International Journal of Parallel Programming, Vol. 32, No. 4, August 2004 (© 2004)

Guest Editor's Introduction

This is the second special issue of the *International Journal of Parallel Programming* based on the 17th International Conference on Supercomputing (ICS'03).

ICS is the premier international forum for the presentation of research results in high-performance computing systems. In its 17th year, the conference also included invited talks, tutorials, workshops, panels, and exhibits, and was held at the Crowne Plaza Union Square hotel in San Francisco during June 21–26, 2003.

Papers were solicited on all aspects of research, development, and application of high-performance systems, including new experimental and commercial systems, architectures with fine and coarse grain parallelism, grid computing, novel infrastructures for the Internet, parallel I/O and storage, embedded and power-aware computer architectures, operating systems and support software, restructuring and optimizing compilers, program development tools, high-performance Java, performance evaluation studies, numerical or non-numerical algorithms, and computationally challenging scientific applications. Out of a large number of fine papers submitted, we could choose only 35 for presentation at the conference due to space limitations.

After the conference, six of the presented papers were picked for possible publication in two special issues of the *IJPP*. The authors were requested to expand and rewrite their conference papers and submit them again for another round of refereeing. A special issue with the first set of three papers from that collection has already appeared. We present in this issue of the *IJPP* the remaining three papers. A brief introduction to each paper follows.

Large shared-memory multiprocessors with a Cache-Coherent Non-Uniform Memory Access (CC-NUMA) architecture have a serious memory latency problem. Since memory nodes are physically distributed across the system, the cost of memory access increases with the distance of the node accessing the data from the node that contains the data. A good

0885-7458/04/0800-0259/0

initial memory placement is no guarantee for good performance. This forces an operating system keen on improving data locality to provide support for dynamic page migration and replication. The aim of the paper, "Page Migration with Dynamic Space-Sharing Scheduling Policies: The case of the SGI O2000" by J. Corbalan, X. Martorell, and J. Labarta is to demonstrate that automatic memory page migration does indeed improve the overall system performance in the CC-NUMA architecture. This is done by evaluating the migration mechanism provided by the IRIX operating system (Version 6.5) in an SGI Origin 2000 machine with 64 processors. The most important conclusion of the paper is that the automatic memory migration mechanism is sensitive to processor allocation stability.

The paper, "A Framework for Incremental Extensible Compiler Construction" by S. Carroll and C. Polychronopoulos presents a global approach to extensible and efficient compiler design that tries to improve the effectiveness and efficiency of analysis and optimization capabilities. They also present a callback system that is provided to automatically track changes to the compiler's internal representation (IR) allowing full pass reordering and an easy-to-use interface for developing lazy update incremental analysis passes. The authors demonstrate the viability of their approach by outlining their implementation of the PROMIS research prototype that was designed explicitly for extensibility. PROMIS is a multilingual parallelizing compilation system targeting a host of RISC and CISC architectures including MIPS and x86 ISAs. The design presented in the paper also addresses the difficulties commonly associated with designing incremental algorithms by providing a simple interface for tracking changes.

Optimizing and parallelizing compilers use data dependence analysis to extract the parallelism inherent in sequential programs. We guarantee that a certain compiler transformation is legal by showing that it preserves data dependence relations. Data dependence problem involving array references within loops is ubiquitous in real programs. When array subscripts and loop bounds are linear functions of loop index variables, this dependence problem turns out to be an integer-programming problem. Since integer programming has an exponential time complexity and a restructuring compiler typically needs to solve a large number of dependence problems in one instance, several dependence tests have been designed over the years to avoid using that general algorithm. At one end there are tests that are simple and work well in many (but not all) real situations. At the other end there are tests that are quite general but have the same complexity of integer programming. The paper, "Data Dependence Analysis Techniques for Increased Accuracy and Extracted Parallelism" by K. Kyriakopoulos and K. Psarris extends one known dependence test

(the I-Test) to a new test called the VI-Test that can deal with complex loop regions and symbolic variables. The authors also present experimental results on comparison of various tests using Perfect Club Benchmarks and the scientific library Lapack.

I would like to thank the authors for taking the extra time and effort to expand and improve the papers, and the referees for doing an excellent job on a tight schedule. As always, special thanks go to Ms. Melanie Sanders, the assistant to Co-Editor, Prof. Alex Nicolau, who forced all of us to do our best and smilingly endured the pain and suffering associated with this process.

Utpal Banerjee
Intel Corporation
Guest Editor