ASSOCIATION FOR AUTOMATED REASONING NEWSLETTER

No. 10 April 1988

From the AAR President, Larry Wos . . .

One of the stated objectives of AAR is the exchange of information and the enhancement of communication. I am pleased to say that the newsletters do, in fact, in part fulfill this objective. In that regard, please note the item on mini-conferences sent to us by Frank Brown.

Conferences

Workshop on Automated Reasoning

The seventh annual workshop on automated reasoning will be held at Argonne National Laboratory on June 9-10, 1988. The workshop will consist of a set of lectures that introduce the elements of automated reasoning and that focus on a number of applications including that concerned with answering open questions. The fee will be $70, which will cover the cost of a written copy of the lectures and two books on automated reasoning: *Automated Reasoning: Introduction and Applications* by Wos, Overbeek, Lusk, and Boyle, and *Automated Reasoning: 33 Basic Research Problems* by Wos. We must limit the number of attendees to approximately 60. If you are interested, please call Larry Wos at Argonne, 312-072-7224, or at his home, 312-493-9767.

CADE-9

The Ninth International Conference on Automated Deduction will be held at Argonne National Laboratory on May 23-26, 1988. This year's meeting celebrates the twenty-fifth anniversary of the discovery of the resolution principle at Argonne. Among the main attractions are the following:

- more than 60 presentations on various aspects of automated deduction
- invited talks from Bill Miller (SRI), J. A. Robinson (Syracuse University), and L. Wos (Argonne National Laboratory)
- facilities for demonstration of and experimentation with automated deduction systems
- tutorials in special areas including verification, rewrite systems, connection graphs, and constraint logic programming

For further information and registration materials, call Mrs. Miriam L. Holden, Director, Conference Services, Argonne National Laboratory, (312) 972-5587.

AIENG 88

The Third International Conference on Applications of Artificial Intelligence in Engineering will be held at Stanford, California, on August 8-11, 1988. The purpose of the conference is to provide an international forum for the presentation of work on the state of the art in applications of artificial intelligence to engineering. Areas include representation, problem solving, constraint reasoning, learning, robotics, diagnosis and evaluation, and tutoring.

For further information, write to AIENG 88 Conference, Computational Mechanics Institute, 25 Bridge Street, Billerica, MA 01821.

IEEE Symposium on the Foundations of Computer Science

The 29th Annual IEEE Symposium on Foundations of Ccmputer Science will be held at the Crowne Plaza Hotel in White Plains, New York on October 24-26, 1988. Suggested topics include

- Algorithms and Data Structures
- Cryptology
- Formal Languages and Automata
- Logic and Semantics of Programs
- Robotics and Machine Learning

Persons wishing to submit a paper should send 15 copies of an extended abstract by May 9 to the program chair Dexter Kozen, Department of Computer Science, 4126 Upson Hall, Cornell University, Ithaca, New York 14853-7501.

Mini-Conference on Automatic Deduction

A mini-conference on automated deduction is being organized. The focus is to be on new approaches to automatic deduction based on non-traditional approaches—topics not well represented at such conferences as CADE. These non-traditional frameworks might include both theoretical and experimental research on proof theory and deduction systems for the following non-classical logics and extensions to first order logic:

modal, nonmonotonic, default, tense, and action logics
circumscription
the frame problem
intentional reasoning
metatheory
reflective reasoning
fixed points
closed world assumption
non-Cantorean set theories
quantifier elimination
possibility, probability, and ontology

Those interested in helping to organize or participate in such a miniconference are invited to contact Dr. Frank M. Brown, Dept. of Computer Science, University of Kansas, Lawrence, Kansas 66045, (913)-864-4482.

