Electromagnetics Computations Using the MPI Parallel Implementation of the Steepest Descent Fast Multipole Method (SDFMM)

M. El-Shenawee[1], C. Rappaport[2], D. Jiang[2], W. Meleis[2] and D. Kaeli[2]
[1]Department of Electrical Engineering
University of Arkansas
Fayetteville, AR 72701
magda@uark.edu
[2]Department of Electrical and Computer Engineering
Northeastern University
Boston, MA 02115
rappaport@neu.edu

ABSTRACT

The computational solution of large-scale linear systems of equations necessitates the use of fast algorithms but is also greatly enhanced by employing parallelization techniques. The objective of this work is to demonstrate the speedup achieved by the MPI (Message Passing Interface) parallel implementation of the Steepest Descent Fast Multipole Method (SDFMM). Although this algorithm has already been optimized to take advantage of the structure of the physics of scattering problems, there is still the opportunity to speed up the calculation by dividing tasks into components using multiple processors and solve them in parallel. The SDFMM has three bottlenecks ordered as (1) filling the sparse impedance matrix associated with the near-field Method of Moments interactions (MoM), (2) the matrix vector multiplications associated with this sparse matrix (3) the far field interactions associated with the fast multipole method. The parallel implementation task is accomplished using a thirty-one node Intel Pentium Beowulf cluster and is also validated on a 4-processor Alpha workstation. The Beowulf cluster consists of thirty-one nodes of 350MHz Intel Pentium IIs with 256 MB of RAM and one node of a 4x450MHz Intel Pentium II Xeon shared memory processor with 2GB of RAM with all nodes connected to a 100 BaseTX Ethernet network. The Alpha workstation has a maximum of four 667MHz processors. Our numerical results show significant linear speedup in filling the sparse impedance matrix. Using the 32-processors on the Beowulf cluster lead to achieve a 7.2 overall speedup while a 2.5 overall speedup is gained using the 4-processors on the Alpha workstation.

INTRODUCTION

The calculation of the scattered electric and magnetic fields from a three-dimensional problem using the conventional techniques (e.g., the Method of Moments (MoM), the Finite Element Method (FEM), or the Finite Differences in the time or frequency domains (FDTD or FDFD)) is a computationally intensive undertaking, especially for soils having a large dielectric constant. Moreover, the computational complexity of the problem dramatically increases upon inserting penetrable objects under the rough ground. Therefore, there was a necessity to use the fast computational algorithms to deal with this complex scenario. There exist few fast algorithms in the literature: the Fast Multipole Method (FMM) [1]-[3]; the SDFMM [4]-[6]; and the Sparse Matrix/Canonical Grid Method (SMCG) [7]-[8]. Basically, the standard FMM, the SDFMM, and the SMCG fast methods have the great advantage of converting the dense matrix obtained using the MoM into a sparse matrix leading to a dramatic reduction in the CPU time and computer memory requirements. In addition the fast algorithm, the Spectral Algorithm combined with the Forward-Backward Method (FB/NSA) [9], has shown to be an efficient iterative MoM for 3-D scattering problems.

In this work we adopted the SDFMM due to its superiority over the other fast algorithms in treating quasi-planar structures. The SDFMM is an integral equation-based fast algorithm that is a hybridization of (1) the Method of Moments (MoM), (2) the Fast Multipole Method (FMM), (3) the Steepest Descent Integration path (SDP) [4]-[6]. Recently the SDFMM has been successfully implemented to handle subsurface sensing applications, in particular, the scattering from a landmine modeled as a PEC and/or penetrable spheroid buried under a two dimensional randomly rough ground [10]-[11]. The SDFMM has computational complexity for the CPU time and for the memory requirement equal to only $O(N)$ per iteration versus $O(N^2)$ for the MoM, where N is the total number of the unknowns [4]-[6]. The reduced complexity of the SDFMM over several other computational electromagnetics techniques has helped in achieving a fast and successful running for the Monte Carlo

simulations [11]. However, the Monte Carlo sample needs in some cases to be greatly increased, e.g. when the ground random roughness increases the size of the Monte Carlo sample needs to be increased to achieve a converging solution. This could dramatically increase the required run time, especially when the dielectric constant of the ground is large and/or the penetrable buried object is electrically large. This necessitates more acceleration to the SDFMM computer code by using the MPI parallel implementation [8],[12],[13].

