LA-UR -93-1338

Conf-9304151--1

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: DISTRIBUTED PROCESSOR MONTE CARLO: MCNP RESULTS ON A 16-NODE IBM CLUSTER

AUTHOR(S): G. W. McKinney

SUBMITTED TO: Centre d'Etudes, Saclay, France, April 27-29, 1993.

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, makes 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.

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Los Alamos National Laboratory
Los Alamos, New Mexico 87545

FORM NO 836 R4
ST NO 2629 5/81

DISTRIBUTED PROCESSOR MONTE CARLO: MCNP RESULTS ON A 16-NODE IBM CLUSTER

Gregg W. McKinney
Los Alamos National Laboratory
Los Alamos, NM 87545

I. INTRODUCTION

The advent of high-performance computer systems has brought to maturity programming concepts like vectorization, multiprocessing, and multitasking. Although there are many schools of thought as to the most significant factor in obtaining order-of-magnitude increases in performance, such speedup can only be achieved by integrating the computer system and application code.

Vectorization leads to faster manipulation of arrays by overlapping instruction CPU cycles. Discrete ordinates codes, which require the solving of large matrices, have proved to be major benefactors of vectorization. Monte Carlo transport, on the other hand, typically contains numerous logic statements and requires extensive redevelopment to benefit from vectorization.

Multiprocessing and multitasking provide additional CPU cycles via multiple processors. Such systems are generally designed with either common memory access (multitasking) or distributed memory access. In both cases, theoretical speedup, as a function of the number of processors (P) and the fraction of task time that multiprocesses (f), can be formulated using Amdahl's Law

$$S(f,P) = 1/(1 - f + f/P)$$

However, for most applications this theoretical limit cannot be achieved, due to additional terms (e.g., multitasking overhead, memory overlap, etc.) not included in Amdahl's Law.[1] Monte Carlo transport is a natural candidate for multiprocessing, since the particle tracks are generally independent and the precision of the result increases as the square root of the number of particles tracked.

II. MCNP MULTIPROCESSING WITH PVM

The Monte Carlo neutron, photon, and electron transport code MCNP.[2] developed by LANL (Los Alamos National Laboratory, X-6 Group), has an extensive list of attractive features, including continuous energy cross sections, generalized 3-D geometry, time dependent transport, and comprehensive source and tally capabilities. It is widely used for nuclear criticality analysis, nuclear reactor shielding, oil well logging, and medical dosimetry calculations (to mention a few) in many research laboratories within the United States, Canada, Europe, and Japan, in addition to over 100 universities and private companies throughout the world. MCNP geometry is described by the union and intersection of general quadratic surfaces and includes such features as repeated structures and nested lattices. The source capability allows for particles to be started from arbitrary user defined distributions with general hierarchical dependencies. Numerous variance reduction techniques provide the user with powerful statistical tools to optimize complex radiation transport problems. Several different tally options allow the user to obtain the flux or current (or virtually any flux weighted quantity) across a surface, averaged over a volume, or at a point.

Since the inception of multitasking software, MCNP developers have made it a high priority to support multitasking on a variety of common memory systems. With the widespread use of high performance workstations, interest in multiprocessing MCNP on distributed memory systems has increased. Such systems, connected by high speed communication networks, make it possible for codes like MCNP to achieve order of magnitude higher performance over current common memory systems. Ideally, MCNP communication on such a system could be limited to a few seconds at the beginning of a calculation (to initialize the tasks) and at the end (to collect the results). However, software QA standards imposed on MCNP require that multiprocessing versions track exactly with that of sequential versions, increasing communication between tasks to sum results at established rendezvous points.

The software communication package currently supported within MCNP is Parallel Virtual Machine, PVM[4] (version 2.4.2), developed by ORNL (Oak Ridge National Laboratory). This package enables concurrent computing on loosely coupled networks of processing elements. PVM may be implemented on a hardware base consisting of different machine architectures, including single CPU systems, vector machines, and multiprocessors (see Table I). These computing elements may be interconnected by one or more networks, which may themselves be different (e.g., Ethernet, Internet, and fiber optic networks). These computing elements are accessed by applications via a library of standard interface routines. These routines allow the initiation and termination of processes across the network as well as communication and synchronization between processes.[3]

III. MCNP SPEEDUP ON AN IBM RS/6000 CLUSTER

The PVM version of MCNP was evaluated on a 16 processor IBM RISC (Reduced Instruction Set Computer) System/6000 Model 560 workstation cluster at LANL. Determining true speedup in a multi-user environment can be difficult due to a lack of appropriate timing routines. On a dedicated system, simple wall-clock time can be used to calculate speedup. The IBM RS/6000 operating system (AIX) provides user, system, and wall clock timing routines; however, the user and system routines do not include "sleep" time. Thus, CPU time "wasted" during communication lag (e.g., the PVM master task waiting for replies from spawned subtasks) is not included in these timing routines. (The term "wasted" here is used in the strict sense—on a multi-user system, these CPU cycles will most likely be used by other applications). The wall clock time on a nondedicated system is a function of the system load and cannot be used to estimate speedup.

