JOURNAL OF PARALLEL AND DISTRIBUTED COMPUTING 16, 271–275 (1992)

Special Issue on Scheduling and Load Balancing

Guest Editors' Introduction

BEHROOZ SHIRAZI

Department of Computer Science Engineering, University of Texas at Arlington, Arlington, Texas 76019

AND

A. R. HURSON

Department of Electrical and Computer Engineering, Pennsylvania State University, University Park, Pennsylvania 16802

Recent database, real-time, defense, and large scale commercial applications have created a computation gap between the performance capabilities of existing uniprocessor systems and the performances required by such applications. A commonly agreed approach to alleviating this computation gap is through the implementation of these applications on concurrent systems; i.e., parallel or distributed systems. However, one of the important issues in such an environment is that of the operating system and the management of the concurrent processes. The problem is how to distribute (or schedule) the processes among processing elements to achieve some performance goal, such as minimum turnaround time or maximum throughput [1, 7]. From a system's point of view, this becomes a resource management problem and should be considered an important factor during the design phases of the multiprocessor systems.

Process scheduling can be performed either statically at compile time or dynamically at run time. Static scheduling methods attempt to [6, 8–10]:

- predict the program execution behavior at compile time (i.e., estimate the task execution times and communication delays);
- partition smaller tasks into coarser-grain processes in an attempt to reduce the communication costs; and
- allocate the processes to the processors.

Static scheduling suffers from a wide range of problems, most notably the following:

- Due to NP-completeness of general optimal scheduling, efforts are focused on heuristic approaches, which are often successful for special applications only.
- Lack of efficient and accurate methods for estimating task execution times and communication delays can cause unpredictable performance degradations [4, 11].
- Existing task/function scheduling methods often ignore the data distribution issue, resulting in run time communication delays in accessing data at remote sites.
- Finally, static scheduling schemes should be augmented with a tool to measure the effectivness of a schedule on a given architecture, before execution.

Dynamic scheduling is based on the redistribution of processes among the processors during the execution time. This is performed by migrating tasks from the heavily loaded processors to the lightly loaded processors with the aim of improving the performance of the application [2, 3, 5]. A typical load balancing algorithm consists of a number of components:

- Information policy, which specifies the amount of load information made available to job placement decision makers.
- Transfer policy, which determines the conditions under which a job should be migrated (e.g., current load of the host and the size of the job under consideration).
- Placement policy, which identifies the processing element to which a job should be transferred to.

The major disadvantage of dynamic load balancing schemes is the run time overhead due to load information transfer among processors, to the decision making process for the selection of processes and processors for job transfers, and to the communication delays due to task migration itself.

Our objectives for this special issue were to solicit novel techniques for addressing the above important issues and to report recent research and development efforts in the scheduling and load balancing area. More specifically, we intended to collect state-of-the-art research results in a wide range of topics in the context of task scheduling, including:

0743-7315/92 $5.00

- program and system partitioning,
- parallelism detection,
- task granularity issues,
- static scheduling,
- load balancing,
- task migration issues, and
- system monitoring and load indices measurement techniques.

A large number of high-quality manuscripts were submitted. Each paper was sent to at least four experts to be reviewed. The selection process was rather painful, since many interesting and qualifying articles had to be rejected due to the limited space. There are 11 papers in the special issue (resulting in an acceptance rate of 15%). The papers may be divided into five categories: static scheduling, load balancing, synchronized scheduling, architecture partitioning, and specialized scheduling.

STATIC SCHEDULING

The paper "A Comparison of Clustering Heuristics for Scheduling DAGs on Multiprocessors" by Gerasoulis and Yang classifies the important characteristics of clustering algorithms and presents a general model for analyzing and evaluating such algorithms. Clustering or partitioning is an important immediate step in the scheduling of parallel programs on multiprocessor systems. Four different clustering algorithms are compared and some experimental results are presented.

Lo's paper, entited "Temporal Communication Graphs: Lamport's Process-Time Graphs Augmented for the Purpose of Mapping and Scheduling," develops the Temporal Communication Graph (TCG) into a new graph theoretic model of parallel computation which is then used for the scheduling of parallel programs on concurrent architectures. TCG integrates the two dominant models used for mapping and scheduling, i.e., static task graphs and DAGs. The paper introduces a language for describing TCGs, illustrates its use in scheduling, and presents a wide range of potential uses for the TCG in parallel programming environments.