In this work, we used the MPI library for the parallel implementation of the SDFMM code [14]-[15]. The advantage of using the Beowulf cluster is that the system can be completely dedicated to the parallelization task, which is demonstrated in this work by executing small-scale cases due to memory limitations. Our emphasis is to demonstrate the overall speedup that can be achieved using the thirty-two processors. Porting the parallelized code to the national supercomputers, where hundreds of processors and adequate RAM are available, will potentially facilitate the computations of large-scale problems.

PARALLELIZATION METHODOLOGY

The SDFMM makes use of the equivalence theorem to calculate the electric and magnetic fields inside and outside a 3-D penetrable object buried under the rough surface interface [10]-[11]. The 3-D arbitrary object is modeled by scatterer R_3 that is immersed in scatterer R_2 which represents the rough ground which is immersed in the free space region represented by R_1. The three regions, R_1, R_2 and R_3 have permittivity and permeability given by ε_1 and μ_1, ε_2 and μ_2, and ε_3 and μ_3, respectively, representing free space, soil medium and penetrable buried object. There are two final sets of unknown equivalent electric and magnetic surface currents in the following formulations. They are $\bar{J}_1, \bar{M}_1$ on the exterior of the rough ground interface between R_1 and R_2, and $\bar{J}_3, \bar{M}_3$ on the exterior of the buried object interface between R_2 and R_3. Upon applying the boundary conditions, continuity of tangential components of the electric and magnetic fields on these interfaces, new integral equation formulations are obtained as equations 1a-d below [10]-[11], in which the integro-differential operators L_i and K_i, i =1, 2, 3 and 4, are given in detail in [11]. In Eqs. 1a-d, the unknown surface electric and magnetic currents are $\bar{J}_1$, $\bar{M}_1$, $\bar{J}_3$, and $\bar{M}_3$, while the tangential components of the incident electric and magnetic fields on the rough surface are given by $\bar{E}^{inc}(\bar{r})\big|_{\text{tang.}}$ and $\bar{H}^{inc}(\bar{r})\big|_{\text{tang.}}$, respectively. The intrinsic impedance in each region is $\eta_i = \sqrt{\mu_i / \varepsilon_i}$, i=1, 2, and 3, where the dielectric permittivity and permeability in each region are ε_i and μ_i, respectively. The equivalent electric and magnetic currents are approximated using the Rao, Wilton and Glisson (RWG) vector basis functions [16]-[17]. Upon applying Galerkin's method for testing and substituting the RWG surface current approximations in 1a-d, the original integral equations are transformed into a set of linear system of equations given by [10]-[11]:

$$\bar{\bar{Z}}\,\bar{I} = \bar{V} \tag{2a}$$

The impedance matrix $\bar{\bar{Z}}$ has order of $2(N+P)\times 2(N+P)$. The vector $\bar{V}$ is a matrix of order $2(N+P)\times 1$ and composed of a submatrix of the tested tangential incident electric field $\bar{E}^{inc}$ of order $N\times 1$ and a submatrix of the tested normalized magnetic field $\eta_1 \bar{H}^{inc}$ of order $N\times 1$, and a null submatrix of order $2P\times 1$. The quantities N and P are the numbers of basis functions (total number of edges of the triangular patches) on the surfaces of the rough ground and the buried object, respectively. If the MoM is used to formulate and solve Eq. (2a), the computation and storage of all the elements of the matrix $\bar{\bar{Z}}$ are required. Furthermore, if an iterative

