Association of Automated Reasoning Newsletter No. 4

From the President

This issue of the AAR newsletter continues the dual focus established in earlier newsletters--a focus on problem sets, open questions, and puzzles on the one hand, and on announcements about workshops, journals, and such on the other. AAR members are encouraged to send any appropriate material to us at Argonne National Laboratory. We shall be pleased to announce current research, new software, experimental results, new books, new problems for testing programs, and descriptions of open questions. We are also interested in letters that contribute information on topics discussed in these newsletters; this issue contains such a letter, a valuable contribution from Dallas Lankford.

New Journals

JAR

The *Journal of Automated Reasoning* published its first issue in February 1985. Featured in that issue is an overview article, actually a compendium of articles, each covering some field for which automated reasoning is relevant. The overview not only presents the goals and current research of the individual fields but also explains how the fields collectively share the interest of automating the process known as reasoning. Eight articles are included:

- What Is Automated Reasoning? - L. Wos
- Logic Programming - F. Pereira
- Research in Intelligent Robots - R. Hong
- Program Verification - R. S. Boyer and J S. Moore
- What Is Automated Theorem Proving? - W. W. Bledsoe and L. J. Henschen
- Expert Systems - B. G. Buchanan
- Nonclassical Logic Theorem Proving - G. Wrightson
- What Is Program Synthesis? - C. Green

JAR continues to seek papers; descriptions of actual successes with existing automated reasoning programs are of especial interest. And a reminder: one of the benefits of AAR membership is that members can subscribe to the journal at a reduced cost. The subscription price is $27 for private members of AAR; private non-members of AAR are charged $36, and institutions $78. Questions about manuscript submissions or membership may be addressed to

L. Wos
Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, Illinois 60439
(312) 972-7224

In Europe, subscription information is available from

Kluwer Academic Publishers Group
P.O. Box 322
3300 AH Dordrecht
Holland

CC-AI

A new international publication, the *Journal for the Integrated Study of Artificial Intelligence, Cognitive Science and Applied Epistemology*, or *CC-AI*, has been announced. The journal will cover such areas as knowledge representation, expert systems, planning and search systems, cognitive modelling, logic programming, applied epistemology, and general aspects of artificial intelligence.

The first issue of *CC-AI* is devoted to expert systems, including articles on

+ strategy for building expert systems
+ knowledge extraction
+ knowledge representation
+ tools for developing expert systems
+ E2S's use of PERQ in developing an expert system

Articles, book reviews, advertising, and news items may be sent to

Michele Drolet, Managing Editor
CC-AI
Blandijnberg 2
B-9000 Ghent, Belgium

Coming Workshops and Meetings

ANL Tutorial/Workshop on Automated Reasoning

On June 4-5, 1985, the automated reasoning group of the Mathematics and Computer Science Division at Argonne National Laboratory is giving its fourth tutorial/workshop on automated reasoning. The workshop consists of a set of lectures based on the book *Automated Reasoning: Introduction and Applications*, by Wos, Overbeek, Lusk, and Boyle. The focus will be on the elements and applications of automated reasoning, including research in mathematics, logic circuit design and validation, and proving properties of computer programs. Attendance is by invitation; anyone interested in attending may call Mike Harris (312-972-5767) or Larry Wos (312-972-7224).

Workshop on Knowledge Engineering/Expert Systems

The twenty-fourth annual workshop sponsored by the Western Committee of the IEEE Computer Society will be held September 4-6, 1985, at the UCLA Conference Center at Lake Arrowhead. The subject of the workshop is "Knowledge Engineering: How?" Sessions are planned on knowledge acquisition, knowledge representation, inferencing strategies, and programming environments.

Attendance is by invitation. People working in the knowledge engineering and expert systems area are encouraged to contact the program chairperson

Greg Kearsley
Courseware, Inc.,
10075 Carroll Canyon Road
San Diego, Ca 92131
(619) 578-1700

Logic, Language, and Computation Meetings

The Stanford Center for the Study of Language and Information and the Association for Symbolic Logic are sponsoring two major events for July: a CSLI Summer School (July 8-13) and an ASL Meeting (July 15-19).

