A SIMULATION TOOL FOR PARALLEL ALGORITHMS DESIGN

Woei Lin
Department of Electrical and Computer Engineering
University of New Mexico
Albuquerque, NM 87131

One of the most difficult problems in the design and development of parallel algorithms is performance analysis. This is a considerably complex task, since many factors jointly determine the performance and variations of some factors may affect others. Software tools for instrumentation and analysis provide a means of solving this problem. The Parallel Algorithm Performance Analyzer (PAPA) is one such tool that provides programming and experimental environment for studying and analyzing the interaction among these factors: amount of inherent parallelism, grain size, communication structure of problems, overlapping between computation and communication, and speed of interprocess communication (IPC).

PAPA has three major design objectives:
(1) Small. The entire simulation tool can be stored in a single-sided, double-density diskette and can be easily brought into main memory by a single load. When it resides in the memory, it does not use up available memory space.
(2) Simple. The overall size and complexity of the tool must be easily understood and installed by the user.
(3) Adaptable. The tool must be of value in solving a wide class of problems, either directly or indirectly with minor modifications.

The design philosophy of PAPA stresses simplicity and ease of use. It runs on the ordinary IBM personal computer and its clones, employing monochrome monitor and keyboard input. But it provides all the facilities necessary to prototype parallel algorithms, to investigate their run-time behavior, and to report summary statistics from the simulation. Simplicity of PAPA simulation environment leads to minimized simulation overhead and memory requirements. It can efficiently simulate large scale parallel algorithms on the personal computer, despite of its inherently limited resources. These unique features make PAPA distinct from other software tools for simulating and evaluating parallel algorithms.

PAPA is based on the Communicating Sequential Process (CSP) model that has been widely considered as a natural way of presenting parallel algorithms. PAPA provides a parallel programming language, called POP, for algorithm implementation. Succinct but powerful POP control structures allow the user to easily describe computation and communication requirements of parallel activities. POP extends the index scheme, commonly used for structured data of array types, to the process level. An aggregate of parallel processes can be efficiently declared through a single indexed program instance.

An event-driven simulation with IPC tracing capability allows the user to observe and step through IPC transactions in time-ordered sequence. We choose behavioral tracing rather than detailed cycle-by-cycle tracing. In the simulation, every instruction is executed. However, only IPC events are traced. Synchronization and scheduling are two major issues in the design of simulators for parallel algorithms. The former refers to coordinating paces of parallel activities to properly accomplish IPC transactions. The latter refers to interleaving executions of parallel activities on a uniprocessor system to maintain the partial-ordering property. In PAPA simulation environment, scheduling is done by a monitor; it essentially decides the execution order. However, processes handle synchronization by themselves without intervention of the monitor. The separation of synchronization and scheduling results in a significant reduction in simulation time. PAPA has proven its value in two ways: it has been used as a research tool for demonstrating and evaluating new parallel algorithm designs, and it has been used as an instructional tool for developing and studying parallel algorithms in a graduate level course on parallel processing.