$$\bar{E}^{inc}(\bar{r})\big|_{\text{tang.}} = \left[(L_1+L_2)\bar{J}_1 - (K_1+K_2)\bar{M}_1 - L_3\bar{J}_3 + K_3\bar{M}_3\right]_{\text{tang.}} \tag{1a}$$

$$\bar{H}^{inc}(\bar{r})\big|_{\text{tang.}} = \left[(K_1+K_2)\bar{J}_1 + \left(\frac{L_1}{\eta_1^2}+\frac{L_2}{\eta_2^2}\right)\bar{M}_1 - K_3\bar{J}_3 - \frac{L_3}{\eta_2^2}\bar{M}_3\right]_{\text{tang.}} \tag{1b}$$

$$0 = \left[-L_2\bar{J}_1 + K_2\bar{M}_1 + (L_3+L_4)\bar{J}_3 - (K_3+K_4)\bar{M}_3\right]_{\text{tang.}} \tag{1c}$$

$$0 = \left[-K_2\bar{J}_1 - \frac{L_2\bar{M}_1}{\eta_2^2} + (K_3+K_4)\bar{J}_3 + \left(\frac{L_3}{\eta_2^2}+\frac{L_4}{\eta_3^2}\right)\bar{M}_3\right]_{\text{tang.}} \tag{1d}$$

solution process is used, matrix-vector products involving multiplying $\overline{\overline{Z}}$ by a vector $\bar{I}$ are required. However, upon using the SDFMM [4]-[6],[10],[11] the matrix $\overline{\overline{Z}}$ becomes sparse and the system of equations in (2a) can be written as:

$$\overline{\overline{Z}}'\bar{I} + \overline{\overline{Z}}''\bar{I} = \bar{V} \qquad (2b)$$

where the matrix $\overline{\overline{Z}}'$ is a sparse matrix whose non-zero elements need to be calculated and stored using the conventional MoM and then multiplying them by the vector $\bar{I}$ (near field interactions) while the matrix-vector multiply $\overline{\overline{Z}}''\bar{I}$ is computed in one step without calculating or storing any elements of the matrix $\overline{\overline{Z}}''$. This was achieved by hybridizing the FMM with the SDP leading to the SDFMM [4]-[6]. There are three bottlenecks in the SDFMM computer code: (i) the subroutines that calculate the elements of the sparse matrix $\overline{\overline{Z}}'$; (ii) the subroutine that executes the matrix vector multiplication $\overline{\overline{Z}}'\bar{I}$ in every iteration; (iii) the subroutine that executes the fast multipole method for $\overline{\overline{Z}}''\bar{I}$ (far-field fast multipole interactions). These three bottlenecks in the serial SDFMM computer code are separately parallelized in the current work as pictorially described in Fig. 1 [18].

The key data structure of bottleneck (i) is the sparse matrix $\overline{\overline{Z}}'$ which is originally stored in the serial SDFMM computer code as blocks of nonzero elements. These elements represent the near-field interactions in the conventional MoM. The computations of these blocks are independent and therefore are parallelized in a straightforward manner by distributing them among all processors with no additional communication. When this routine is parallelized we achieved almost a linear speedup for bottleneck (i) on 32 processors. It is necessary to mention that the elements of $\overline{\overline{Z}}'$ remain distributed among the processors at the end of parallelizing bottleneck (i).

In the second bottleneck (ii) in the serial SDFMM computer code, the matrix-vector multiplication $\overline{\overline{Z}}'\bar{I}$ is executed every iteration of the transpose-free quasi-minimal residual (TFQMR) iterative solver [19]. This multiplication is parallelized by distributing the vector $\bar{I}$ to all processors similar to the elements of the sparse matrix $\overline{\overline{Z}}'$ in bottleneck (i). Therefore, the multiplication proceeds in parallel without additional communications and the vector components that result from the multiplication are then distributed to all processors.

