Laboratory, can be instructed to seek shorter proofs. Such proofs are cited by Dr. E. Dijkstra of the University of Texas as important to proving the correctness of computer programs. OTTER can be given hints and suggestions based on the researcher's knowledge and intuition. The program can be instructed to seek one, two, or as many proofs as the allotted time allows, and then present each of the proofs with enough detail to permit checking for accuracy. What makes this program of particular appeal is its use of strategy, one type to restrict its reasoning, and one type to direct it.

Rather than detracting from the beauty, elegance, and even magic of mathematics and logic, the use of a computer program—especially a reasoning program—can add to each. It has already. Some of the proofs found with such a program have been published in journals well respected by scientists, for example, the *Journal of Algebra* and the *Notre Dame Journal of Formal Logic*. Those proofs are logical in structure, not ones of computation.

When I was a student in the Mathematics Department at the University of Chicago almost four decades ago, I would have thought it impossible for a computer to provide such proofs. After all, computers were to be used for calculations, mainly of a numerical character, not for logical reasoning. However, as it turns out, computer programs do reason, and reason so effectively that open questions are sometimes answered. Indeed, in my role as mathematician, I constantly use such a program (OTTER) in my own research.

As for the difference of opinion expressed here and in the article published in the *New York Times* on July 14, the most probable explanation is the simplest: Most likely the scientists quoted in the article have no acquaintance with the program OTTER and with the successes obtained with it. Clearly, the enthusiasm of Dr. Scott, Dr. Smullyan, Dr. Kalman, and Dr. Kaplansky must be well placed. When I once asked Dr. Kaplansky about the automation of reasoning in the context of proving theorems, he replied, "You have a friend in court. Should you continue in your research, the mathematics papers written in 2060 or thereabouts will be at a new level of reading, with much of the current level within reach of a theorem-proving program."

The beauty and magic of mathematics will never die, or even lessen. A proof consisting of a sequence of steps, each obtained by applying logical reasoning, is a marvel—whether produced by a person or produced by a computer!

Conference Announcements

EMCSR 1994

The Twelfth European meeting on cybernetics and system research will take place on April 5–8, 1994, at the University of Vienna, Austria. The following sessions are of particular interest to AAR members:

- fuzzy sets, approximate reasoning, and knowledge-based systems;
- intelligent autonomous systems;
- artificial intelligence and cognitive science; and
- artifical intelligence and systems science for peace research.

For information about submission of papers, contact

Robert Trappl
Department of Med. Cybernetics and AI
University of Vienna
Freyung 6/2
A-1010 Vienna
Austria
e-mail: sec@ai.univie.ac.at

General Announcements

Logic Preprints Available from Indiana University

Logic preprints are available to the public through FTP from the directory `pub/logic` on cs.indiana.edu (129.79.254.191). Abstracts of papers are in the directory `pub/logic/ABSTRACTS`, and archive information is in the file `pub/logic/README`. At present, available preprints are authored or coauthored by Gerard Allwein, Jon Barwise, Michael Dunn, Joseph Goguen, Yuri Gurevich, Daniel Leivant, Jean-Yves Marion, Jose Meseguer, Alice ter Meulen, Lawrence Moss, Jerry Seligman, and Satish Thatte.

Researchers Welcome to Join EACSL

The European Association for Computer Science Logic (EACSL) welcomes new members. Founded in July 1992, EACSL

- organizes the annual conference Computer Science Logic and publishes its proceedings;
- organizes and sponsors summer schools; sponsors workshops, meetings, and publications in the field;
- cooperates with related societies and institutions;
- and broadcasts an electronic newsletter to its members.

Membership is open to interested people from all countries. Benefits include discounts on CSL registration and on proceedings and publications, electronic information, and reciprocity agreements with related societies and associations.

The annual fee is 20 DM (10 DM for students and for people from countries with currency problems), payable in cash or by postal money order (to EACSL, account no. 3000 47-751, Postgiroamt Karlsruhe (BLZ 660 100 75), 7500 Karlsruhe, Germany) or by Eurocheque (mailed to Klaus Ambos-Spies, Mathematisches Institut, Universitat Heidelberg, Im Neuenheimer Feld 288, W-6900 Heidelberg 1, Germany). Provide full name and address, e-mail, and fax (when available).

Call for Papers

3rd International Symposium on Artificial Intelligence and Mathematics

The International Symposium on Artificial Intelligence and Mathematics will be held on January 2–5, 1994, in Fort Lauderdale, Florida. This the third of a biennial series featuring applications of mathematics in artificial intelligence as well as artificial intelligence techniques and results in mathematics. There has always been a strong relationship between the two disciplines; however, the contact between practitioners of each has been limited, partly by the lack of a forum in which the relationship could grow and flourish. This symposium represents a step towards improving contacts and promoting cross-fertilization between the two areas.

Please submit five copies of extended abstracts (up to 10 double-spaced pages) by July 30, 1993, as follows. For authors from Europe, Australia, Asia, Africa: Erol Gelenbe, EHEI/Mathematiques, 45 rue des Saints-Peres, 75006 Paris, France, e-mail: erol@masi.ibp.fr. For authors from North and South America: Kedem, New York University, 251 Mercer Street, New York, NY 10012, e-mail: kedem@cs.nyu.edu. Authors will be invited to submit within one month after the symposium a final full-length version of their paper to be considered for inclusion in a thoroughly refereed volume of the series *Annals of Mathematics and Artificial Intelligence,* J. C. Baltzer Scientific Publishing Co.

