ASSOCIATION FOR AUTOMATED REASONING NEWSLETTER

No. 19 December 1991

From the AAR President, Larry Wos . . .

Never before has your membership in AAR provided such a deal! Starting with 1992, AAR members will receive *half off* the subscription price to the *Journal of Automated Reasoning.* Yes, that's *half off* the price—instead of $131, AAR members pay only $65. And, AAR membership is still only $5. Naturally, we expect a rush on AAR membership renewals and new subscriptions, so we encourage you to renew your membership soon (forms are enclosed).

The bargain is even greater when one realizes that in 1992, the *Journal of Automated Reasoning* will expand to six issues, for a total of approximately 840 pages. This expansion is in recognition of the tremendous growth in the field of automated reasoning.

We are especially pleased to announce the start of a new section, entitled "Studies in Automated Reasoning." Its editor will be Dr. Ewing Lusk of the Mathematics and Computer Science Division at Argonne National Laboratory (who for the past seven years has been editor of the Problem Corner). This new section will feature papers that include substantial detail concerning methodology for solving problems, papers that offer questions that are particularly amenable to study with an automated reasoning program, and papers that one might conjecture will spawn immediate research in a particular area. We encourage AAR members to submit manuscripts for publication in this new section of the *Journal*; manuscripts should be sent to directly to Dr. Ewing Lusk, Editor, *Journal of Automated Reasoning,* Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439 (lusk@mcs.anl.gov). Papers that simultaneously emphasize advances in the areas of theory, implementation, or application are especially welcome.

OTTER 2.2 Is Now Available

William W. McCune
(mccune@mcs.anl.gov)

Version 2.2 of OTTER has been released. Improvements over Version 2.0 include the following (2.1 was never released):

- The ratio strategy for selecting the given clause combines selection by weight with breadth-first search.
- The passive list can be used to prevent some of the input clauses from entering the search.
- New built-in $ functions perform bit operations.

- On Unix systems, the user can interrupt OTTER during its search and change the flag and parameter settings.
- Indexing can be used for back demodulation.
- The notion of the answer literal has been extended to terms with $IGNORE.
- The inference rule negative hyperresolution has been included.
- The user can assign a maximum to the number of distinct variables in clauses.
- A new automatic memory-control feature has been added.
- Parent lists can be ordered in a new way.
- $ operations during hyperresolution are much more efficient.

To obtain a copy by FTP, connect to info.mcs.anl.gov, with username anonymous, and with your name as password. Go to pub/Otter/Otter-2.2, and follow the directions in README.FTP. Please let me know (otter@mcs.anl.gov) if you take a copy by FTP. Two versions of the manuals are included: one is LaTeX input, and the other is already formatted by nroff.

Distribution of the PC and Macintosh versions has not yet been set up. (If you can FTP and then get the files to your PC or Macintosh, that might work, because the FTP version contains the PC- and the Mac-executable files.) (The second edition of *Automated Reasoning: Introduction and Applications*, by Wos et al., contains a PC diskette of OTTER 2.2 and should be available from McGraw-Hill on January 17, 1992.)

For those who don't like parentheses, experimental version 2.2xa will soon be available by FTP in pub/Otter/Otter-2.2xa. That version will accept infix, prefix, and postfix operator declarations as in most Prolog systems.

For further information, write to W. McCune, MCS-221, Argonne National Laboratory, Argonne, IL 60439-4844 (otter@mcs.anl.gov).

Call for Papers

Navy Verification Workshop

A call for papers has been issued for the Workshop on the Effective Use of Automated Reasoning Technology in System Development, to be held in Washington, D.C., on April 6–8, 1992.

The purpose of this workshop is to explore issues in the application of automated reasoning technology to systems with critical requirements, to investigate how different automated tools might be brought to bear on a single system specification, and to consider how future automated tools might be organized to promote greater interchangeability of component parts. Areas of interest include the following:

- Application of different automated reasoning tools to the same problem or specification.
- Methods for exchanging useful data among automated theorem provers, software engineering tools, and other analysis tools.
- Methods for promoting the interchangeability of components of automated reasoning tools.
- Establishing a formal basis for exchanging information among automated reasoning tools and between formal specification processing tools and automated theorem provers.
- Effectiveness of various user interfaces and interaction styles for automated reasoning tools.
- Significant applications of automated reasoning technology to system development.

The workshop is sponsored by the Technical Panel on Trustworthy Computing Technology (XTP-1) under The Technical Cooperation Program (TTCP), an agreement under which the United States, United Kingdom, Canada, Australia, and New Zealand share defense research and development information.