In the third bottleneck (iii) in the serial SDFMM computer code, the fast multipole part for the matrix vector multiplication represented by $\overline{\overline{Z}}''\bar{I}$ in (2b) is computed for every iteration of the TFQMR iterative solver. This part of the serial code includes the computations of the Green's function approximations for the air and for the medium (e.g. soil). These two approximations of the Green's function are independent and are represented in the serial computer code by two separate subroutines; therefore they are executed concurrently as a first parallelization phase of bottleneck (iii). The load balance between these two subroutines is achieved using a detailed performance model based on the serial execution time of each routine, the time required for collective communication operations, and the amount of communication overhead needed. Moreover, in the serial computer code each one of these two subroutines includes all the multi-level FMM computations such as the inhomogeneous plane wave expansions and the dipole interactions at the finest level (aggregation), all interactions going up the tree, all the multi-level translation operations, all interactions going down the tree and finally the disaggregation process at the finest level which concludes the far-field interactions producing $\overline{\overline{Z}}''\bar{I}$ [1]-[6]. As a second phase of parallelizing bottleneck (iii), each Green's function approximation subroutine is parallelized but only at the finest level. In this work, no parallelization was conducted for the multi-level portion in these subroutines due to the existing complex interdependencies in the serial computer code. More parallelization future work is needed for this part.

NUMERICAL RESULTS

We evaluated the parallel implementation of the SDFMM computer code on a 32-node Intel Pentium-based Beowulf cluster. Thirty one nodes of the Beowulf cluster are 350MHz Intel Pentium IIs with 256 MB of RAM in addition to one node of a 4x450MHz Intel Pentium II Xeon shared memory processor with 2GB of RAM. The nodes are connected to a 100 BaseTX Ethernet network and they use the SuSE 6.1 operating system with Linux kernel 2.2.13, and the MPICH 1.2.1 implementation of the MPI library. Moreover, we tested the parallelized code on a 4-node shared memory Compaq Alpha-based workstation (667Mhz Alpha 21264) of 16GB total RAM. The processor uses the UNIX OSF/1 V5.1 operating system with the MPICH 1.1.2 MPI library.
Our benchmark includes three small-scale cases executed on the 256MB Intel cluster, and in addition one moderate-scale case that is executed on the Alpha workstation. To evaluate the speedup achieved by the parallel code, we considered a range of values for the

ground roughness and/or for the buried object. All results obtained by executing the parallel version of the code are validated with those computed by the serial version of the code [10]-[11]. In all computations a 10^{-3} tolerance is assumed for the TFQMR iterative solver [19]. The scattering problem configurations used in [11] are employed here, but for only one rough surface realization as shown in Fig. 2. The rough ground is characterized by Gaussian statistics with zero mean for the height, thus the roughness parameters can be described by the rms height σ and the correlation length l_c. In all cases, the relative dielectric constant of the ground soil and the penetrable buried object (anti-personnel mine) are $\varepsilon_r = 2.5 - j0.18$ and $\varepsilon_r = 2.9 - j0.0092$, respectively. A Gaussian beam with horizontal polarization is employed for the incident waves at normal incidence for Cases 1-3 and at 10^o from normal direction for Case 4 [11].

In the small-scale Cases 1-3, the dimensions of the modeled ground are assumed to be $3\lambda_0 \times 3\lambda_0$ leading to almost 8,800 of total number of surface current unknowns, while these dimensions are increased to be $8\lambda_0 \times 8\lambda_0$ for the moderate-scale Case 4 leading to 60,320 unknowns, where λ_0 is the free space wavelength. In Case 1, the scattered electric fields from a rough ground alone (no buried target) with $\sigma = 0.3\lambda_0$ and $l_c = 0.5\lambda_0$ are calculated at height of $1.2\lambda_0$ above the ground. In Case 2, the scattered electric fields from a rough ground with a buried penetrable sphere are calculated at height of $0.5\lambda_0$ above the ground. The ground roughness is assumed to be $\sigma = 0.1\lambda_0$ and $l_c = 0.5\lambda_0$ and the sphere has radius of $a = 0.16\lambda_0$ with burial depth equal to $z = -0.32\lambda_0$ measured from its center to the mean plane of the ground. The sphere in Case 2 is replaced by a spheroid of dimensions $a = 0.3\lambda_0$ and $b = 0.15\lambda_0$ in Case 3 that is buried at $z = -0.3\lambda_0$ with ground roughness equal to $\sigma = 0.04\lambda_0$ and $l_c = 0.5\lambda_0$.