Courses for the summer school include the following:

+ Logic programming and Prolog
+ A semantical reconstruction of Lisp
+ Abstract data types
+ Situation theory

Invited addresses to be presented at the ASL meeting range from resolution and model theory to the polymorphic typed lambda calculus to automated reasoning applications.

For further information, write to Ingrid Deiwiks, CSLI, Ventura Hall, Stanford, CA 94305 (Telex: EURTEL 290163 Code 235).

IBM Europe Seminar

The IBM Europe Institute is offering a one-week seminar on Knowledge-Based Systems and Logic Programming in Lech/Oberlech (Austria) on July 29-August 2, 1985. The topics will include representation, reasoning, programming methodology, logic programming, and machine learning and concept formation.

For further information, write to

J. P. Adam
IBM France, Centre Scientifique
36 Avenue Raymond Poincare
75116 Paris, France

Symposium on Artificial Intelligence in Engineering

A symposium entitled Artificial Intelligence in Engineering will be held at George Washington University on October 21-23, 1985. Sessions at the symposium will cover AI in research and development, design, test and evaluation, project and production management, military systems, and technology insertion. Both government and industrial viewpoints will be presented.

To submit a paper for presentation, send a 50-word abstract by May 30 to

Dr. Barry G. Silverman
Institute for Artificial Intelligence
Gelman Library - Room 636A
George Washington University
Washington, DC 20052

Open Questions

Attempting to answer open questions with the aid of a reasoning program is exciting and challenging. Such an activity also contributes to the likelihood of automated reasoning eventually producing a computer program that functions as a high-level reasoning

assistant. We encourage AAR readers to submit open questions to this newsletter. Presented below are two open questions sent to us recently.

Combinator Theory (A. Meyer)

The following question was originally raised by Myhill.

Determine whether or not the following set of clauses is satisfiable.

1. f(f(K,x),y) = x
2. f(f(f(S,x),y),z) = f(f(x,z),f(y,z))
3. f(x,h(x,y)) ≠ f(y,h(x,y)) x = y
4. K ≠ S
5. u = w f(g(u,v,w,x),u) = v
6. u = w f(g(u,v,w,x),w) = x

(K and S are constants, and u,v,w,x,y are variables.) Clauses 1 through 4 are a first-order axiomatization of the untyped lambda calculus, and clauses 5 and 6 are a statement of double transitivity. Meyer claims that if a model exists, then it must be infinite.

Latin Squares (C. C. Lindner)

It is known that any $n \times n$ Latin square defines a quasigroup on n elements. A separation of an $n \times n$ Latin square $A=(a_{ij})$ into two $n \times n$ matrices $B=(b_{ij})$ and $C=(c_{ij})$ is defined by

$$b_{ij}=b_{ji}=a_{ij} \quad \text{and} \quad c_{ij}=c_{ji}=a_{ji}$$

or

$$b_{ij}=b_{ji}=a_i \quad \text{and} \quad c_{ij}=c_{ji}=a_{ij}$$

What kinds of conditions on A guarantee that there is at least one separation of A into a pair of commutative quasigroups?

Note: If A is symmetric, then a separation exists (here we have $A=B=C$).

Schubert's Steamroller Problem with Linked UR-Resolution

(W. McCune)

The following is a statement of the Steamroller Problem (taken from Christoph Walther, "Schubert's Steamroller--A Case Study in Many-Sorted Resolution," SEKI memo, Karlsruhe, 5/84):

"Wolves, foxes, birds, caterpillars, and snails are animals, and there are some of each of them. Also there are some grains, and grains are plants. Every animal either likes to eat all plants or all animals much smaller than itself that like to eat some plants. Caterpillars and snails are much smaller than birds, which are much smaller than foxes, which in turn are much smaller than wolves. Wolves do not like to eat foxes or grains, while birds like to eat caterpillars but not snails. Caterpillars and snails like to eat some plants. Therefore there is an animal that likes to eat a grain-eating animal."

The following set of clauses is a denial of the theorem.