Attendance will be limited to 40 individuals and is by invitation on the basis of a submitted paper, extended abstract, or position statement. Four copies of submitted manuscripts are due January 10, 1992, and should be sent to Carl E. Landwehr, XTP-1 Workshop, Code 5542, Naval Research Laboratory, Washington, D.C. 20375-5000 (phone 202-767-3381; e-mail landwehr@itd.nrl.navy.mil). Copies of all accepted papers will be mailed to participants prior to the workshop and will be published subsequently in a publicly available TTCP report.

Computer-Aided Verification

The Fourth Workshop on Computer-Aided Verification will be held in Montreal, Quebec, Canada, on June 29 - July 1, 1992.

This workshop is the fourth in a series dedicated to bringing together researchers and practitioners interested in the development and use of methods, tools, and theories for the computer-aided verification of concurrent systems. The goal of the workshop is to compare various verification methods and practical tools that can be used to assist the applications designer. Emphasis will be placed on new research results and applications of existing results to real verification problems. Special sessions for the demonstration of verification tools will be organized. A balanced participation of researchers and practitioners is sought. Papers are solicited on the following topics:

* Verification and validation tools for hardware and software systems, including protocols, distributed systems, real-time control systems, and digital circuits and systems

* Verification and validation methods based on model checking, theorem proving, and automata-based methods

* Verification theories and their applicability

* Complexity and efficiency issues in automatic verification

Papers in related areas that fit with the intentions of the workshop will also be considered. An author may submit a paper by mailing five copies of a preliminary version, not more than twelve double-spaced typed pages, to the Program Chairman: Gregor von Bochmann, Université de Montreal, Departement d'IRO, Room V-240, 2900, boul. Edouard-Montpetit, Montreal, Quebec H3C 3J7 Canada (514-343-7484; e-mail: bochmann@iro.umontreal.ca).

The submission deadline is February 1, 1992. There will be a Proceedings at the workshop; the final versions of accepted and invited papers will be published after the workshop.

Conference on Logic Programming and Automated Reasoning

LPAR'92 is the successor of the first and second Russian conferences on logic programming held in Irkutsk in 1990 and in St. Petersburg in 1991. Russia has good traditions in mathematical logic and automated reasoning; there is also a rapidly growing logic programming community. The aim of LPAR is to bring together researchers who hardly had the opportunity to meet in the past. LPAR'92 will take place on the ship *Michail Lomonosov* during July 12–17, 1992—the famous "white nights" of this region of Russia.

Topics of interest include the following:

- ⋆ Applications
- ⋆ LPAR in artificial intelligence
- ⋆ Common sense reasoning
- ⋆ Constraints
- ⋆ Deductive databases
- ⋆ Executable specifications
- ⋆ Implementation techniques
- ⋆ Inference systems for LPAR
- ⋆ Languages for LP and its extensions
- ⋆ Meta-programming
- ⋆ Parallelism and concurrency
- ⋆ Program synthesis and verification
- ⋆ Programming in constructive logic
- ⋆ Rewriting
- ⋆ Theorem proving
- ⋆ Theory and foundations
- ⋆ Unification theory

Among the invited speakers are Pascal van Hentenryck (Brown University), Steffen Hölldobler (University of Darmstadt, Germany), Vladimir Lifschitz (University of Texas), Ewing Lusk (Argonne National Laboratory), and Gregory Mints (Institute of Cybernetics, Estonia – Stanford University).

Authors are invited to submit *long papers* (up to 12 pages), *short papers* (up to 6 pages), or *system descriptions* (up to 3 pages) written in English. The affiliation of the authors should be given, including a full address and an e-mail address, telephone, or telefax number. Late, too long, and e-mail submissions will not be considered. The proceedings will be published by a major publisher, most likely Springer Verlag. Papers should be sent to the following address to arrive *before February 15, 1992:* Andrei Voronkov, ECRC, Arabellastrasse 17, D–W 8000 Munich 81, Germany.

Thirty minutes will be given for presenting long papers, and fifteen minutes for short papers and system descriptions. A special session will be devoted to system descriptions. It will be possible to demonstrate systems implemented on IBM PC and compatibles.

CSL'92

The sixth workshop on *Computer Science Logic* will be held from September 28 to October 2, 1992, in San Miniato, a historic town near Pisa (Italy).

The workshop is intended for computer scientists whose research activities involve logic as well as for logicians working on algorithmic aspects of logical problems. The scientific program will consist of invited lectures and of short contributions, which will be selected from the submitted papers. All contributions will be refereed for a proceedings volume.