--

FGCS '88

A call for papers has been issued for the International Conference on Fifth Generation Computer Systems 1988, to be held in Tokyo on November 28-December 2, 1988. The focus of this year's conference is on knowledge information processing, logic programming, and parallel architectures. Specific topics of interest include

Formal semantics
Computation models
Theory of parallel computation
Automated reasoning

Logic/functional/object-oriented programming
Parallel programming languages and methodologies
Program verification and debugging
Implementation techniques

Inference machines
AI and VLSI architectures
Knowledge-based machines

Natural language understanding and machine translation
Real-time AI systems
Knowledge representation and acquisition

Authors should send six copies of manuscripts to Prof. Hidehiko Tanaka, FGCS'88 Program Chairman, ICOT, Mita Kokusai Bldg. 21F, 1-4-28 Mita, Minato-ku, Tokyo 108, Japan. Papers must be received by May 18, 1988.

COLOG-88

An international conference on computer logic is scheduled for December 12-16 in Tallinn, Estonia, USSR. Topics of interest include

applications of deductive systems
deductive program synthesis and analysis
theorem proving
logic programming
computer experiments in logic-related fields

Papers should be sent to G. Mints, Institute of Cybernetics, Estonian Academy of Sciences, Akadeemia tee 21, Tallinn 200108, USSR.

--

Non-Obviousness — One More Time
(from Gerald Peterson, McDonnell Douglas Corporation)

The following "non-obvious" theorem has been discussed in AAR newsletters 6, 7, and 9:

1. $\neg P(xy)\ \neg P(yz)\ P(xz)$
2. $\neg Q(xy)\ \neg Q(yz)\ Q(xz)$
3. $\neg Q(xy)\ Q(yx)$
4. $P(xy)\ Q(xy)$
5. $\neg P(ab)$
6. $\neg Q(cd)$

If one compares the results in these previous discussions, one discovers that this theorem is a great deal more non-obvious to some theorem provers than it is to others. Peterson decided to attack the problem to see how obvious it was to a mathematician and to see whether something could be learned about automated mathematics.

One of the first things one ought to notice about this problem is that its Herbrand universe is just $\{a,b,c,d\}$ and its Herbrand base contains 32 elements. These are severe constraints which should be exploited in the proof.

Essentially, the problem is to show that there is no way to define P and Q relations between all pairs of elements taken from $\{a,b,c,d\}$ in a way such that all six of the clauses are satisfied. Peterson's approach was to build two matrices, labeled on the rows and columns with a,b,c,d. One of these indicated which P and Q relations held, and the other indicated which did not hold. In the beginning these matrices were as follows:

Holds

$$\begin{array}{c|c|c|c|c|}
 & a & b & c & d \\
\hline
a & & & & \\
\hline
b & & & & \\
\hline
c & & & & \\
\hline
d & & & & \\
\hline
\end{array}$$

Does not hold

$$\begin{array}{c|c|c|c|c|}
 & a & b & c & d \\
\hline
a & & P & & \\
\hline
b & & & & \\
\hline
c & & & & Q \\
\hline
d & & & & \\
\hline
\end{array}$$

Now clauses 1 to 4 are used to derive additional entries for the matrices. This leads easily to

Holds

$$\begin{array}{c|c|c|c|c|}
 & a & b & c & d \\
\hline
a & Q & Q & & \\
\hline
b & Q & Q & & \\
\hline
c & & & P & P \\
\hline
d & & & P & P \\
\hline
\end{array}$$

Does not hold

$$\begin{array}{c|c|c|c|c|}
 & a & b & c & d \\
\hline
a & & P & & \\
\hline
b & & & & \\
\hline
c & & & & Q \\
\hline
d & & & Q & \\
\hline
\end{array}$$

Every interpretation on the Herbrand base will assign $Q(ca)$ either **T** or **F**. If we try **T** and get a contradiction, and try **F** and get a contradiction, we are finished. Each of these was tried by filling in the appropriate slot and then continuing to derive consequences as before. Each led in a few steps to a contradiction, so the proof was complete.