Both the overall speedup and the initial speedup (filling matrix $\overline{\overline{Z}}'$) are plotted versus the number of processors for Cases 1, 2 and 3 in Figs. 3a, 3b and 3c, respectively. The speedup is defined as the ratio of the serial runtime to the parallel runtime. The results in these figures show the significant speedup in the initial time (set up) that is consumed to fill the sparse matrix $\overline{\overline{Z}}'$ as explained in Section II. This initial speedup dramatically affects the overall speedup of the code as shown in these figures. In addition, the results show that almost the same overall speedup can be achieved by employing only twelve instead of thirty-two processors.

The efficiency for a given number of processors is defined as the ratio of the speedup to the number of processors. In each case, the peak speedup is achieved when running on 32 processors, where for case 1, the peak speedup is 7.1 as shown in Fig. 3a, with a reduction in runtime from 99 minutes on one processor to 14 minutes on 32 processors. For Case 2, the peak speedup is 6.2 as shown in Fig. 3b, with a reduction in runtime from 90 minutes to 14 minutes while for Case 3, the peak speedup is 7.2 as shown in Fig. 3c, with a reduction in runtime from 88 minutes to 12 minutes. Over these three cases, the average speedup on 32 processors is 6.8, giving an efficiency of 0.21. Based on the serial runtimes, 88% of the code is executed in parallel. Therefore by Amdahl's Law [20], the peak speedup achievable is 8.3. We conclude that communication overhead and load imbalance among the processors account for the reduction in speedup from 8.3 to 6.8. An interesting comparison between the speedup achieved in each one of the bottlenecks (i)-(iii) mentioned in Section II, is shown in Fig. 4. These results show that the matrix-vector multiplication $\overline{\overline{Z}}'\bar{I}$ (that is the bottleneck (ii)) governs the overall speedup of the parallelized computer code.

In the second set of experiments, we solved the moderate-scale problem of Case 4 (60,320 unknowns) on the Alpha SMP using all four available processors. The penetrable spheroid of dimensions $a = 0.3\lambda_0$ and $b = 0.15\lambda_0$ is buried at $z = -0.3\lambda_0$ under the $8\lambda_0 \times 8\lambda_0$ rough ground with $\sigma = 0.04\lambda_0$ and $l_c = 0.5\lambda_0$. The magnitude of the total scattered electric field from the ground with the buried target is shown in Fig. 5. The magnitude of the scattered electric fields for just the buried spheroid is computed by subtracting the return from the rough ground using complex vector representation from the total return from the ground with the buried target [10]-[11]. The output is shown in Fig. 5b. The results of Fig. 5a and 5b clearly demonstrate that the signature of the buried plastic landmine is relatively small compared with the return from the ground which is considered a major source of clutter in landmine detection application. Moreover, the distortion observed in Fig. 5b is due to the roughness of the ground which is modeled here as only one random rough surface realization, however the Monte Carlo simulations case was presented in [11]. The serial version took 96 minutes to run this case while the parallel version took 37 minutes, giving a speedup of 2.5 and an efficiency of 0.63. The predicted peak speedup on the four processors is 2.9. This

implies that executing the parallel code on the 4-Alpha 667MHz processor gives a remarkable reduced absolute runtime for this moderate-scale case. This achievement can be exploited to execute large-scale scattering problems as mentioned in Section II. For the memory requirements, the serial version of the code requires 950MB of RAM while the parallel version requires 1154MB of RAM distributed over the four processors as 288, 290, 289 and 287MB, respectively. Table I summarizes the parameters and output results for all cases presented in this section.

