Introduction—Parallel and Distributed Computing

The research domains of parallel and distributed computing have a significant overlap. With the advent of general-purpose multiprocessors, this overlap is bound to increase. This Special Issue attempts to draw together several papers from both of these separate research domains to illustrate commonality and to encourage greater interaction among researchers in the two communities.

Papers for this Special Issue were solicited by an open call for papers as well as by direct requests to authors of papers in the last Parallel Processing and Distributed Computing Systems Conferences. 87 papers were received and 11 were accepted. Approximately 20 other papers received good reviews and, after revision, will be further processed for possible inclusion in regular issues of this Transactions.

The 11 papers in this Special Issue are categorized into eight areas: architecture, interprocessor communication, partitioning and assignment, fault tolerance, distributed programming, parallel algorithms, debugging, and performance evaluation. Obviously, many of the papers could have been placed in more than one area. I have attempted to place each paper in its most appropriate area. In the remainder of this introduction each paper is briefly summarized.

Recently, there has been an increased interest in large multiprocessors. One of the critical issues in developing such machines is dealing with hot spots, i.e., when a large number of processors attempt to access a common variable there can be significant contention and delay, referred to as hot-spot accesses. The paper entitled “Distributing hot-spot addressing in large-scale multiprocessors,” by P.-C. Yew, N.-F. Tzeng, and D. H. Lawrie shows that even if only a small percentage of all requests are to a hot spot, these requests can cause serious performance problems. The paper then develops an inexpensive software method for decreasing memory contention by distributing hot-spot accesses over a software tree whose nodes are allocated to many memory modules.

Another difficult and related problem faced by designers of parallel systems is developing a good interprocessor connection and communication scheme. The paper by R. P. Bianchini, Jr., and J. P. Shen, entitled “Interprocessor traffic scheduling algorithm for multiple-processor networks,” discusses an automated design and implementation methodology for developing an interprocessor communication function for certain classes of multiple-processor systems. The authors claim that for many mission-oriented, multiple-processor systems, the interprocessor communication is deterministic and can be specified at system inception. This specification is then automatically mapped onto a physical multiple-processor system using a (polynomial time) network traffic scheduler algorithm which is also described in the paper.

A major objective of parallel processing research is to obtain substantial speedups for given problems by utilizing a large number of processors. This Special Issue contains three papers on this topic. The first paper, entitled “Processor allocation for horizontal and vertical parallelism and related speedup bounds,” by C. D. Polychronopoulos and U. Banerjee, directly addresses this problem. This paper considers a general program represented as a graph. General bounds on program speedup are discussed, and measurements of code parallelism for the LINPACK numerical package are presented. The authors show that a high degree of parallelism is available in ordinary programs. The paper also develops a heuristic for allocating processors to general program graphs that operates in linear time.

The second paper on this topic is by Z. Cvetanovic, entitled “The effects of problem partitioning, allocation, and granularity on the performance of multiple-processor systems.” In addressing the allocation of subproblems to nodes, Cvetanovic specifically considers the effect of grain size of the subproblems and communication overhead. The results show that if a parallel algorithm allows full decomposition of processing and communication among N processors, then the speedup grows as the number of processors increases for all values of bandwidth except the worst case value. However, if the communication is not fully decomposable, then the bandwidth can seriously limit the performance. A model is presented that captures the interaction of the various parameters which influence the performance of multiple-processor systems. Various situations are analyzed including cases where processing is overlapped with communication and where processing and communication cannot be overlapped.

The third paper in this Special Issue addressing speedup in multiprocessors is the paper entitled “A mapping strategy for parallel processing,” by S.-Y. Lee and J. K. Aggarwal. This paper formulates four objective functions to evaluate the optimality of mapping a problem graph onto a system graph. An efficient mapping scheme which takes communication overhead of different applications into account is developed. The mapping scheme makes an initial assignment, and then iteratively applies a pairwise exchange scheme to this initial assignment.

