REPORT DOCUMENTATION PAGE

4. TITLE AND SUBTITLE
QLISP for Parallel Processors

6. AUTHOR(S)
John McCarthy

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Computer Science Department
Stanford University
Stanford, CA 94305

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
naming agency:
SPAWAR 3241C2
Space & Naval Warfare Systems
Command
Washington, D.C. 20363-5100

monitoring agency:
ONR Resident Representative
Mr. Paul Biddle
Stanford Univ., 202 McCullough
Stanford, CA 94305

11. SUPPLEMENTARY NOTES
Approved for public release: distribution unlimited

13. ABSTRACT (Maximum 200 words)
See attached report.
"The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government."
Qlisp Final Report

Stanford University has completed the initial phase of the Qlisp project for research on parallel Lisp programming. The research made use of the Qlisp implementation on the Alliant FX/8 delivered by Lucid, Inc. under subcontract to Stanford University. The results we have obtained confirm, in general, our predictions that a shared-memory multiprocessor is an effective tool for executing symbolic programs in Lisp.

The major highlights of Qlisp have been:

- Demonstrating that Qlisp could be implemented as originally specified, and incorporating additional features that we found to be necessary or desirable.
- Discovering the importance of dynamic control of parallelism.
- Implementing the basic parts of several major Lisp applications, and finding that Qlisp was a good vehicle for expressing the parallelism in these programs.

The implementation of Qlisp proceeded with few difficulties. Several major enhancements to Lucid Common Lisp, that had been envisioned as necessary, were performed. These include deep binding (for special variables); support for multiple control stacks; and the removal or synchronization of many parts of the system that otherwise would not run correctly in parallel.

The control of parallelism at runtime was known from the start to be an important part of programming in Qlisp. Our belief in the best way to accomplish this, however, has changed. Experiments with control by cutting off parallelism based on the height or depth of a program's computation tree showed that this method was hard to deal with and not as flexible as we had hoped. Cutting off parallelism based on the runtime state of the machine, however, performed better than we expected and is now our preferred method. Along the way we discovered what kinds of programs benefit the most from this method (those whose computation trees are fairly balanced), and found theoretical justification for the efficacy of dynamic control, which had not been done before.

The applications with which Qlisp has been tested include small and medium-sized programs in the areas of:

- polynomial arithmetic and symbolic computation
- theorem proving
- functional programming and semantics
- production systems
- backtracking search
Several members of the Qlisp project attended the US/Japan Parallel Lisp Workshop in June 1989. Dan Pehousheck attended the Oregon Workshop on Parallel Lisp held in June 1990. Lucid gave Qlisp presentations to the Computer Science Department and Coordinated Science Lab of the University of Illinois, Los Alamos National Labs and Sandia Labs.

The work on Qlisp is reported in publications and reports listed at the end of this report. Qlisp has been a powerful tool, and fairly easy to use, in most of these areas. Our best results have been in “and-parallel” computations, as found in polynomial manipulations, functional programs, and many parts of theorem proving. Potential available parallelism is easy to achieve in the context of only 8 processors. The next phase of research should include both more substantial applications and experiments with systems of 20-30 processors, to verify that the initial results scale. In addition, we still need more experience, and Qlisp needs more tuning, to handle well the “or-parallel” computations found in many search algorithms.
Qlisp users

The following is a list of people who have used Qlisp.

Qlisp people at Stanford:
- Ian Mason
- John McCarthy
- Dan Pehoushek
- Igor Rivin
- Kelly Roach
- Dan Scales
- Carolyn Talcott
- Joe Weening

Qlisp people at Lucid:
- Dick Gabriel
- Ron Goldman
- Richard Mlynarik
- Peter Benson
- Carol Sexton
- Harlan Sexton
- Claire Wolfe

Other Stanford researchers and visitors:
- Charles Buckley — computer algebra and geometric reasoning
- Anoop Gupta — production systems
- Hiroshi Okuno — production systems

Berkeley researchers:
- Jim Larus — parallel compilation techniques
- Carl Ponder — computer algebra
Qlisp Publications