The results described in this section demonstrate that by implementing the fine grained parallelism, we have achieved good speedups when using a single rough surface realization (one run of the code). This achievement is suitable for some subsurface scattering configurations where we may need to obtain multiple views of a target buried under the same rough surface realization [10]. This requires running the code several times. However, the current speedup is not suitable when the number of rough surface realizations is much larger than the number of available processors, e.g. Monte Carlo simulations, due to the saturation occurs in the speedup of the second and third bottlenecks.

CONCLUSIONS

Good overall speedup has been achieved as the SDFMM computer code is parallelized using the MPI library. The linear speedup obtained for the first bottleneck associated with filling the sparse impedance matrix is significant. Sensible speedups are obtained for the second and third bottlenecks associated with the matrix vector multiplication in the near-field and the far-field FMM approximations, respectively. However, the later speedups saturate upon using only 12 processors out of the 32 nodes available on the system. This saturation affects the overall speedup of the computer code and limits its application. More parallelization work is needed to enhance the speedup to be used for large Monte Carlo simulations.

ACKNOWLEDGMENTS

This research was sponsored in part by the Army Research Office Demining MURI grant # DAA 0-55-97-0013, in part by the Engineering Research Centers Program of the NSF under award number EEC-9986821, and in part by the College of Engineering at the University of Arkansas. This work benefited from the allocation of time at the Joulian Cluster and at the Advanced Scientific Computation Center (NU-ASCC) at Northeastern University. The SDFMM was originally developed by V. Jandhyala, E. Michielssen and W. Chew at the UIUC.

REFERENCES

[1] V. Rokhlin, "Rapid solution of integral equations of scattering theory in two dimensions," *J. Comput. Phys.,* vol. 36, pp. 414-439, 1990.

[2] C. C. Lu and W. C. Chew, "A multilevel fast-algorithm for solving a boundary integral equation of wave scattering," *Microwave Opt. Tech. Let*., vol. 7, pp. 466-470, July 1994.

[3] N. Geng, A. Sullivan and L. Carin, "Multilevel fast-multipole algorithm for scattering from conducting targets above or embedded in a lossy half space," *IEEE Trans. Geosci. Remote Sensing*, vol. 38, no. 4, pp. 1561-1573, July 2000.

[4] V. Jandhyala, *Fast Multilevel Algorithms for the Efficient Electromagnetic Analysis of Quasi-Planar Structures*, Ph.D. Thesis, Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, 1998.

[5] V. Jandhyala, B. Shanker, E. Michielssen and W. C. Chew, "A fast algorithm for the analysis of scattering by dielectric rough surfaces," *J. Opt. Soc. Am. A,* vol. 15, no. 7, pp. 1877-1885, July 1998.

[6] M. El-Shenawee, V. Jandhyala, E. Michielssen and W. C. Chew, "An enhanced steepest descent fast multipole method for the analysis of scattering from two dimensional multilayered rough surfaces," *Proc. of the IEEE APS/URSI '98 Conf.*, Atlanta, Georgia, p. 182, June 1998.

Table I Overall speedup

Case #	Number of Unknowns	σ	Object	System	Number of Processors	Serial/Par. time (min.)	Speedup (overall)
1	8,800	$0.3\lambda_o$	None	Cluster	32	99/14	7.1
2	8,800	$0.1\lambda_o$	Sphere	Cluster	32	90/14	6.2
3	8,800	$0.04\lambda_o$	Spheroid	Cluster	32	88/12	7.2
4	60,320	$0.04\lambda_o$	Spheroid	Alpha Server	4	96/37	2.5

[7] G. Zhang, L. Tsang and K. Pak, "Angular correlation function and scattering coefficient of electromagnetic waves scattered by a buried object under a two-dimensional rough surface," *J. Opt. Soc. Am. A*, vol. 15, no. 12, pp. 2995-3002, December 1998.

