Finite-Difference Time-Domain (FDTD) Analysis Using Distributed Computing

V. Varadarajan and Raj Mittra, *Fellow, IEEE*

Abstract— **This paper describes an implementation of the Finite-Difference Time-Domain FDTD calculations using PVM [Parallel Virtual Machine] 3.2 and a cluster of workstations for a test problem involving a three-dimensional rectangular cavity. It is found that the speedup factors achieved in FDTD calculations can approach the maximum values for large problem sizes if the computation-to-communication ratios are maintained at values significantly greater than unity. It is also found that in order to achieve linear speedups, the problem sizes should be increased with the number of processors while adequate computation-to-communication ratios are maintained in the individual processors. The results demonstrate the potential of parallel distributed computing for FDTD calculations in Electromagnetics.**

I. Introduction

A POPULAR software for distributed scientific computing is PVM 3.2 [1]. The message-passing protocol featured in PVM 3.2, its free availability, and the portability of the application codes written in PVM make it specifically suited for effective utilization of workstation clusters. This has opened up the possibility of solving the Maxwell's equations using the Finite-Difference Time-Domain [FDTD] scheme on a large mesh by using the collective power of a group of workstations.

The finite-differencing scheme of the Maxwell's equations using Yee-cells involving interlaced edge-based discretization of the field quantities using rectangular cells is quite well known, and the details can be found in [2]–[4]. The results from distributed computing of 3-D/FDTD calculations for a homogeneous and isotropic rectangular cavity problem are discussed in this paper.

II. Distributed Implementation

The numerical results reported in this paper have been obtained by using an HP-735 workstation cluster running under a version of UNIX System V. The PVM jobs were run in dedicated parallel queues using the DQS (Distributed Queuing System).

The FDTD algorithm exhibits nearest-neighbor communication pattern. The HP-735 cluster used in this study is configured as a linear array connected by an Ethernet LAN segment. Hence, the total computational burden is split into N_P equal subtasks by partitioning the problem domain into N_P equal-sized subdomains in the $z-$ direction. The communication phases were regulated in order to minimize network contention. The task idle times were also minimized by arranging the communication phases suitably. The number of Yee-cell edges in the $x-$, $y-$, and $z-$directions are, respectively, N_x, N_y, and N_z. The $H_x - H_y$ and $E_x - E_y$ variables that are to be communicated between neighboring tasks are packed into arrays of size $2(N_x+1)(N_y+1)$ and given to the local PVM daemons with appropriate message tags. These data are packed and routed by the local PVM daemons to the neighboring PVM daemons, and the data are finally received by appropriate tasks. The message-passing approach is implemented in PVM using the UDP/TCP/IP protocols over the Ethernet. The required program was written in master-slave model.

Manuscript received January 6, 1994.

The authors are with the Electromagnetic Communication Laboratory, University of Illinois at Urbana, Champaign, IL 61801 USA.

IEEE Log Number 9401067.

III. Performance Factors

A detailed discussion of the distributed processing issues in tightly-coupled computer architectures can be found in the literature: [5], [6]. The performance factors for distributed parallel processing using workstation clusters are not yet clearly understood. However, some important performance factors can be defined as follows: The computation-to-communication ratio (CC ratio) for the FDTD algorithm in the presence of saturated network traffic in a *serial bus topology* can be approximately given by

$$\text{CCratio} = \frac{C_1 mnpT_{cal}/N_P}{C_2 mnT_{com}(N_P - 1)} \tag{1}$$

The factor $N_P - 1$ in (1) reflects the fact that $N_P - 1$ pairwise communications exist among N_P processors in the cluster. In (1), mnp is the number of Yee-cells, N_P is the number of processors, $C_1(\approx 42 - 60)$ is the number of floating point operations per cell, T_{cal} is the average time for a floating point operation in FDTD calculations, and T_{com} is the average time for communicating a byte between two processors at peak throughput, mn is the approximate size of array for each variable to be communicated to the neighbors, and C_2 has a maximum value of 8 or 16. When the communication time is less than the computation time, the actual value of C_2 would be less. Here m, n, and p are nearly equal to N_x, N_y, and N_z. In the definition of CC ratio (1), small overheads from task switching, synchronization, initial startup overheads, and I/O overheads are not included. Also, the speedup S achieved for large problems can be approximately given by

$$S = \frac{C_1 mnpT_{cal}}{C_1 mnpT_{cal}/N_P + C_2 mnT_{com}(N_P - 1)} = \frac{N_P}{1 + (\text{CCratio})^{-1}}. \tag{2}$$

In practice, an accurate measure of the communication time

TABLE I
PERFORMANCE OF THE PVM CODE FOR SEVERAL 3-D FDTD PROBLEMS

Problem Size	Procs	Speedup	CC ratio	PVM Time	Single-task Time
75x75x150	4	2.77	2.25	3.8 hrs	10.5 hrs
75x75x300	6	3.90	1.85	5.4 hrs	21.1 hrs
75x75x150x8	8	7.06	7.50	12.0 hrs	84.5 hrs
100x100x80x8	8	6.24	3.50	12.8 hrs	80.1 hrs
125x125x50x8	8	4.88	1.56	16.5 hrs	80.5 hrs
140x140x40x8	8	4.07	0.95	19.2 hrs	78.2 hrs

can be obtained from the difference between the real time and user time of the time-stepping loop that does the majority of the computations in a FDTD code.