This idea can be incorporated into a theorem prover. All clauses that are not ground units are used only as operators to produce additional ground units. The operation is performed by matching all but one of the literals in the clause with an opposite ground unit. As we proceed, either a proof will be generated or there will come a time when additional ground units cannot be generated. When generation ceases, select some ground unit that has not been generated and add it to the set of ground units and continue generating. Add additional selected ground units if necessary as the proof proceeds. When a proof is generated, back up to the point that the last selected clause was added, add its negation instead, and continue generating, and so on. If the tree that is being generated can be completed with a proof at each leaf, then the proof is complete. This approach leads to the following 18-step proof of the original problem:

7. $Q(ab)$	4,5
8. $Q(ba)$	3,7

9. $Q(ca)$	assumed	9'. $\neg Q(ca)$	opposite assumed
10. $Q(cb)$	2,7,9	10'. $\neg Q(ac)$	3,9'
11. $\neg Q(a,d)$	2,6,9	11'. $P(ac)$	4,10'
12. $\neg Q(bd)$	2,6,10	12'. $\neg P(cb)$	1,5,11'
13. $\neg Q(db)$	3,12	13'. $Q(cb)$	4,12'
14. $P(db)$	4,13	14'. $Q(ca)$	2,8,13'
15. $P(ad)$	4,11	15'. $\square$	9',14'
16. $P(ab)$	1,14,15		
17. $\square$	5,16		

Peterson notes that this approach is probably going to work only when the number of elements in the Herbrand base is small. He adds two morals:

1. Constraints existing in theorems ought to be exploited by theorem provers.

2. Present-day provers could be improved by incorporating several proof methods and building a front-end discriminator that would select the method based on features of the theorem.

The Logic of Skolem Functions: A Subtle Construction and a Subtle Error

(from Joseph S. Fulda, Hofstra University)

The use of Skolem functions in resolution-refutation proofs is typical AI: it works, so we use it. But replacement of an existentially quantified variable with a Skolem function does not result in a perfect translation. This is evident when considering the negation of an existentially quantified expression: $\neg(Ex)Px$ gives the clause $\neg Px$, not $\neg Pj$, even though we would replace $(Ex)Px$ with Pj. This is because a function has *exactly* one output, while a variable bound to an existential quantifier has *at least* one substitution instance. Yet it works, since in any given existentially quantified expression no one can be sure that the number of substitution instances of the existentially bound variable exceeds 1 (and if one was sure, one should not use just a single existential quantifier). Thus, replacing existence with unique existence—something of a mathematician's nightmare—in the limited context of resolution theorem proving does no harm.

The following example of a resolution-refutation proof brings these issues into relief.

Premise 1.
Every person with a lover is a romantic.

Clause 1.
$(x)(Px \rightarrow ((Ey)(Lxy \rightarrow Rx))$ $\neg Px \ v \ \neg Lxf(x) \ v \ Rx$

Premise 2.
Jonathan has Kim for a lover.

Clause 2.
Ljk

Clause 3.
Pj (implied premise)

Conclusion.
Jonathan is a romantic.

Clause 4.
$\neg Rj$ (denial of conclusion)

The unification of $\neg Lxf(x)$ with Ljk is often explained with something akin to Digricoli's RUE (resolution by unification and equality) rule: $f(j)$ must equal k because $f(j)$ is a function, and if Ljk is given, then $Ljf(j)$ must equal Ljk (the negation here doesn't change the argument). The argument seems reasonable but is, in fact, unsound—precisely because we dared replace existence with unique existence. Yet this example (and isomorphics) may well be unique.

What, then, of the original argument about Jonathan? Is it valid? Yes, and when translated properly, a resolution-refutation proof is obvious. The first premise should have been translated $(x)(Px \rightarrow ((Ey)\ (Lxy) \rightarrow Rx)$; that is, a scoping error was made. The error is subtle in the sense that the scope of the quantifier matters, even though the predicate involved has no occurrence of the quantified variable. The subtlety disappears, however, if the implication is replaced by its definition $\neg ... v ...$ or if exportation is used to rewrite the proposition.