The paper by Bultan and Aykanat, entitled "A New Mapping Heuristic Based on Mean Field Annealing," uses the recently proposed Mean Field Annealing (MFA) to develop a new mapping heuristic. The paper also gives an efficient implementation scheme which reduces the complexity of the proposed algorithm by asymptotic factors. This scheme exploits the inherent parallelism of the MFA to achieve an efficient parallel algorithm for the proposed MFA heuristic. Experimental results show that the proposed MFA algorithm is superior to the Kemighan–Lin heuristic and comparable to simulated annealing.

LOAD BALANCING

The paper "Analysis of the Generalized Dimension Exchange Method for Dynamic Load Balancing," written by Xu and Lau, generalizes the dimension exhange method by adding a parameter called the exchange parameter. The dimension exchange method is used for load balancing in distributed systems to control the splitting of the load between a pair of directly connected processors. Xu and Lau argue that splitting the workload into equal halves does not necessarily lead to an optimal result for certain structures, thus the need for the exchange parameter. They then show that even though equal splitting would yield optimal performance in hypercube structures, for chains, rings, and meshes optimal choices of the exchange parameter are related to the scale of these structures.

SYNCHRONIZED SCHEDULING

The paper "Gang Scheduling Performance Benefits for Fine-Grain Synchronization," written by Feitelson and Rudolph, addresses the important issue of scheduling in a multiprogrammed multiprocessor environment. One of the promising new ideas is *gang scheduling*, where a set of threads is scheduled to execute simultaneously on a set of processors. Without gang scheduling, threads have to block in order to synchronize, thus suffering the context switching overhead. Feitelson and Rudolph develop a model to evaluate the performance of different combinations of synchronization mechanisms and scheduling policies, and validate it by an implementation on the Makibilan multiprocessor. It is shown that gang scheduling is required for efficient fine grain synchonization on multiprogrammed multiprocessor systems.

The paper by Atallah, Lock, Marinescu, Siegel, and Casavant, entitled "Models and Algorithms for Co-Scheduling Compute Intensive Tasks on a Network of Workstations," considers the problem of using the idle cycles of a number of interconnected workstations for solving computationally intensive tasks. The classes of distributed applications examined require synchronization among the subtasks. Therefore, the authors use co-scheduling to ensure that subtasks start at the same time and execute at the same pace. The paper presents a model of the systems that allows the definition of an objective function to be maximized. Finally, a quadratic time and linear space algorithm is derived for computing the optimal coschedule.

ARCHITECTURE PARTITIONING

The paper entitled "Processor Allocation for Hypercubes," by Al-Bassam, El-Rewini, and Lewis, addresses the hypercube recognition problem, i.e., partitioning the

hypercube into a number of subcubes, each running different tasks. The authors develop an efficient subcube algorithm that recognizes all possible subcubes. The algorithm is based on the buddy tree, but exploits more subcubes at different levels. The number of recognized subcubes, for different subcube sizes, can be adjusted by restricting the search level down the buddy tree. In a multiprocessor system, each processor can execute the proposed algorithm on a different tree. For a given number of processors in a multiprocessor system, the paper presents a method for constructing the trees that maximizes the overall number of recognized subcubes. In addition, the paper introduces a "best fit" allocation method that reduces hypercube fragmentation.

Zhu's paper, entitled "First Fit and Best Fit Allocation Strategies for Mesh Computer Systems," proposes two new processor allocation strategies in partitionable mesh systems: First Fit (FF) and Best Fit(BF). Both strategies can allocate a submesh of exact size requested by the incoming task, while avoiding internal fragmentation of the mesh. As the names indicate, FF allocates an incoming task to the first fitting submesh it finds, while BF attempts to allocate a task to the smallest region of free processors. The simulation results demonstrate the superior performance of the proposed algorithms. In addition, it is shown that FF and BF strategies offer the same performance, although the former is much simpler to implement.

SPECIALIZED SCHEDULING

The paper by Prakash and Parker, entitled "SOS: Synthesis of Allocation-Specific Heterogeneous Multiprocessor Systems," describes a formal synthesis approach for the design of optimal application-specific heterogeneous multiprocessor systems. The proposed approach consists of the creation of a Mixed Integer–Linear Programming model and the solution of the model. An important aspect of the model is the set of relations that must be satisfied to ensure the proper ordering of various events in the task execution as well as to ensure completeness and correctness of the system. The experimental results demonstrate the usefulness of the model is designing application-specific multiprocessor systems.

