LA-UR 95-2065

CONF-951006--18

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405 ENG-36

TITLE MCNP LOAD BALANCING AND FAULT TOLERANCE WITH PVM

AUTHOR(S) Gregg W. McKinney

SUBMITTED TO ANS Winter Meeting, 10/29 - 11/2/95, San Francisco, CA

DISCLAIMER

(11

Los Alamos National Laboratory
Los Alamos, New Mexico 87545

MASTER

MCNP LOAD BALANCING AND FAULT TOLERANCE WITH PVM

Gregg W. McKinney
Los Alamos National Laboratory
XTM, MS B226
Los Alamos, NM 87545
(505)665-8367

I. INTRODUCTION

Version 4A of the Monte Carlo neutron, photon, and electron transport code MCNP[1], developed by LANL (Los Alamos National Laboratory), supports distributed-memory multiprocessing through the software package PVM[2] (Parallel Virtual Machine, version 3.1.4). Using PVM for interprocessor communication, MCNP can simultaneously execute a single problem on a cluster of UNIX-based workstations. This capability provided system efficiencies that exceeded 80% on dedicated workstation clusters;[3,4] however, on heterogeneous or multiuser systems, the performance was limited by the slowest processor (i.e., equal work was assigned to each processor). The next public release of MCNP will provide multiprocessing enhancements that include load balancing and fault tolerance which are shown to dramatically increase multiuser system efficiency and reliability.

II. LOAD BALANCING AND FAULT TOLERANCE

Any attempt at load balancing effectively trades a reduction in efficiency on a dedicated system for an increase in efficiency on heterogeneous or multiuser systems. Three approaches to load balancing were investigated:

(1) Polling machine loads - while this approach provides load information with

minimal communication, it suffers from a lack of a universal means of obtaining machine loads and a variable polling frequency that is likely a strong function of the system load.

(2) Measuring machine loads - while this approach overcomes the first obstacle of the previous approach, it suffers from an increase in bookkeeping and a measurement frequency that again is a strong function of the system load. Changes in system load between measurements could have a dramatic effect on system efficiency.

(3) Microtasking - with a slight increase in communication, this approach achieves inherent load balancing that accounts for real-time changes in system load. Microtasking involves dividing a block of work into small pieces and assigning these pieces on an availability basis, where machines with smaller loads complete more pieces of work. As reported below, the optimal degree of microtasking is not a strong function of the system load.

The microtask approach to load balancing was implemented into MCNP with a dynamic control algorithm for the degree of microtasking (i.e., number of microtasks created per processor) that is a function of the system load. Parameters for this algorithm were determined by extensive testing and are not a strong function of the system load.

Treating machine failure as a rare event makes the approach to fault tolerance secondary to that of load balancing. With the implementation of microtasking for load balance, two approaches to fault tolerance became evident:

(1) Rerun all microtasks of the failed host - while this approach minimizes ineffi-

ciency, the coding required to ensure sequential tracking (e.g., resetting of random number seeds, repositioning of input files, etc.) was excessive.

(2) Restart from the previous rendezvous - consistent with the rare event assumption, this approach minimizes coding while increasing inefficiency. This decrease in efficiency should be negligible if indeed failures are rare.

The latter approach to fault tolerance was implemented. Failure of the master task results in the loss of work subsequent to the previous restart dump. Failure of all subtasks results in a sequential completion of the problem. With this enhancement, MCNP multiprocessing reliability is likely in the 90+ percentile.

III. SYSTEM EFFICIENCY

As mentioned above, any attempt to increase system efficiency for heterogeneous or multiuser systems will decrease efficiency on dedicated homogeneous systems. The goal is to minimize any loss while maximizing the gain. Table 1 lists the measured efficiency loss (relative to MCNP 4A) of these enhancements for a dedicated Sun IPX cluster and four test problems. These test problems were taken from the MCNP 4A test set and include a neutron, coupled neutron/photon, coupled photon/electron, and a criticality problem. The total number of histories was increased to require about 7 hours of sequential execution time. Note in Table 1 that most of the loss in efficiency is due to the dedication of the master task to microtask assignment and fault detection (i.e., a loss of 50% for 2 processors, 25% for 4, etc.). For moderate-sized clusters, the total loss in efficiency is shown to be less than 20-30%.

Table 1:

	Percent Change In Efficiency*			
CPUs	INP05	INP10	INP23	INP18
2	-50	-47	-54	-45
4	-26	-22	-32	-21
8	-14	-9	-20	-14
16	-5	-4	-14	Sys. Fault
* For a dedicated Sun IPX cluster.				

Table 2 gives the measured efficiency gain of these enhancements for a homogeneous multiuser IBM RS/6000 590 cluster. The multiple entries in this table indicate results from multiple trials. Note, the gains achieved from load balancing appear to just offset the loss of the master task processor for clusters consisting of four CPUs. For moderate-sized clusters, the gain in efficiency can exceed 20%.

Table 2:

	Percent Change In Efficiency*			
CPUs	INP05	INP10	INP23	INP18
4	-9,-1,20	6,-4,15	3,-10,-1	2,-7,0
8	4,25,34	2,32,26	1,21,21	17, 3,13
* For a multi-user IBM RS/6000 590 cluster.				

Table 3 shows efficiency gains for a heterogeneous Sun/IBM cluster (4 CPUs: Sparc 2, IPX, Sparc 10, RS/6000; 8 CPUs: Sparc 2, Sparc 5, 2 IPXs, 2 Sparc 10s, 2 RS/6000s; 16 CPUs: 2 Sparc 2s, Sparc 5, 5 IPXs, 4 Sparc 10s, 4 RS/6000s). For moderate-sized heterogeneous clusters, the gain in efficiency is shown to exceed 100%.

Table 3:

	Percent Change In Efficiency*			
CPUs	INP05	INP10	INP23	INP18
4	-17,-17	-12,-11	-13,-9	-14,-14
8	115,123	109,120	136,137	75
16	92,106	80,90	119,104	

* For a heterogeneous Sun/IBM cluster.

REFERENCES

1. "MCNP - A General Monte Carlo N-Particle Transport Code, Version 4A," J. F. Briesmeister, Editor, LA-12625-M, Los Alamos National Laboratory (1993).

2. A. Geist et al., "PVM 3 User's Guide and Reference Manual," ORNL/TM-12187, Oak Ridge National Laboratory (1993).

3. G. W. McKinney, "Parallel Processing Monte Carlo Radiation Transport Codes," Proceedings of the 8th ICRS, Arlington, Texas, April 24-28 (1994).

4. G. W. McKinney et al., "Multiprocessing MCNP on an IBM RS/6000 Cluster," Trans. Am. Nucl. Soc., 68, 212 (1993).