Replacement by its definition shows, in particular, that it is the universal quantifier that is indicated, giving a clausal form of $\neg Px \ v \ \neg Lxy \ v \ Rx$. It also reminds us of the generality that $(Ex)(Px \rightarrow Q)$ is different from $(Ex)(Px) \rightarrow Q$ and that the latter is, indeed, $(x)(Px \rightarrow Q)$ and that this generality holds for all Q without a free occurrence of x. Using $\neg Px \ v \ \neg Lxy \ v \ Rx$ allows a derivation of *nil* in just three steps, and without elaborate justifications.

Acknowledgments: The author thanks Professor M. Levin for his helpful and clarifying comments.

Some Fixed Point Problems in Combinatory Logic

(from Bill McCune and Larry Wos, Argonne National Laboratory)

Raymond Smullyan's book *To Mock a Mockingbird* (Knopf, 1985) contains a wealth of problems in applicative systems and combinatory logic. Most can be attacked in first-order logic with equality, and some of those provide quite a challenge for theorem-proving programs. This note contains a few fixed point problems from Smullyan's book, and many additional fixed point problems.

Some basic proper combinators and their equations (reduction rules) are

$Sxyz = xz(yz)$	$Qxyz = y(xz)$
$Kxy = x$	$Q_1xyz = x(zy)$
$Ix = x$	$Txy = yx$
$Bxyz = x(yz)$	$Cxyz = xzy$
$Lxy = x(yy)$	$Vxyz = zxy$
$Mx = xx$	$Hxyz = xyzy$
$Wxy = xyy$	$Nxyz = xzyz$
$W^1xy = yxx$	$Uxy = y(xxy)$
$L_2xy = y(xx)$	$S_2xyz = xz(yy)$
$Oxy = y(xy)$	

Notation. Variables are x, y, and z. There exists an implicit binary operator 'apply', and left association is assumed when parentheses are omitted. For example, the clause

$$a(a(a(S,x),y),z) = a(a(x,z),a(y,z))$$

might be used to present the first equality in the preceding list to a theorem prover.

We focus on proving that one or both of the following fixed point properties holds, given a set of proper combinators. The *weak fixed point property* is

$$\forall y \exists t (t = yt),$$

and the *strong fixed point property* is

$$\exists \Theta \forall x (\Theta x = x(\Theta x))$$

(Θ is a *fixed point combinator*). Note that the weak fixed point property follows immediately from the strong fixed point property by letting t be Θy, for any fixed point combinator Θ. The respective denials (Skolemized and in clause notation) of the two fixed point properties are

$x \neq a(f,x)$ (denial of weak fixed point property)

and

$a(y,f(y)) \neq a(f(y),a(y,f(y)))$ (denial of strong fixed point property)

For example, to prove that the weak fixed point property holds for $\{M, B\}$, a program could refute the following set of clauses.

$a(M,x) = a(x,x)$
$a(a(a(B,x),y),z) = a(x,a(y,z))$
$x \neq a(f,x)$

Problems (some are easy, some are hard)

Prove that the weak fixed point property holds for each of the following sets.

$\{L\}$, $\{M, B\}$, $\{B, L_2\}$, $\{B, S_2\}$, $\{O, Q_1\}$

Prove that the strong fixed point property holds for each of the following sets. (Also prove that the weak fixed point property holds for each of the following sets).

$\{U\}$, $\{S, L\}$, $\{L, O\}$, $\{Q, M\}$, $\{B, M, L\}$, $\{B, M, T\}$, $\{W, Q, L\}$, $\{B, S, T\}$, $\{B, S, C\}$, $\{B, M, V\}$, $\{B, O, M\}$, $\{B, N\}$, $\{B, M, C\}$, $\{B, W\}$, $\{B, W^1\}$, $\{B, H\}$, $\{B, N\}$, $\{S, K\}$
✓5 ✓w45

New Book

A new textbook entitled *Artificial Intelligence: A Knowledge-Based Approach,* by M. Firebaugh, has been published by Boyd and Fraser. Of particular interest to AAR newsletter readers is Chapter 5, "Automated Reasoning." Other features include

- a brief introduction to LISP and PROLOG,
- emphasis on building expert systems, with examples
- a comprehensive introduction to robotics
- an introduction to massively parallel architectures for AI