In previous versions of MCNP, results were summed every 1000 particles to update certain tally tables. (e.g., Tally Fluctuation Chart, detector/XTRAN Russian roulette criteria). The value of 1000 changes with the number of particles tracked and was chosen somewhat arbitrarily. However, in the current version of MCNP this parameter can be altered by the user and determines the amount of inter-task communication. Thus, this parameter has a significant impact on speedup. The sleep time of the master task, associated with this communication, is a result of the random nature of Monte Carlo tracking (i.e., processors tracking the same number of particles will not finish at the same time). Wall clock timing on a virtually dedicated system indicates that this sleep time can be substantial, especially if communication is required every 1000 particles.

Speedup estimates were made using the following equation

$$S(P) = T_s/T_m(P)$$

where T_s is the CPU time for a single processor to complete execution and $T_m(P)$ is the CPU time for the PVM master task to complete execution with P subtasks.

In MCNP, the master task initializes the problem, spawns the subtasks, collects the results, and writes the output files. Communication sleep time was accounted for in $T_m(P)$ by using the wall-clock time (AIX TIME utility) on a virtually dedicated system. Multiple off-hour runs and system-load monitoring were used to ensure that the system was dedicated. While this approach provided estimates of speedup to within about ±2%, it most likely underestimates the speedup since system applications also compete for CPU cycles.

Table II presents MCNP speedup for the 16 processor IBM RS/6000 cluster (based on MCNP version 4xe with PVM version 2.4.1). Ten test problems, representing a wide range in geometry and complexity, were chosen from the MCNP 25 problem test set for inclusion in this analysis. Execution times were made sufficient (~ 120 minutes) to eliminate any effect of the sequential problem initialization time (1-20 seconds). Figure 1 is a plot of the average speedup (of all ten problems) for saturated and standard communication (saturated being every 1000 particles and standard being the current MCNP default of 10 rendezvous during execution). The curve for saturated communication does indeed show that above 8 processors communication saturates the performance. On the other hand, the average speedup for standard communication increases linearly with the number of processors. Longer execution times result in a curve that approaches that of the theoretical limit.

IV. CONCLUSIONS

The PVM version of MCNP on a 16 processor IBM RS/6000 cluster produced speedups that approach the number of processors. Such speedup, in units of a single processor Cray Y MP (see figure 1, right ordinate), leaves little doubt that MCNP performance on a workstation cluster can greatly surpass that of most supercomputers. Reliability and speed of the communication network are critical factors in exploiting such distributed memory systems. Of the links available on the LANL IBM cluster, the Ethernet and FDDI links proved most reliable (95%).

REFERENCES

1. G. W. McKinney, "Multiprocessing Monte Carlo Codes on Distributed and Common Memory Computer Systems," Ph.D. Thesis, University of Washington, May 1987.

2. J. F. Briesmeister, Editor, "MCNP - A General Monte Carlo Code for Neutron and Photon Transport, Version 3A," LA 7396 M, Rev. 2, 1986, Los Alamos National Laboratory.

3. "A Users' Guide to Parallel Virtual Machine," Adam Beguelin et al., Oak Ridge National Laboratory, ORNL/TM- 11826, July 1991.

TABLE I

COMPUTER SYSTEMS SUPPORTED BY PVM*

Architecture Mnemonic	Description	
AFXS	Alliant FX/S	
ALPHA	DEC Alpha	(OSF-1)
BAL	Sequent Balance	(DYNIX)
BFLY	BBN Butterfly TC2000	
BSD386	80386/486 Unix box	(BSDI)
CM2	Thinking Machines CM2	
CM5	Thinking Machines CM5	
CNVX	Convex C-series	
DGAV	Data General Aviion	
CRAY	C-90, YMP, Cray-2	(UNICOS)
CRAYSMP	Cray S-MP	
HP300	HP-9000 model 300	(HPUX)
HPPA	HP-9000 PA-RISC	
I860	Intel iPSC/860	
IPSC2	Intel iPSC/2 386 host	(SysV)
KSR1	Kendall Square KSR-1	(OSF-1)
NEXT	NeXT	
PGON	Intel Paragon	
PMAX	DECstation 3100, 5100	(Ultrix)
RS6K	IBM/RS6000	(AIX)
RT	IBM RT	
SGI	Silicon Graphics IRIS	
SUN3	Sun3	(SunOS)
SUN4	Sun 4, SPARCstation	
SYMM	Sequent Symmetry	
TITN	Stardent Titan	
UVAX	DEC MicroVAX	

*Listed in a preliminary release of PVM version 3.0.