IV. RESULTS

Several parallel performance results pertaining to different problem sizes from the discretization of the cavity resonance problem using FDTD approach are given in Table I. Two fixed problem sizes 75x75x150 and 75x75x300, and four fixed per-processor problem sizes 75x75x150, 100x100x80, 120x125x50, and 140x140x40 are chosen to contrast the results under different assumptions. In the last four cases, the total problem size increases linearly with the number of processors. The speedups achieved for example problem sizes 1, 2, and 4 in Table I are also illustrated in Fig. 1 as a function of the number of processors.

The calculations were executed at about 14 MFLOP's in each HP-735 processor. The MFLOP estimate is based on the user time. In Table I, the problem size, the number of HP-735 workstations that executed the PVM code, the speedup achieved, the average CC ratio, turnaround time in hours for 16500 time steps using PVM, and the estimated time for single-task calculations using an HP-735 workstation are exhibited in columns 1 through 6.

From the first two rows of results in Table I and from Fig. 1, it is seen that high speedup is not achieved when the fixed size problem is distributed evenly over more than 3 or 4 processors. However, the speedup for the first two fixed per-processor problem sizes is much improved as seen in the two middle rows of Table I, since in these cases the computation times are higher than the communication times and the CC ratios are high. This is also illustrated in Fig. 1 at 14 MFLOP's. The other two fixed per-processor problem sizes 125x125x50 and 140x140x40 achieve poor speedups with 8 processors because of small CC ratios. However, given adequate RAM, the problem size can be set sufficiently large to maintain near-linear speedup despite using the slow networks such as Ethernet for message-passing. However, with optimized coding, higher MFLOP rates and hence smaller CC ratios would be obtained, and the speedup would suffer. Such a case is also illustrated in Fig. 1 for execution at 24 MFLOP's. Additional results [7] have also been compiled for the distributed computing of the FDTD approach, and in general the results indicate that PVM 3.2 is a flexible and reliable software for distributed computing as applied to the FDTD approach in Electromagnetics.

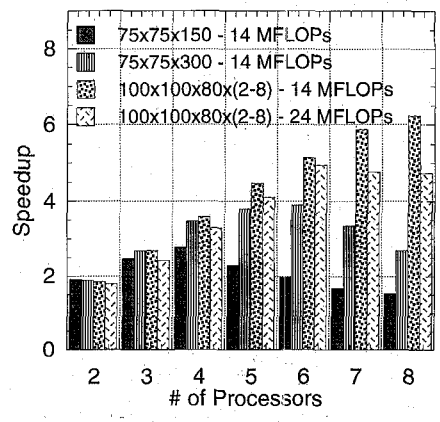

Fig. 1. Speedup achieved for two fixed problem sizes of 75x75x150, 75x75x300, and a fixed per-processor problem size of 100x100x80.

V. SUMMARY AND CONCLUSIONS

The 3-D FDTD calculation in Electromagnetics can be accomplished by using a cluster of workstations, and the job turnaround times can be significantly reduced. Cost-effective solutions can be found for FDTD computations since the existing reliable workstation hardware can be fully utilized using a freely available software such as PVM 3.2 and DQS and standard compilers and libraries. The large latency imposed by the serial nature of Ethernet communications and its low throughput (bandwidth) can be greatly minimized by using faster networking technologies such as ATM and FDDI. Moreover, the routers or 'Etherswitch' devices can increase, in a cost-effective manner, the aggregate Ethernet communication rate among N_P clustered processors by a factor of $N_P/2$, thereby significantly improving the speedup and the job turnaround times. This is accomplished by parallelizing the half-duplex communication in Ethernet using a star configuration for the cluster and an 'Etherswitch' for packet routing.

ACKNOWLEDGMENT

The authors acknowledge the help of Dr. Karthik Mahadevan of the Electromagnetic Communication Laboratory of the University of Illinois and John Quinn and the National Center for Supercomputing Applications at Urbana, IL for helping with various aspects of the work reported here.

REFERENCES

[1] J. J. Dongarra, G. A. Geist, R. Manchek, and V. S. Sunderam, "Integrated PVM Framework supports Heterogeneous Network Computing," *Comput. in Phys.*, April 1993.
[2] K. S. Yee, "Numerical solution of initial boundary-value problems involving Maxwell's equations in isotropic media," *IEEE Trans. Ant. Prop.*, vol. AP-14, pp. 302–307, 1966.
[3] A. Taflove and K. R. Umashankar, "The finite-difference time-domain method for numerical modeling of electromagnetic wave interactions," *Electromagnetics*, vol. 10, pp. 105–126, 1990.
[4] K. S. Kunz and R. J. Luebbers, *The Finite Difference Time Domain Method for Electromagnetics*. CRC Press, 1993.
[5] D. A. Reed and R. M. Fujimoto, *Multicomputer Networks - Message-Based Parallel Processing*. Cambridge, MA: MIT Press, 1987.
[6] G. C. Fox, M. A. Johnson, G. A. Lyzenga, S. W. Otto, J. K. Salmon, and D. W. Walker, *Solving Problems on Concurrent Processors*. Englewood Cliffs, NJ: Prentice Hall, 1988.
[7] V. Varadarajan and Raj Mittra, *Distributed Parallel Implementation of the Finite Difference Time Domain Algorithm using Parallel Virtual Machine* [PVM] *3.2*, in preparation.