Six copies of an extended abstract (1–2 pages) of papers to be submitted should be sent to the program committee chairman (E. Börger) to arrive not later than June 1, 1992.

The workshop has been made possible by the contribution from CNR (Italian National Research Council) and will take place in the Centro Studi *I Cappuccini*, made available by Cassa di Risparmio di San Miniato.

Correspondence should be sent to Prof. Dr. E. Börger / Dr. S. Martini, CSL'92, Dipartimento di Informatica, Università di Pisa, Corso Italia 40, I - 56125 Pisa, Italy (e-mail: csl92@di.unipi.it)

IWAR'92

An International Workshop on Automated Reasoning will be held in Beijing, China, on July 13–16, 1992. The workshop will bring together researchers, developers, and users of artificial intelligence, mathematics, and logic to share information and to explore future directions in the field of automated reasoning.

Authors are invited to submit original papers on topics from the following list:

- ⋆ Analogical reasoning
- ⋆ Constraint reasoning
- ⋆ Geometric reasoning
- ⋆ Multiagent reasoning
- ⋆ Nonmonotonic reasoning
- ⋆ Pansystem theory
- ⋆ Qualitative reasoning
- ⋆ Search reasoning
- ⋆ Temporal reasoning
- ⋆ Uncertainty reasoning

Three copies of an original paper (up to 5,000 words) in English are required. The paper should include the author's name, affiliation, complete address, and telephone and fax number. Submissions should be sent by March 1, 1992, to Prof. Zhongzhi Shi, Institute of Computing Technology, Academia Sinica, P.O. Box 2704, Beijing 100080, China (86-1 2565533-419).

Proof Methods for Non-Horn Sets

Chris Merz and Ralph Wilkerson
Department of Computer Science
University of Missouri-Rolla
Rolla, MO 65401

In its purest form, resolution is a brute-force, semi-decidable, and combinatorially explosive approach to automated theorem proving. Numerous strategies have been devised to help focus this process, such as the set of support strategy, input resolution, unit resolution, linear resolution, binary resolution, hyperresolution, unit-resolvent resolution, and lock resolution. It is well known that some of these procedures are complete only for Horn sets; and since many problems cannot be formulated as Horn sets, a method for directing the resolution process on non-Horn sets could be advantageous. Peterson [3] generalized input resolution and unit resolution to apply to non-Horn sets by integrating them with a specialized lock resolution procedure. The Lock-T proof method generalizes unit resolution and is a breadth-first search for the empty clause, while the LNL-T proof method generalizes input resolution and is a depth-first search for the empty clause. Peterson postulated that these two strategies would work well together if run in parallel and allowed to share "important" intermediate results known as lemmas. The implementation of these methods was done by modifying the automated theorem prover, OTTER, developed at Argonne National Laboratory by McCune [1]. This permitted the use of existing routines that handled the internal representation, manipulation, and bookkeeping of the clauses. Test problems from Tarskian geometry [4] and the nonobvious problem [2] were used to evaluate algorithm performance.

A non-Horn set S can also be thought of as a Horn$_T$ set, where T is a set of literal occurrences chosen from the nonunit clauses of S such that the deletion from S of all the literal occurrences in T changes S into a Horn set. Any literal in T will be called a T-literal, and any clause composed entirely of T-literals is a T-clause or a T-lemma. Using Peterson's terminology, we can say that a lock negative linear T-lemma (LNL-T) proof of a clause C is a lock proof of C in which the only lemmas are T-clauses, and each lemma and C are proved linearly. In such proofs, the side clauses are either members of S or earlier proved lemmas, and the center clauses are composed only of negative literals and T-literals. The top clause in an LNL-T proof must be negative, and factoring is performed only on lemmas and only on literals with the same ancestor and with the lowest lock in their clause. A lock-T proof of a clause C is a lock proof of C in which factoring only occurs in T-clauses and only on literals of smallest lock and of common ancestry.

Both of the methods are actually specializations of lock resolution, where LNL-T proofs have restrictions on the locking of literals and for choosing clauses to be resolved. Lock-T proofs only have restrictions on the locking of literals. The implementation actually resulted in the addition of three resolution proof techniques, the first of which was lock resolution. The first modification was to allow OTTER to accept a new input format for clauses with numbered/locked literals. When any lock resolution-based inference method is used, the new format requires literals in any list of clauses to be preceded by an "@", followed by a "T" if it is a T-literal, followed immediately

by an integer and a blank. For instance, an acceptable form of

$$\{-P_3(x)|-L_6(x,y)|L_{T7}(g(z),y)\}$$

would be

$$\{@3\ -P(x)|@6\ -L(x,y)|@T7\ L(g(z),y)\}$$

Since Lock-T resolution is a specialization of lock resolution, it was implemented with a simple call to the lock resolution routines. The preprocessing conditions for the initial clause set are handled at the time of input where the initial clause set is inspected to see whether it satisfies the locking criteria and it is indeed a Horn$_T$ set. The postprocessing stipulations are that factoring is only allowed on T-lemma resolvents and only on the lowest locked literals, and that lemmas are crossfed to the LNL-T proof strategy if it is active.