1. Wolf(Lupo);
2. Fox(Foxy);
3. Bird(Tweety);
4. Caterpillar(Maggie);
5. Snail(Slimey);

6. Grain(Stalky);
7. -Wolf(x1) animal(x1);
8. -Fox(x1) animal(x1);
9. -Bird(x1) animal(x1);
10. -Caterpillar(x1) animal(x1);
11. -Snail(x1) animal(x1);
12. -Grain(x1) plant(x1);
13. -animal(x) -plant(y) -animal(z) -Smaller(z,x) -plant(w) -eats(z,w) eats(x,y) eats(x,z);
14. -Caterpillar(x1) -Bird(x2) Smaller(x1,x2);
15. -Snail(x1) -Bird(x2) Smaller(x1,x2);
16. -Bird(x1) -Fox(x2) Smaller(x1,x2);
17. -Fox(x1) -Wolf(x2) Smaller(x1,x2);
18. -Bird(x1) -Caterpillar(x2) eats(x1,x2);
19. -Caterpillar(x1) plant(f1(x1));
20. -Caterpillar(x1) eats(x1,f1(x1));
21. -Snail(x1) plant(f2(x1));
22. -Snail(x1) eats(x1,f2(x1));
23. -Wolf(x1) -Fox(x2) -eats(x1,x2);
24. -Wolf(x1) -Grain(x2) -eats(x1,x2);
25. -Bird(x1) -Snail(x2) -eats(x1,x2);
26. -animal(x1) -animal(x2) Grain(f3(x2,x1));
27. -animal(x1) -animal(x2) -eats(x1,x2) -eats(x2,f3(x2,x1));

Note: The phrase "grain-eating animal" is interpreted as "animal that eats all grains" rather than "animal that eats some grains" or "animal that eats grain only." Either-or is usually interpreted as exclusive or, but here it is interpreted as disjunction. Also, the sentence "Caterpillars and snails like to eat some plants" is interpreted as "Caterpillars like to eat some plants, and snails like to eat some plants" rather than "There are some plants that caterpillars and snails like to eat." Several different clause sets for this problem have appeared in the literature.

The following proof was discovered by the LMA-based theorem prover tp0 using linked UR-resolution.

29. -eats(Foxy,f2(Slimey)); (6,24,1,13,7,1,12,6,8,2,17,2,1,21,5,23,1,2)
57. eats(Tweety,Stalky); (6,12,13,9,3,11,5,15,5,3,21,5,22,5,25,3,5)
58. eats(Foxy,Tweety); (57,13,8,2,21,5,9,3,16,3,2,12,6,29)
62. contradiction; (58,27,8,2,9,3,13,9,3,12,26,8,2,9,3,11,5,15,5,3,21,5,22,5,25,3,5)

The time required to find this proof was about 2 minutes on a VAX 11/780. The number of linked UR-resolvents inferred was 92, of which 35 were retained. There were 2663 successful unifications.

Briefly, linked UR-resolution (linked unit-resulting resolution) (Wos et al.,"The Linked Inference Principle II: The User's Viewpoint," in the Proceedings of CADE 7) is a rule that can be used to infer unit clauses from a set of clauses. It differs from standard UR-resolution in that more than one nonunit clause can be used to infer a linked UR-resolvent. Our implementation of linked UR-resolution has the following features. (Factoring, which is required for many non-Horn problems, is not performed.)

- Various options and parameters can be used to limit the length or depth of deductions that result in linked UR-resolvents, to limit the frequency of use of certain clauses in deductions, and to limit the kinds of clauses that can be used in deductions.

- A user can require that linked UR-resolvents have certain properties. This kind of restriction is called the target strategy. If the target strategy is being used, then various

parameters can be used to control it.

• Although the rule is defined as being "unit-resulting," a contradiction will be reported if it is discovered.

For the steamroller problem, the target strategy was used in such a way that all linked UR-resolvents must be positive or negative units in the "eats" predicate. This target strategy was chosen through semantic considerations: it would seem more helpful to know who is or is not eaten by whom, rather than, for example, who is or is not a plant. The number of clauses participating in the deduction of a linked UR-resolvent was limited to 30. The initial set of support was the single clause Grain(Stalky).