TABLE II

MCNP SPEEDUP ON THE LANL 16 PROCESSOR IBM RS/6000 CLUSTER

Number of Processors	MCNP Test Problem										
	3	5	10	11	12	14	15	16	20	23	Average
	SATURATED COMMUNICATION										
2	2.0	1.9	1.9	1.9	1.8	1.9	1.9	1.9	2.0	1.9	1.9
4	3.7	3.5	3.5	3.3	2.9	3.6	3.7	3.5	3.7	3.6	3.5
8	6.5	6.1	5.7	5.3	4.0	6.6	6.8	6.3	6.6	6.1	6.0
12	9.0	7.8	7.3	6.7	4.3	8.7	9.0	8.2	9.1	7.8	7.8
16	10.2	9.5	7.7	7.0	3.6	7.6	9.3	8.4	9.9	7.5	8.1
	STANDARD COMMUNICATION										
2	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0
4	3.9	3.9	3.8	3.8	3.7	3.9	3.9	3.9	3.9	3.9	3.9
8	7.7	7.5	7.2	7.2	6.5	7.7	7.7	7.6	7.5	7.4	7.4
12	11.2	10.6	10.1	10.1	8.9	11.2	11.3	11.0	10.9	10.7	10.6
16	14.3	13.4	12.6	12.4	10.8	14.3	14.5	13.8	13.6	13.5	13.3

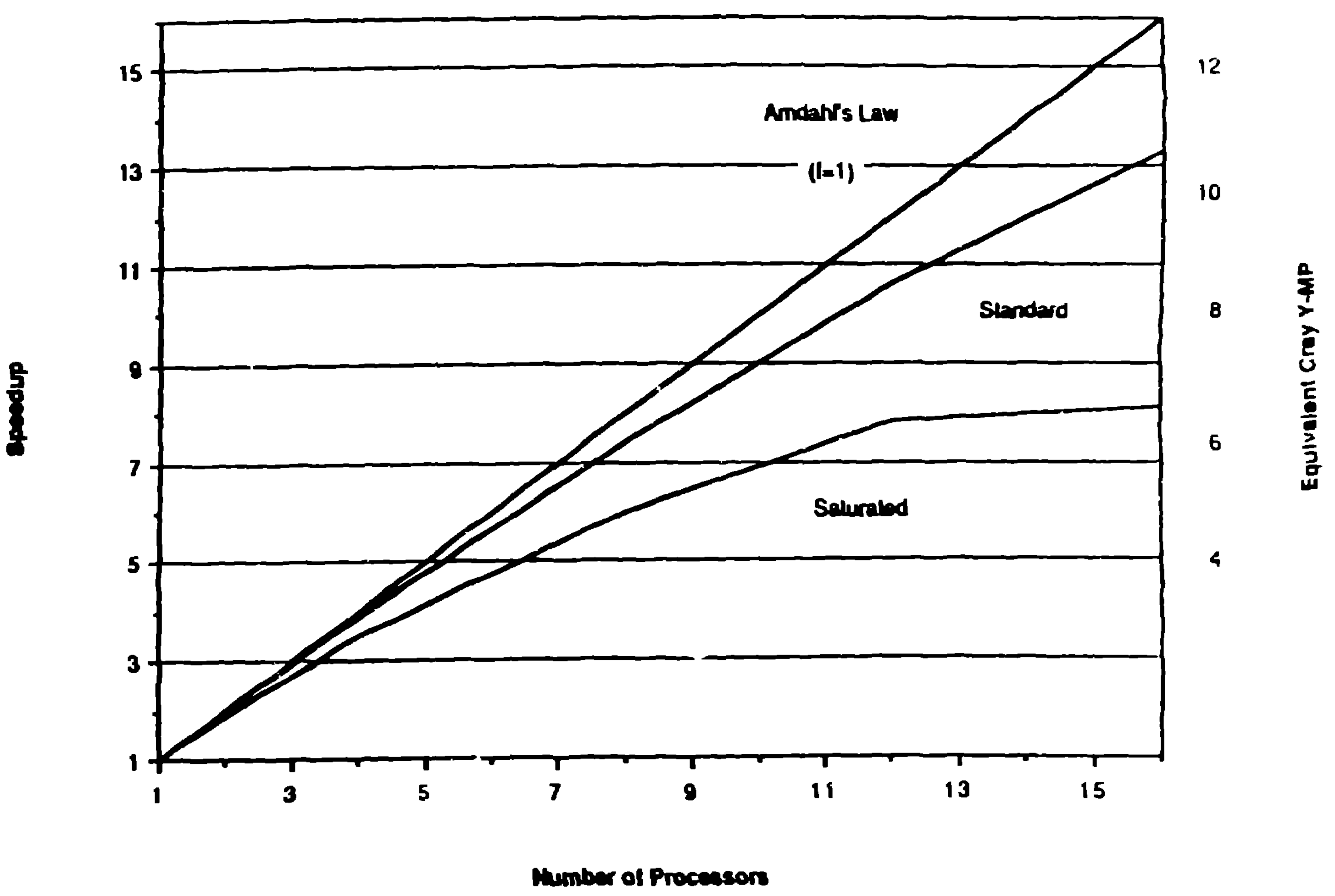

Fig. 1. MCNP Speedup on the LANL 16 Processor IBM RS/6000 Cluster.