[8] S. Li, C. H. Chan, L. Tsang, Q. Li, and L. Zhou, "Parallel implementation of the sparse matrix/canonical grid method for the analysis of two-dimensional random rough surfaces (three-dimensional scattering problem) on a Beowulf system," *IEEE Trans. Geosci. Remote Sensing*, vol. 38, no. 4, pp. 1600-1608, July 2000.

[9] D. Torrungrueng, H. Chou and J. T. Johnson, "A novel acceleration algorithm for the computation of scattering from two-dimensional large scale perfectly conducting random rough surfaces with the forward-backward method," *IEEE Trans. Geosci. Remote Sensing*, vol. 38, no. 7, pp. 1656-1666, July 2000.

[10] M. El-Shenawee, C. Rappaport, E. Miller and M. Silevitch, "3-D subsurface analysis of electromagnetic scattering from penetrable/PEC objects buried under rough surfaces: Use of the steepest descent fast multipole method (SDFMM)," *IEEE Trans. Geosci. Remote Sensing*, vol. 39, no. 6, pp. 1174-1182, June 2001.

[11] M. El-Shenawee, C. Rappaport and M. Silevitch, "Monte Carlo simulations of electromagnetic wave scattering from random rough surface with 3-D penetrable buried object: Mine detection application using the SDFMM," the *J. Optical Society of America A*, vol.18, no. 12, pp.3077-3084, December 2001.

[12] S. V. Velamparambil, J. E. Schutt-Aine, J. G. Nickel, J. M. Song, and W. C. Chew, "Solving large scale electromagnetic problems using a linux cluster and parallel MLFMA," *IEEE Antennas Propagat. Symp.*, 1:636-639, July 1999.

[13] S. Velamparambil, J. Song, and W. C. Chew, "On the parallelization of dynamic multilevel fast multipole method on distributed memory computers," *Proc. International Workshop on Innovative Architecture for Future Generation High-Performance Processors and Systems, Maui, Hawaii*, November 1999.

[14] J. J. Dongarra and D. W. Walker, "MPI: A Standard Message Passing Interface", Supercomputer, Vol. 12, No. 1, 1996, pp. 56-68.

[15] Message Passing Interface Forum. MPI: A Message-Passing Interface standard. *The International Journal of Supercomputer Applications and High Performance Computing*, no. 8, 1994.

[16] S. M. Rao, D. R. Wilton, and A. W. Glisson, "Electromagnetic scattering by surfaces of arbitrary shape," *IEEE Trans. on Anten. and Prop*., vol. AP 30, no. 3, pp.409-418, May 1982.

[17] L. Medgyesi-Mitschang, J. Putnam, and M. Gedera, "Generalized method of moments for three–dimensional penetrable scatterers," *J. Opt. Soc. Am. A*, vol. 11, no. 4, pp. 1383-1398, April 1994.

[18] D. Jiang, W. Meleis, M. El-Shenawee, E. Mizan, M. Ashouei and C. Rappaport, "Parallel Implementation of the Steepest Descent Fast Multipole Method (SDFMM) On a Beowulf Cluster for Subsurface Sensing Applications," the *IEEE Microwave and Wireless Components Letters* (*MWCL*), vol. 12, no. 1, pp. 1-3, January 2002.

[19] R. W. Freund, "A transpose-free quasi-minimal residual algorithm for non-hermitian linear systems," *SIAM J. Sci. Comput*., vol. 14, no. 2, pp. 470-482, March 1993.

[20] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities", *Proc. AFIPS Spring Joint computer Conference* 30, Atlantic City, N. J., pp. 483-485, April 1967.