Some comments are in order. First, we solved this problem with standard UR-resolution also. On this problem, the use of linked UR allows a smaller initial set of support than does standard UR-resolution. (This has been the case with a number of other problems as well.) The initial set of support was the 6 unit clauses; 131 clauses were generated, of which 124 were retained. There were 26,190 successful unifications. It took about 11 minutes on a VAX 11/780. Aside from the question of whether or not linked UR leads to quick proof discovery, its use in this example leads to a natural proof. The steps are large enough to be interesting, but not too large to be grasped. Finally, it should be noted that in the report cited above, Walther shows that many-sorted resolution is also effective on this problem.

Comments on Graduated Problems for Testing Equality Reasoning

(D. Lankford)

The following letter was sent by Dallas Lankford in response to the discussion of graduated problems for testing equality reasoning by R. Overbeek and E. Lusk in the AAR Newsletter #3.

The earliest published computer proof of problem 1 that I have found is in Huet's "Experiments with an interactive prover for logic with equality," Case Western Reserve University, Jennings Computing Center, Report 1106, 1972, pp. 41-42 and pp. 62-63. The computer proof required five rounds of resolution and paramodulation, used some equations as rewrite rules, generated seventeen clauses during the proof search, and required about twelve seconds of CPU. Mike Ballantyne and I subsequently derived a complete set for this problem using completion and commutative-associative completion. The complete set for problem 1 is as follows:

1. $[x^2] \rightarrow [e]$
2. $[x^{-1}] \rightarrow [x]$
3. $[x \cdot e] \rightarrow [x]$

where congruence classes are defined by the commutative and associative axioms. This and other computer experiments with the completion method were presented by Ballantyne and me at the 3rd CADE, MIT, August 1977, including complete sets for Abelian groups and commutative rings. The most impressive solution of problem 2 is contained in one of the computer experiments by Knuth and Bendix where the first complete set for free groups was derived. (Their paper is found in *Computational Problems in Abstract Algebras*, Pergamon Press, 1970, and in Vol. 2 of *Automation of Reasoning*, Springer-Verlag, 1983.) Problem 3 also has a solution by completion, apparently first observed by Hsiang in his July 1981 paper, "Refutational theorem proving using term rewriting systems." The complete set is

1. $[x+0] \rightarrow [x]$
2. $[x+x] \rightarrow [0]$
3. $[x \cdot 1] \rightarrow [x]$

4. $[x \cdot x] \rightarrow [x]$
5. $[x \cdot (y+z)] \rightarrow [(x \cdot y)+(x \cdot z)]$
6. $[x \cdot 0] \rightarrow [0]$
7. $[-x] \rightarrow [x]$

where congruence classes are defined by the commutative and associative axioms for addition and multiplication. The first computer proof of problem 4 that I know of is contained in Nevins's 1974 *JACM* paper, required 30 minutes of CPU time, and generated a search space of over 400 formulas. Subsequently a completion-based theorem prover implemented by Ballantyne and Lankford solved problem 4 in 30 seconds, and terminated with a search space of 11 formulas (cf. Bledsoe's "Non-resolution theorem proving" in *IJCAI-75* and *AI Journal*, 1977.) A variant of problem 5 is proved by Nevins's computer program; see his 1974 *JACM* paper. Concerning the difficulty of problem 6, it depends on how much information is given to the computer program. In Veroff's computer proof, much information was provided by the human in the form of considerable clausal information, and so the computer proof was relatively easy. By contrast, little information about the problem was given to his program by Stickel, and so the computer proof was quite difficult. Moreover, Stickel's program found a decision algorithm for free $(x^3 = x)$ - rings, which is a much deeper result than just showing $(x^3 = x)$ - rings are commutative. Because complete sets are decision algorithms, the computer completion proofs for problems 1, 2, and 3 also found decision algorithms for $(x^2 = e)$ - groups, groups, and $(x^2 = x)$ - rings (i.e., Boolean rings). Whether completion decision algorithms exist for $(x^3 = e)$, groups and ternary Boolean algebras (problems 4 and 5) appears to be currently unknown. In my opinion, these are two very important open problems in applied equational logic. Although it is open whether complete sets exist for problems 4 and 5, problem 4 is known to be decidable. Groups satisfying $x^n = e$ are called Burnside groups, and have word problem decision algorithms for n = 2, 3, 4, and 6, and for odd $n \geq 665$, see Adian's *The Burnside Problem and Identities in Groups*, Springer-Verlag, 1979, p. 250. It is unknown whether tractable computer implementations of the approach described by Adian can be developed.