LNL-T refutation posed the most significant implementation problem because its depth-first pursuit of lemmas is not compatible with the set of support strategy which is the basis of OTTER's main loop. Thus, OTTER had to be modified in such a way as to allow LNL-T proofs to be conducted independently of the other inference mechanisms. Recall that in each iteration of the main loop, a given-clause is moved from the set of support list to the list of axioms and each active inference mechanism is called upon to resolve the given-clause against the other axioms. Since OTTER is a serial program, only one inference mechanism can be active at one time, so the pursuit of LNL-T proofs cannot be completely independent of the other active inference mechanisms. In order to accomplish the necessary time slicing, LNL-T-proof is called each time through the main loop allowing each LNL-T proof to be advanced a little further by an amount determined by the user. Lemmas generated by LNL-T-proof are crossfed to the other active inference mechanisms by placing them in the set of support. In order to sustain the development of the LNL-T proofs, the set of support list must never become empty before LNL-T-proof has finished. When this happens (if LNL-T-proof is turned on), the last given-clause is retained as dummy set of support clause until LNL-T-proof generates and crossfeeds a lemma to replenish the set of support, which in turn also has the effect of reviving the active set of support based resolution strategies and the main loop continues.

One non-Horn problem used to test the newly implemented proof strategies is known as the "non-obvious problem" [2]:

$$\{-P(x,y)|-P(y,z)|P(x,z)\}$$
$$\{-Q(x,y)|-Q(y,z)|Q(x,z)\}$$
$$\{-Q(x,y)|Q(y,x)\}$$
$$\{P(x,y)|Q(x,y)\}$$
$$\{-P(a,b)\}$$
$$\{-Q(c,d)\}$$

In order to avoid the incorporation of any heuristics in the clause set, two of the most generic of the various possible Horn$_T$ locking schemes that satisfy only the minimum locking restrictions for the two techniques were used. That is, in all of the Horn clauses, the negative literals are locked

at the same level, and the positive literal in each clause has the lowest lock. As for the non-Horn clause, the first positive literal is locked lowest, while the second literal is locked as a T-literal, and vice versa. The two unit clauses represent the negation of the theorem and are placed in the set of support.

All of the successful runs find proofs of about the same length (28 steps), however, the Lock-T-proof method exhibited the best overall performance by generating and keeping fewer clauses on both sets of clauses. The first locking for these clauses is

$$\{-P_1(a,b)\}$$
$$\{-Q_1(c,d)\}$$
$$\{-P_2(x,y)|-P_2(y,z)|P_1(x,z)\}$$
$$\{-Q_2(x,y)|-Q_2(y,z)|Q_1(x,z)\}$$
$$\{-Q_2(x,y)|Q_1(y,x)\}$$
$$\{P_1(x,y)|Q_{T4}(x,y)\}$$

The second locking replaces the last clause by

$$\{P_{T4}(x,y)|Q_1(x,y)\}$$

When run with the unlocked set of clauses, hyperresolution produced 1,995 clauses, retaining 355 for a proof of length 18; binary resolution produced 673 clauses, retaining 283 for a proof of length 22; and UR-resolution was unable to find a proof. Lock-T resolution generated 201 clauses, retaining 87; thus Lock-T refutation outperformed each of these methods in clause generation and retention. Some other, less general locking schemes were tried where the first negative literals (if any) in each clause were locked at 2 and the second negative literals (if any) were locked at 3 and all other literals were locked as before. This resulted in proofs of about the same length but with as few as 105 clauses generated and 72 clauses kept.

References

[1] McCune, W. W. (1990). "OTTER 2.0 Users Guide", Argonne National Laboratory, Mathematics and Computer Science Division, Argonne, Ill.

[2] Pelletier, F. J., and Rudnicki (1986). "Non-Obviousness," *AAR Newsletter,* **6.**

[3] Peterson, G. E. (1976). "Theorem proving with lemmas," *Journal of the ACM,* **23**(4), 574–581.

[4] Quaife, A. (1989). "Automated development of Tarski's geometry," *Journal of Automated Reasoning,* 5, 97–118.