A major research topic for distributed computing systems is developing techniques to realize the potential of distributed systems with respect to high reliability. However, increased reliability does not come without costs. The paper by M. Herlihy, entitled “Extending multiversion time-stamping protocols to exploit type information,” presents a reliability technique that exploits semantic information for the purpose of improved performance. The paper shows that by using a multiversion timestamping technique which relies on static preanalysis of conflicts between operations, there are fewer delays and restarts, and no additional runtime overhead. Furthermore, the technique is deadlock-free.

The Ada programming language was designed for programming embedded real-time systems. Many such systems
involve distributed computing. The paper by R. A. Volz and T. N. Mudge, entitled "Timing issues in the distributed execution of Ada programs," examines Ada constructs which deal with time. They point out a number of significant unresolved questions of interpretation of time in two Ada constructs: the conditional entry call and the timed entry call. Interesting alternative implementations for the timed entry call are proposed. The paper highlights the difficulty of implementing distributed task execution in Ada owing to the neglect of including network delay in the interpretation of conditional and timed entry calls.

Parallel systems will become more and more effective when more efficient parallel algorithms exist. Such algorithms are becoming available for many problems. The paper entitled "Optimal graph algorithms on a fixed-size linear array," by K. A. Doshi and P. J. Varman, develops parallel algorithms for computing the minimum spanning tree, bridges, and articulation points of a graph represented by its adjacency matrix. The algorithms operate on a linear array of p processors, and for a graph of n vertices require O(n^2/p) time for all p, 1 ≤ p ≤ n. The algorithms employ new data reduction techniques for solving these problems on a network with large communication latency and small communication bandwidth.

Many distributed and parallel computer systems now exist. Most of these systems and applications running in these systems are built with limited debugging support packages. The paper, "Debugging parallel programs with instant replay," by T. J. LeBlanc and J. M. Mellor-Crummey, presents a general solution for reproducing the execution behavior of parallel programs. The technique works by saving the relative order of significant events as they occur. The advantages of this technique include the ability to replay the entire program rather than individual processes, and the fact that there is less time and space needed to save the information needed for the replay compared to other methods. The paper also describes a prototype implementation on the BBN Butterfly™ Parallel Processor.

Designing and building distributed and parallel systems is a complex, costly, and time-consuming proposition. Analytical modeling serves as an important technique to avoid costly mistakes early, as an aid in developing new algorithms, and to provide predictions on the overall performance of various alternatives. Two modeling papers are included in this Special Issue. The first paper by A. E. Kamal, entitled "Star local area networks: A performance study," analyzes existing star network protocols and then proposes a new access protocol for star networks. Kamal shows that the performance of his new protocol is very close to perfect scheduling.

The second modeling paper in this issue is by D. Peng and K. G. Shin and is entitled "Modeling of concurrent task execution in a distributed system for real-time control." This paper describes how to distribute tasks over different nodes to improve response time and system reliability. The authors use a generalized stochastic Petri net (GSPN) to model tasks and precedence constraints among the tasks. A sequence of homogeneous continuous time Markov chains is then built from the GSPN to model concurrent task execution in the system. The model is applied to the computation of the probability of missing a deadline in a hard real-time system.

John A. Stankovic (S’77–M’79–SM’86) received the B.S. degree in electrical engineering, and the M.S. and Ph.D. degrees in computer science, all from Brown University, Providence, RI, in 1970, 1976, and 1979, respectively.

He is currently an Associate Professor in the Computer and Information Science Department, University of Massachusetts, Amherst. He has held visiting positions in the Department of Computer Science, Carnegie-Mellon University, Pittsburgh, PA, and at INRIA. His current research interests include investigating various approaches to distributed scheduling both on local area networks and on distributed hard real-time systems. He is also performing research on two distributed testbed systems: a distributed database testbed called CARAT, and a distributed real-time system testbed called SPRING. The CARAT testbed has been operational for several years, and the SPRING testbed is currently being developed. He has recently published a tutorial text entitled Reliable Distributed System Software, and is an Editor for the IEEE TRANSACTIONS ON COMPUTERS.

Dr. Stankovic is a member of the Association for Computing Machinery and Sigma Xi. He is the recipient of the University of Massachusetts 1983 Outstanding Scholar Award and is an IEEE Computer Society Distinguished Visitor.