LEWIS GRANT
NAG3-677
IN-60-CR
287497
5P.

MULTIPROCESSOR ARCHITECTURE: SYNTHESIS AND EVALUATION

Hilda M. Standley
Department of Computer Science and Engineering
The University of Toledo
Toledo, Ohio 43606

INTRODUCTION

Multiprocessor computer architecture evaluation for structural analysis computations is the focus of the research effort described in this paper. Results obtained from this work are expected to lead to more efficient use of existing architectures and to suggest designs for new, application specific, architectures. The brief descriptions ~~that~~ given follow outline a number of related efforts directed toward this purpose.

The difficulty in analyzing an existing architecture or in designing a new computer architecture lies in the fact that the performance of a particular architecture, within the context of a given application, is determined by a number of factors. These include, but are not limited to, the efficiency of the computation algorithm, the programming language and support environment, the quality of the program written in the programming language, the multiplicity of the processing elements, the characteristics of the individual processing elements, the interconnection network connecting processors and non-local memories, and the shared memory organization covering the spectrum from no shared memory (all local memory) to one global access memory.

These performance determiners may be loosely classified as being software or hardware related. This distinction is not clear or even appropriate in many cases. For example, the instruction set of the processing elements may be considered a hardware characteristic, while at the same time it impacts the programming language implementation and support environment.

(NASA-CR-186618) MULTIPROCESSOR ARCHITECTURE: SYNTHESIS AND EVALUATION (Toledo Univ.) 5 p CSCL 09B

N90-25579

Unclas
G3/60 0287497

The goal of this work is to narrow the scope of observation to a more manageable set of independent variables. The effect of the choice of algorithm is ignored by assuming that the algorithm is specified as given. Effort directed toward the removal of the effect of the programming language and program resulted in the design of a high-level parallel programming language. Finally, two characteristics of the fundamental structure of the architecture--memory organization and interconnection network--are examined.

REMOVING THE LANGUAGE/PROGRAMMING EFFECT

The ideal situation in a program that is written for a multiprocessor system is to have the algorithm expressed with the maximum amount of parallelism evident. An automatic language processor may take the program and partition and allocate it to the granularity appropriate to the number and capability of the processing elements and also to the interprocessor communication overhead. Under this scheme, with any given algorithm, the result of the partitioning and allocation is the same for any program representing the algorithm.

The data flow computation scheme has gained recognition as a vehicle for expressing large amounts of operator-level potential parallelism and a few high-level data flow languages have been developed. With the exception of implementation on a few machines specifically designed to take advantage of that scale of parallelism, when implemented on more conventional machines, much of the advantage of the parallelism is lost by the necessity of collapsing the fine-grained representation of the program into a more suitable large-grained result.

The idea of large-grained data flow programming is not a new one [3]. A large-grained high-level data flow language offers an alternative to the low-level approach. The EASY-FLOW [5] high-level, large-grained, data flow language is well suited for the parallel adaptation of existing programs. EASY-FLOW calls for modules or subprograms written in a traditional language to be left unchanged. Only the interfaces between the modules will require rewriting using the data flow constructs.

STRUCTURAL PARAMETERS OF THE ARCHITECTURE: MEMORY ORGANIZATION

The problem of modeling memory interference in a multiprocessor shared memory organization is addressed [6]. A model is developed based upon the General Model for Memory Interference (GMI) [4], a queuing network mathematical model. This model for shared memory interference is enhanced to include a hierarchy of memories as the server associated with each request queue. In the model, n processors are interconnected with m memory modules via a crossbar switch which itself causes no delay in memory access. Within a memory module, each memory unit is connected via one bus to a number of memories in a hierarchical fashion. Each memory unit in the hierarchy has an associated size and access time.

The model generalizes the assumption of the independence of requests to different memories to include independence between requests to different levels within a single memory hierarchy. Performance of a given memory organization is defined by the expected number of busy memories and the system execution rate.

The resulting analytic model has been compared with a simulation model for various patterns of access to memory modules and to the memories within a hierarchy. Additionally, the times spent doing computation between memory accesses has varied. With data collected from more than sixty simulation runs, the results of the analytic model and simulation model correlate to the extent of 0.9950.

STRUCTURAL PARAMETERS OF THE ARCHITECTURE: INTERCONNECTION NETWORK

A model is developed to represent the relationship between the performance of a multiprocessor system and the topology of the interconnection network [2]. A topology is described as a k-tuple of values of independent variables, most of which correspond to graph theoretic properties. Consideration is given to such quantitative and qualitative properties as the number of nodes, average degree, diameter, radius, girth, connectivity, and minimum dominating set size. Dependent response variables offer measures of performance such as message completion rate, throughput, and channel utilization. Each response variable taken with the data from the k-tuples determines a (k+1)-space in which statistical evaluation techniques and/or classical optimization techniques may be applied.

Once optimization techniques have indicated an optimum, or multiple local optimums, each point of interest in the k-space must be interpreted as an actual interconnection network. It is important to note that any point under consideration specifies the characteristics of a network that theoretically corresponds to the accompanying levels of the independent variables used in the analysis. It remains to determine an actual network from a point. The resulting network may be one of the set determining the data points used in the "sample" to form the surface. It is perhaps more likely that the optimum point would correspond to levels of the characteristics that dictate a "new" or "hybrid" network. Any optimum point is not likely to be integer-valued, so it is necessary to use rounded values for the variables or to examine integer-valued points neighboring the optimum point.

Typically, an architectural design is chosen to meet the needs of a particular application in an ad hoc, compromising fashion. There are many criteria to consider, and there are necessary tradeoffs in making a choice of architecture. Simple regression-type models with their prediction ability, combined with techniques for predicting an optimal design for a given application provide the basis for this easy-to-use, powerful design tool. This modeling approach is flexible enough to allow the model to fit the problem--a performance metric is chosen to be optimized, along with any desired constraints on the architectural parameters. The potential applications are evident, both in multiprocessor design and in the development or enhancement of existing computing systems.

BIBLIOGRAPHY

[1] T. Agerwala and Arvind, "Data Flow Systems--Guest Editors' Introduction," Computer, Vol. 15, No. 2, Feb. 1982, pp. 10-13.

[2] D. S. Auxter and H. M. Standley, "Modeling and Synthesis of Multiprocessor Interconnection Networks," Technical Report, Department of Computer Science and Engineering, The University of Toledo, November 1987.

[3] R. Babb II, "Parallel Processing with Large-Grain Data Flow Techniques," Computer, Vol. 17, No. 7, July 1984, pp. 55-61.

[4] C. H. Hoogendoorn, "A General Model for Memory Interference in Multiprocessors," IEEE Transactions on Computers, Vol, C-26, pp. 998-1005, Oct. 1977.

[5] H. M. Standley, "A Very High Level Language for Large-Grained Data Flow," 1987 kACM Computer Science Conference (Proceedings), February 17-19, 1987, St. Louis, Mo.

[6] H. M. Standley and B. A. Taha, "Interference in Multiprocessor Systems with Multimemory Hierarchy," in preparation.