INFORMATION PROCESSING 86, H.-J. Kugler (ed.)
Elsevier Science Publishers B.V. (North-Holland)
© IFIP, 1986

PERFORMANCE EVALUATION IN PERSPECTIVE

Hisashi KOBAYASHI
School of Engineering and Applied Science, Princeton University
Princeton, New Jersey, U.S.A.

Over the past 20 years, computer performance evaluation methodology has evolved into a matured discipline in computer science/engineering. As the multiprogrammed and time-shared system came into wide use during 1960's and 1970's, the need to construct a total system model was recognized in order to understand quantitatively the complex interactions between different processes, and the effect of various resource allocation strategies. This gave an impetus to the performance evaluation community to seek powerful analytic models, thus the era of queueing network theory flourished since the mid 1970's, followed by the various approximate models and computational algorithms that were developed to numerically evaluate such performance measures as system throughput and response time. Now we, performance modellers, seem poised to seek major redirection. Where should we go from here?

Until recently a major thrust to speed up a computer has been to reduce its cycle time by employing gates with faster switching. This is exactly what mainframe computer and super computer manufacturers have accomplished in the past four generations of computer system technologies - vacuum tubes, transistors, integrated circuits, and VLSI circuits. With gate delays reduced to a nano-second level, the machine cycle time is now limited by propagation time required for electrical signals. In addition high-speed devices require expensive cooling systems. Therefore, high performance computers with single-processor architecture appear to be close to the performance maximum.

Designers of super computers thus look to machines with multiple processors arranged in parallel architectures. There are now three types of parallel architecture; (1) vector processor, (2) tightly coupled parallel processor, and (3) massively parallel machines. For a vector processor to achieve a reasonable performance gain it requires vectoriging at least 90 percent of the operations involved. Therefore vector processing is useful for special applications that are amenable to high vectorization. Similarly, for a tightly coupled parallel processor to achieve significant speed-up, the programs in execution must have highly parallel algorithms.

Now we begin to see several projects in universities and industries on massively parallel system with hundreds (or thousands) of processors communicating with hundred (or thousands) of memories. The RP3 project of IBM Research, the Ultra computer project of NYU are such examples. These emerging projects not only signal an enthusiasm for parallel architectures, but also present challenges and opportunities for the performance evaluation community. The following issues must be addressed in understanding the performance of parallel architectures: (1) How to determine the optimal degree of parallelism? (2) How to connect a large number of processors? (e.g. crossbar connections, nearest-neighbor-only connections, etc.) (3) How to attach processors to memory? (shared memory or non-shared memory). (4) How to decompose an application problem into parallel parts? (e.g. dataflow approach).

For instance, in a recent study reported by Pfister and Norton on the performance of RP3 (which is a MIMD shared-memory multiprocessor system) they discuss the need and difficulty for predicting complex interactions between processors, network and memory. We believe that time is ripe for the performance evaluation community to get involved in these exciting projects and develop methodologies and techniques to tackle the challenges that future parallel processing architectures will present.