Conf-950601--3

SAND95-0140C

Distributed Multitasking ITS with PVM

Wesley C. Fan
Radiation and Electromagnetic Analysis Department
and
John A. Halbleib, Sr.
Simulation Technology Research Department
Sandia National Laboratories
Albuquerque, NM. 87185

* Prepared for the American Nuclear Society Annual Meeting
in Philadelphia, June 25-29, 1995.

** This work was supported by the U. S. Department of Energy
under Contract No. DE-AC04-94AL8500.

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, make any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

Distributed Multitasking ITS with PVM

Wesley C. Fan
Radiation and Electromagnetic Analysis Department
and
John A. Halbleib, Sr.
Simulation Technology Research Department
Sandia National Laboratories
Albuquerque, NM. 87185

INTRODUCTION

Advances of computer hardware and communication software have made it possible to perform parallel-processing computing on a collection of desktop workstations. For many applications, multitasking on a cluster of high-performance workstations has achieved performance comparable or better than that on a traditional supercomputer. From the point of view of cost-effectiveness, it also allows users to exploit available but unused computational resources, and thus achieve a higher performance-to-cost ratio.

Monte Carlo calculations are inherently parallelizable because the individual particle trajectories can be generated independently with minimum need for interprocessor communication. Furthermore, the number of particle histories that can be generated in a given amount of wall-clock time is nearly proportional to the number of processors in the cluster. This is an important fact because the inherent statistical uncertainty in any Monte Carlo result decreases as the number of histories increases. For these reasons, researchers have expended considerable effort to take advantage of different parallel architectures for a variety of Monte Carlo radiation transport codes, often with excellent results.

The initial interest in this work was sparked by the multitasking capability of MCNP[1] on a cluster of workstations using the Parallel Virtual Machine (PVM) software. On a 16-machine IBM RS/6000 cluster, it has been demonstrated that MCNP runs ten times as fast as on a single-processor CRAY YMP. In this paper, we summarize the implementation of a similar multitasking capability for the coupled electron/photon transport code system, the Integrated TIGER Series (ITS), and the evaluation of two load balancing schemes for homogeneous and heterogeneous networks.

MULTITASKING ITS WITH PVM

The ITS system[2] provides a state-of-the-art Monte Carlo solution of linear, time-integrated, coupled electron/photon radiation transport problems with or without the presence of external electric and magnetic fields. In ITS, the particle histories are divided into "batches" of equal size and the evaluation of the estimated quantities are performed using batch-averaged sample statistics. Since the batchwise evaluation can be performed independently, it provides a natural partition for multitasking.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

The multitasking version of ITS is based upon a message-passing model in conjunction with a master/slave paradigm. The master process performs the input/output functions and starts up the slave processes, while the slaves perform the majority of the computational work, i.e., generating particle trajectories and scoring. The problem-dependent data (such as the geometry and cross-section data) and the tallied results are transferred between master and slaves, but no data is shared between the slaves. All the communication tasks, including the data transfer and process control, are handled by the software system PVM.[3]

In a multiuser environment, appropriate load balancing can further enhance performance of a parallel program. The current version of ITS/PVM provides two load-balancing schemes, namely, the static and dynamic methods. The static method is simple and easy to implement. In this method the required tasks (or batches) are divided up and assigned to the available machines in the configured PVM. The number of batches can vary from machine to machine to account for different computation powers for different machines. These assignments are set at the start and will not be adjusted to the actual loading and performance. As one may expect, this scheme can be quite effective on a lightly loaded network (either homogeneous or heterogeneous).

Dynamic load-balancing in ITS/PVM is accomplished by the classic "pool-of-tasks" paradigm. Initially, each slave process is given a batch. As a slave process finishes its batch it will receive another one. With this scheme all the slave processes are kept busy as long as there are batches remaining in the pool. The work load for each machine is adjusted according to the "realistic" computational performance which can be changing dynamically as other users share the resources. In this case, one may wish to divide up the problem into small batches (small number of particles per batch) which may be easier to balance across the available machines than the large batches.

SPEEDUP AND PERFORMANCE EVALUATIONS

The goal of multitasking is to make the program run faster (shorter wall-clock time) than it would in the corresponding serial run. A speedup ratio is often used to evaluate the performance of a multitasking program. On a dedicated system, the speedup ratio can be calculated using the following manner:

$$S_N = \frac{T_S}{T_N} = \frac{1}{1 - F_P + \frac{F_P}{N}} ,$$

where S_N is the speedup ratio if N processors are used in the calculation, T_S is the elapsed wall-clock time for a single processor, T_N is the wall-clock time for N processors, and F_P is the fraction of program that can be run in parallel (sometimes called the parallel efficiency). The second part of this equation is known as Amdahl's law from which one can estimate the parallel efficiency based on a set of measured speedup ratios.

Table 1 summaries the measured speedup ratios for seven test problems on a cluster of SUN workstations. These test problems include the three standard codes, two P-codes, and two M-codes of the ITS system, and utilize many tally and biasing options of the system. Sufficient particle histories were required so that the input/output times were negligible in comparison to the overall CPU times. It is observed that the speedup ratio increase almost linearly with the number of processors. The parallel efficiency approaches 99%, except for the ACCEPT-M code, where it is around 93%.

Further studies indicated that the relative poor performance of the ACCEPT-M code was caused by an anomalous batch which consumed ~ 50% more CPU time than the other batches. It is believed that one or more electrons entered a vacuum region with a uniform magnetic field with velocities almost perpendicular to the field so that they drifted very slowly through this region. Consequently, extra computing time was needed to calculate these orbits, thus prolonging the CPU time for that batch.

CONCLUSIONS

Using the PVM communication software, we have implemented a distributed-multitasking capability in the ITS code system. An update to ITS Version 3.0 was developed and tested on a cluster of workstations. For selected problems, the multitasking version of ITS performs very well with estimated efficiencies approaching the theoretical limit. This multitasking capability will undoubtedly become a standard feature in the future releases.

REFERENCES

1. G. W. McKINNEY and J. T. WEST, "Multiprocessing MCNP on an IBM RS/6000 Cluster," *Trans. Am. Nucl. Soc.*, Vol. 56, pp. 212-214, 1993.

2. J. A. HALBLEIB et al., "ITS: The Integrated TIGER Series of Coupled Electron/Photon Monte Carlo Transport Codes - Version 3.0," *IEEE Trans. Nucl. Sci.*, Vol. 39, No. 4, pp.1025-1030, 1992.

3. G. A. GEIST et al., "PVM 3 User's Guide and Reference Manual," Technical Report ORNL/TM-12187, Oak Ridge National Laboratory, 1994.

Table 1. **Measured Speedup Ratios* for Various ITS/PVM Applications**

Code	Number of Processors				
	2	4	8	12	16
TIGER	1.99	3.81	7.39	10.88	14.19
CYLTRAN	1.97	3.92	7.32	10.93	14.31
ACCEPT	1.99	3.93	7.42	10.64	14.2
TIGER-P	1.97	3.93	7.77	11.5	14.39
ACCEPT-P	1.96	3.88	7.64	11.35	14.03
CYLTRAN-M	1.96	3.87	7.42	11.06	14.56
ACCEPT-M	1.80	3.49	5.83	7.75	10.78
Amdahl's Law with 99% Efficiency	1.98	3.88	7.48	10.81	13.91

* Speedup relative to a single SUN4/75 workstation.