For further information and future announcements contact Frederick Hoffman, Florida Atlantic University, Department of Mathematics, PO Box 3091, Boca Raton, FL 33431, USA, e-mail: hoffman@acc.fau.edu or hoffman@fauvax.bitnet.

TARK V

The 5th Conference on Theoretical Aspects of Reasoning about Knowledge (TARK V) will be held in Pacific Grove, California. March 13–16, 1994. TARK V is an interdisciplinary conference, with relevance to such fields as artificial intelligence, cryptography, distributed computing, economics and game theory, linguistics, philosophy and psychology. Topics include semantic models for knowledge and belief, bounded rationality and resource-bounded reasoning, knowledge acquisition, belief revision and learning, commonsense epistemic reasoning, knowledge and action, and applications of reasoning about knowledge and belief.

Interested authors should submit eleven copies of a detailed abstract (up to 4000 words) by September 7, 1993, to the program chair: Ronald Fagin, IBM Almaden Research Center, Dept. K53/802, 650 Harry Road, San Jose, CA 95120-6099, USA. Phone (408) 927-1726, Fax (408) 927-3030, Email: fagin@almaden.ibm.com, fagin@almaden.bitnet. Full information by anonymous FTP: external.nj.nec.com: pub/grove/tark.

KR'94

The fourth international conference on Principles of Knowledge Representation and Reasoning will be held in Bonn, Germany, on May 24–27, 1994. The conference will emphasize both the theoretical principles of knowledge representation and reasoning and the relationships between these principles and their embodiments in working systems. Submissions are encouraged in (but are not limited to) the following topic areas: representational formalisms (including various logics – temporal, spatial, taxomonic, nonmonotonic); reasoning methods and tasks (including deduction, induction, learning, analogical reasoning, and reasoning about reasoning); generic ontologies; and implementation issues (including reasoning architectures, benchmarking, and testing).

Authors are asked to submit extended abstracts (up to 12 pages) by November 8, 1993; in addition, authors are *strongly* encouraged to submit an electronic abstract of about 200 words to KR94-abstracts@medg.lcs.mit.edu to aid in the reviewing process.

CADE-12

The Twelfth International Conference on Automated Deduction will take place on June 28–July 1, 1994, in Nancy, France. The CADE conferences are the major forum for the presentation of new research in all aspects of automated deduction. Original research papers, descriptions of working reasoning systems, and problem sets that provide innovative, challenging tests for automated reasoning systems, are solicited.

CADE conferences cover all aspects of automated deduction:

First *vs.* Higher Order Logics	Classical *vs.* Non-Classical Logics
Special *vs.* General Purpose Inference	Interactive *vs.* Automatic Systems

Specific topics of interest include resolution, sequent calculus, decision procedures, unification, rewrite rules, and mathematical induction. Also of interest are applications of automated deduction, including deductive databases, logic and functional programming, commonsense reasoning, and software and hardware development.

The Proceedings of CADE-12 will be published by Springer-Verlag in their *Lecture Notes in Artificial Intelligence* series. Research papers should not exceed 15 proceedings pages; system descriptions and problem sets should not exceed 5 proceedings pages. Springer style files should be used if possible; send an e-mail with contents "HELP" to `svserv@dhdspri6.bitnet`. Alternatively, FTP anonymously from `dream.dai.ed.ac.uk` (see instructions in `pub/cade-12/README`). The title page of the submission should include the name, address (with email if possible) and telephone number of each author. Papers must be unpublished and not submitted for publication elsewhere. Submissions that are late or too long or that require major revision will not be considered.

Authors should send 4 (four) copies of their submission to the Programme Chair

Alan Bundy
Department of Artificial Intelligence
University of Edinburgh
80 South Bridge

Edinburgh EH1 1HN
Scotland

Tel: [+44] 31-650-2716
Fax: [+44] 31-650-6516
France

Tel: [+33] 83 59 30 11
Secretary: [+33] 83 59 30 09
Fax: [+33] 83 27 83 19
E-mail: `Claude.Kirchner@loria.fr`

LFCS'94

The third symposium on Logical Foundations of Computer Science will take place in St. Petersburg, Russia, July 11–14, 1994. Authors are invited to submit extended abstracts by November 29, 1993, to one of the following:

E. Ya. Dantsin
LFCS'94
Laboratory of Mathematical Logic
Steklov Institute of Mathematics
27 Fontanka
St.Petersburg 191011, Russia
Phone: (7)(812) 311-4392
Fax: (7)(812) 310-5377
E-mail: lfcs@sovam.com

(preferred for non-CIT countries)
LFCS '94
MSRI
Cornell University
407 College Ave.
Ithaca, NY 14850
Phone: (1)(607) 255-7752
Fax: (1)(607) 255-8005
E-mail: lfcs@msiadmin.cit.cornell.edu

For electronic submission consult with lfcs@msiadmin.cit.cornell.edu.