The paper entitled "Scheduling Parallel I/O Operations in Multiple Bus Systems," written by Jain, Somalwar, Werth, and Browne, presents an algorithm for the optimal scheduling of batched I/O requests for a common class of shared memory multiprocessors. Parallel I/O requires scheduling multiple resources simultaneously, rather than a single resource serially. The proposed algorithm is essentially an optimal k-coloring of a bipartite graph with arbitrary edge weights, where the vertices represent processors and memories and the edges represent I/O transfers. The complexity of the algorithm is $O(n^3(\log n + \log k))$, where n is the number of vertices and k is the maximum edge weight, or length, of the longest I/O transfer.

Swami, Young, and Gupta's paper, entitled "Algorithms for Handling Skew in Parallel Task Scheduling," addresses the problem of scheduling a collection of independent tasks on multiple processors. The collection of tasks often constitutes a single parallel operation such as a parallel database join operation. If there is significant skew among different tasks, there can be a large imbalance of load assigned to the processors. This paper presents two new algorithms for handling skew in parallel task scheduling. The proposed algorithms are based on the "largest processing time first" concept and have better performance than competing algorithms.

ACKNOWLEDGMENTS

We express our appreciation to the authors of the submitted papers for their contributions to this special issue. Our deepest gratitude goes to the 83 reviewers, listed below, who spent their valuable time and effort to ensure a high-quality special issue. Without them the special issue would not have been possible. We thank Professor Kai Hwang for his technical and editorial support and for his encouragement in putting together this special issue.