Logic Problems

(from C. Morgan and solved by E. Lusk and R. Overbeek)

In the last AAR newsletter, we included the following logic problems from Charles Morgan:

1. P(i(x,i(y,x)))
2. P(i(i(x,i(y,z)),i(i(x,y),i(x,z))))
3. P(i(i(n(x),n(y)),i(y,x)))
4. If P(i(x,y)) & P(x) then P(y)

Roughly speaking, P means "is provable", i means "implies", and n means "not". Thus Axiom 4 prepresents the inference rule of modus ponens.

Problem 1: For all x, P(i(x,n(n(x)))).

Problem 2: For all x, P(i(n(n(x)),x)).

Harder problems arise when Axiom 3 above is replaced by

3'. P(i(i(y,x),i(n(x),n(y))));

So the next problem is:

Problem 3: With axiom 3 replaced by axiom 3', P(i(x,n(n(x)))) for all x.

Here are some of the experiments we did: the first is a proof of Theorem 2, and the second is a proof of Theorem 1 from Theorem 2.

1 P(i(x1,i(x2,x1)));

```
  2  P(i(i(x1,i(x2,x3)),i(i(x1,x2),i(x1,x3))));
  3  P(i(i(n(x1),n(x2)),i(x2,x1)));
  4  If P(i(x1,x2)) & P(x1) then P(x2);
  5  -P(i(n(n(a)),a));
  8  P(i(x1,i(i(n(x2),n(x3)),i(x3,x2)))); 3 4 1
 12  P(i(i(x1,x2),i(x1,x1))); 2 4 1
 16  P(i(i(x1,i(n(x2),n(x3))),i(x1,i(x3,x2)))); 2 4 8
 18  P(i(x1,x1)); 12 4 1
 20  P(i(x1,i(x2,x2))); 18 4 1
 21  P(i(i(i(x1,x2),x1),i(i(x1,x2),x2))); 18 4 2
 55  P(i(i(i(x1,x1),x2),x2)); 21 4 20
 64  P(i(x1,i(i(i(x2,x2),x3),x3))); 55 4 1
 79  P(i(i(x1,i(i(x2,x2),x3)),i(x1,x3))); 64 4 2
268  P(i(n(x1),i(x1,x2))); 16 4 1
296  P(i(n(n(x1)),i(x2,x1))); 268 4 16
316  P(i(n(n(x1)),x1)); 296 4 79
328  null; 316 5
```

```
  4  If P(i(x1,x2)) & P(x1) then P(x2);
  5  -P(i(a,n(n(a))));
  6  P(i(n(n(x1)),x1));
  9  P(i(x1,n(n(x1)))); 6 4 3
 10  null; 9 5
```

An NFL Scheduling Problem

(H. Mills)

Here is an innocent-sounding problem that was given to Harlan Mills by the National Football League in 1970: "For the 1970 season, 26 teams had formed a league committed to 182 games over a 14-week season. The League was divided into two 13-team conferences, each with divisions of 4, 5, and 4 teams. Each team was to play 7 games at home with other specified teams, and 7 games away with still other."

There were other complications, such as games to be played within a division, interconference games, and special deals, but the basic problem is clear: 182 games to be scheduled in 14 weeks.

The NFL did arrive at a solution, thanks to Harlan Mills. Bill McCune, too, has solved the problem interactively. Can you discover an effective algorithm to solve the problem? (Note: This deceptively simple problem has been described as "what may very well be the most complex, non-computerized, man-made jigsaw puzzle ever attempted.")