USING THE PROCESS TRILLIS TO ORGANIZE LARGE-SCALE PARALLEL REALTIME MONITORS AND EXPERT SYSTEMS (U)

Professor David Gelernter

Yale University
Dept of Computer Science
New Haven CT 06520

Funding Numbers

2304/FS
AFOSR-91-0098

Performing Organization Report Number
AFOSR-TR-95-0195

Performing/Monitoring Agency Name(s) and Address(es)

AFOSR/NI
110 Duncan Ave, Suite B115
Bolling AFB, DC 20332-0001

Sponsor/monitoring agency report number
AFOSR-91-0098

Supplementary Notes

Approved for public release: distribution is unlimited

Abstract

Accomplishments under this grant were: The researchers defined a new "sensor/actuator" view of process trellis software architecture for data fusion. The trellis architecture was ported to a LAN environment. The researchers tested a new LAN-capable sensor/actuator package by developing a monitor-controller for the Piranha adaptive parallelism environment. A trellis-structured wide area Piranha system is the next goal of the researchers.

19950403 041
ANNUAL FINAL Project Report
AFOSR-91-0098
"Using the process trellis to organize large-scale parallel realtime monitors and expert systems"

Submitted by:

David Gelernter, Associate Professor
Principal Investigator
Department of Computer Science
Yale University
51 Prospect Street
New Haven, CT 06520
(203) 432-1278
gelernter@cs.yale.edu
Final Report on AFOSR-91-0098

Research in connection with this grant accomplished a number of things:

1. We defined a new “sensor/actuator” view of the process trellis software architecture for data fusion. The original architecture served only in the role of passive monitor. The sensor/actuator version can serve not only to monitor but to control realtime processes. The sensor/actuator extension is a natural one in the trellis context, but required extensions to the design and the implementation of the existing software.

2. We ported the trellis architecture to a LAN environment. The original (parallel) program ran only on shared-memory multiprocessors. The code is written in C-Linda, which is portable to essentially any asynchronous parallel environment, including LANs. But the trellis package depends not only on the correct execution of code; it also performs (heuristically) optimized realtime scheduling of trellis modules onto available nodes. The heuristic scheduler is based on an analytic model of the hardware. Porting to the LAN environment required the development of a new (and more complicated) underlying model and heuristic scheduler.

3. We tested the new LAN-capable sensor/actuator package by developing a monitor-controller for the Piranha adaptive parallelism environment. Piranha is a system that allows processes of a parallel application to be created dynamically (for example on newly-idle LAN nodes) and removed dynamically (for example, when an owner resumes work at his node) while the computation as a whole continues without interruption. The Piranha system poses a number of monitoring and control problems: it requires that the current idle/busy and “idleness criteria” status of all nodes be maintained, that predictions be developed with respect to likely future idleness patterns of each node in the pool, that the status of all Piranha applications and their behaviors be maintained and that Piranha jobs be assigned to particular idle nodes, among other issues. The problem was a good test bed for the sensor/actuator LAN trellis because it required actuator and not just sensor capability, it was inherently distributed and required LAN capacity, and it was inherently a significant, interesting problem. The Piranha-trellis we developed
worked sufficiently well to suggest that the entire Piranha system (and not just monitor-control functions) might be structured as a trellis. A trellis-structured wide area Piranha system is our next research goal.

Publications