Abu-Amara, H.
Texas A&M University

Agrawal, D. P.
North Carolina State University

Ahmad, I.
Hong Kong University

Ali, H. H.
University of Nebraska, Omaha

Armstrong, J.
Purdue University

Avritzer, A.
AT&T Laboratories

Banerjee, P.
University of Illinois

Bhattacharyya, S.
Kuck & Associates

Bhuyan, J. N.
University of Tuskegee, Alabama

Bhyuan, L. W.
Texas A&M University

Birk, Y.
Israel Institute of Technology

Bowles, J. B.
University of South Carolina

Burns, A.
University of York

Casavant, T. L.
University of Iowa

Chase, C.
Cornel University

Chen, H. B.
University of Texas, Arlington

Cherkassky, V.
University of Minnesota

Chuang, P.-J.
Rutgers University

Du, D. H. C.
University of Minnesota

Efe, K.
University of Southwestern Louisiana

El-Rewini, H.
University of Nebraska, Omaha

Eskicioglu, M. R.
University of Alberta

Finkel, D.
Worcester Polytechnic Institute

Gerasoulis, A.
Rutgers University

Ghafoor, A.
Purdue University

Gil, J.
University of British Columbia

Gill, H.
The MITRE Corporation

Gillies, D. W.
University of Illinois

Gosal, D.
University of Maryland

Greenwood, G. W.
University of Washington

Gupta, A.
IBM

Haddad, E.
Virginia Tech, Polytechnic Institute and State University

Hou, C. J.
University of Michigan

Jamieson, L. H.
Purdue University

Jereb, B.
Oregon State University

Kant, K.
Bell Communications Research

Karimi, B.
University of New Haven

Korfhage, W.
Polytechnic University of New York

Latifi, S.
University of Nevada, Las Vegas

LeBlanc, T. J.
University of Rochester

Lee, B.
Oregon State University

Lee, E. A.
University of California, Berkeley

Lewis, T. G.
Oregon State University

Lim, J.
Penn State University

Liu, H.-N.
University of California, San Diego

Lo, V. M.
University of Oregon

Marinescu, D.
Purdue University

Markatos, E. P.
University of Rochester

Mauney, J.
North Carolina State University

McCreary, C. L.
Auburn University

Menasce, D. S.
George Mason University

Mutka, M. W.
Michigan State University

Nau, R. W.
Carleton College

Pakzad, S.
Penn State University

Pande, S. S.
North Carolina State University

Parhi, K. K.
University of Minnesota

Peeray, M.
University of Illinois

Raghavendra, C. S.
Washington State University

Ramanujan, J.
Louisiana State Unversity

Ramkumar, B.
University of Illinois

Ranade, A. G.
University of California

Ravi, S. S.
SUNY at Albany

Reeves, A. P.
Cornell University

Sen, A.
Arizona State University

Sih, G. C.
University of California, Berkeley

Shin, K. G.
University of Michigan

Siegel, H. J.
Purdue University

Sih, G. C.
University of California, Berkeley

Singh, J. P.
Stanford University

Stoyenko, A. D.
New Jersey Institute of Technology

Swami, A.
IBM Almaden Research Center

Tan, J.
University of Houston

Tayyab, A.
University of Iowa

Tick, E.
University of Oregon

Tripathi, S. K.
University of Maryland

Wei, S.
Rutgers University

Wills, C. E.
Worcester Polytechnic Institute

Xu, J.
IBM Corporation

Yang T.
Rutgers University

Zhas, W.
University of Puskegee, Alabama

Zhu, Y.
North Dakota State University

Ziavras, S. G.
New Jersey Institute of Technology

Znati, T. F.
University of Pittsburgh

REFERENCES

1. Casvant, T. L., and Kuhl, J. G. A taxonomy of scheduling in general-purpose distributed computing systems. *IEEE Trans. Software Engrg.* **14,** 2 (1988), 141–154.
2. Eager, D. L., Lazowska, E. D., and Zahorjan, J. Adaptive load sharing in homogeneous distributed systems. *IEEE Trans. Software Engrg.* **12,** 5 (1986), 662–675.
3. Harget, A. J., and Johnson, I. D. A study of load balancing algorithms in distributed systems. *Proceedings of the IASTED Conference on Applied Informatics*. 1988, pp. 11–14.
4. Lee, B., Hurson, A. R., and Feng, T. A vertically layered allocation scheme for dataflow systems. *J. Parallel Distrib. Comput.* **11,** 3 (1991), 175–187.
5. Lin, F. C. H., and Keller, R. M. The gradient model load balancing method. *IEEE Trans. Software Engrg.* **13,** 1 (1987), 32–38.
6. Lo, V. M. Heuristic algorithms for task assignment in distributed systems. *Proceedings of the 4th International Conference on Distributed Computing Systems*. 1984, pp. 30–39.
7. Polychronopoulos, C. D., and Kuck, D. J. Guided self-scheduling: A practical scheduling scheme for parallel supercomputers. *IEEE Trans. Comput.* **C-36,** 12 (1987), 1425–1439.
8. Sarkar, V., and Hennessy, J. Compile-time partitioning and scheduling of parallel programs. *Proceedings of the Symposium on Compiler Construction*. 1986, pp. 17–26.
9. Shirazi, B., Wang, M., and Pathak, G. Analysis and evaluation of heuristic methods for static scheduling. *J. Parallel Distrib. Comput.* **10,** 3 (1990), 222–232.
10. Stone, H. S. Multiprocessor scheduling with the aid of network flow algorithms. *IEEE Trans. Software Engrg.* **13,** 1 (1977), 85–93.
11. Wang, M. F., Shirazi, B., Lee, B., and Hurson, A. R. Accurate communication cost estimation in static task scheduling. *Proceedings of the Hawaii International Conference on Systems Sciences*. 1991, pp. 10–16.

BEHROOZ SHIRAZI is currently an associate professor of computer science engineering at the University of Texas at Arlington. From 1985 to 1990, he was an assistant professor of computer science and engineering at Southern Methodist University. He received his Ph.D. degree in computer science from the University of Oklahoma in 1985. His research interests include computer architecture, parallel and distributed systems, scheduling and load balancing, heterogeneous systems, dataflow machines, and VLSI systolic arrays. He has published widely in the above-mentioned areas. Professor Shirazi is the principal founder of the IEEE Symposium on Parallel and Distributed Processing and has served on the program committee of the International Conference on Distributed Computing Systems. He is a member of the IEEE and ACM.

A. R. HURSON is a member of the computer engineering faculty at the Pennsylvania State University. His research for the past 12 years

has been directed toward the design and analysis of general as well as special purpose computer architectures. He has published over 100 technical papers in areas including computer architecture, parallel processing, database machines, dataflow architectures, and VLSI algorithms. Dr. Hurson has served as the Co-Guest Editor of a special issue of the IEEE Proceedings on Supercomputing Technology. He has been a member of the technical committees for various IEEE/ACM conferences and is the cofounder of the IEEE Symposium on Parallel and Distributed Processing. Professor Hurson is the coauthor of the IEEE Tutorial on Parallel Architectures for Database Systems. He is a member of the IEEE Computer Society Press Editorial Board, and a member of the IEEE Distinguished Visitor Program.