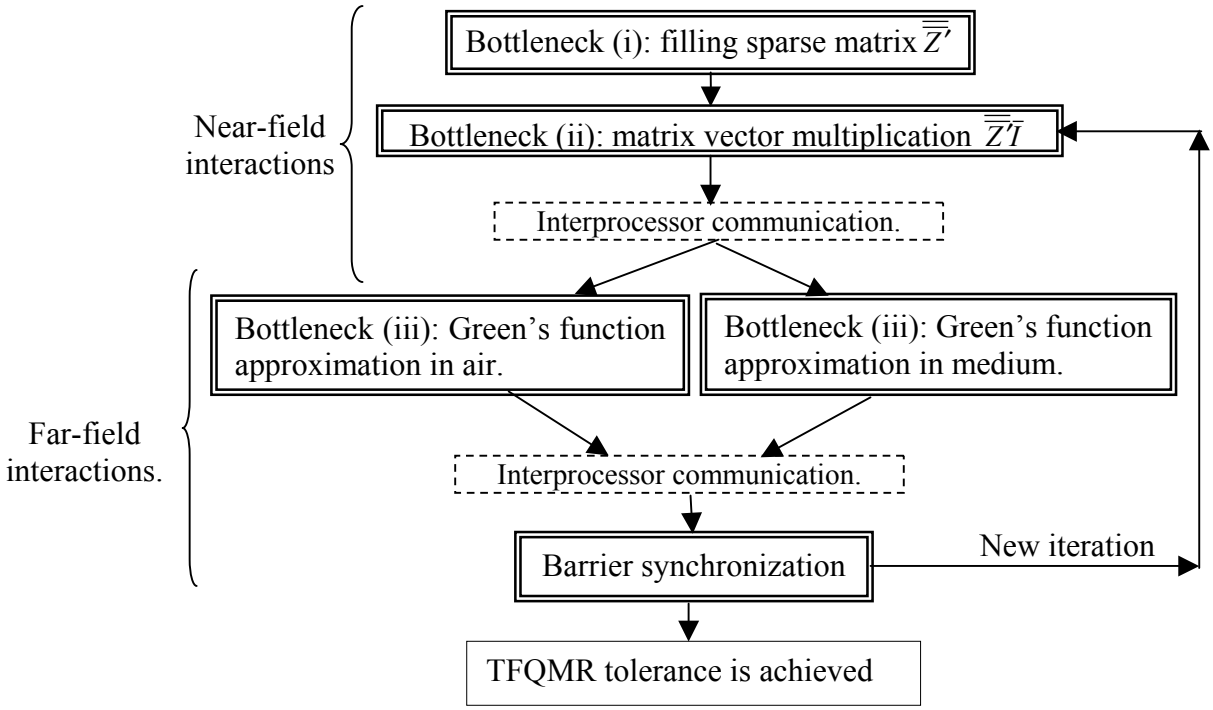


Fig. 1. Structure of parallelized SDFMM showing major computational tasks and their interrelation.

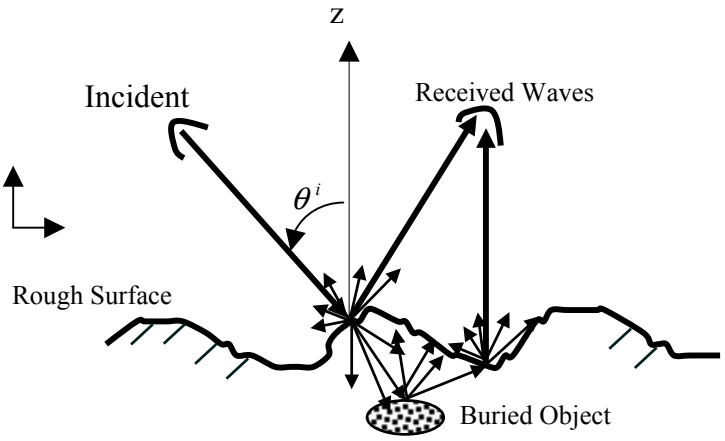
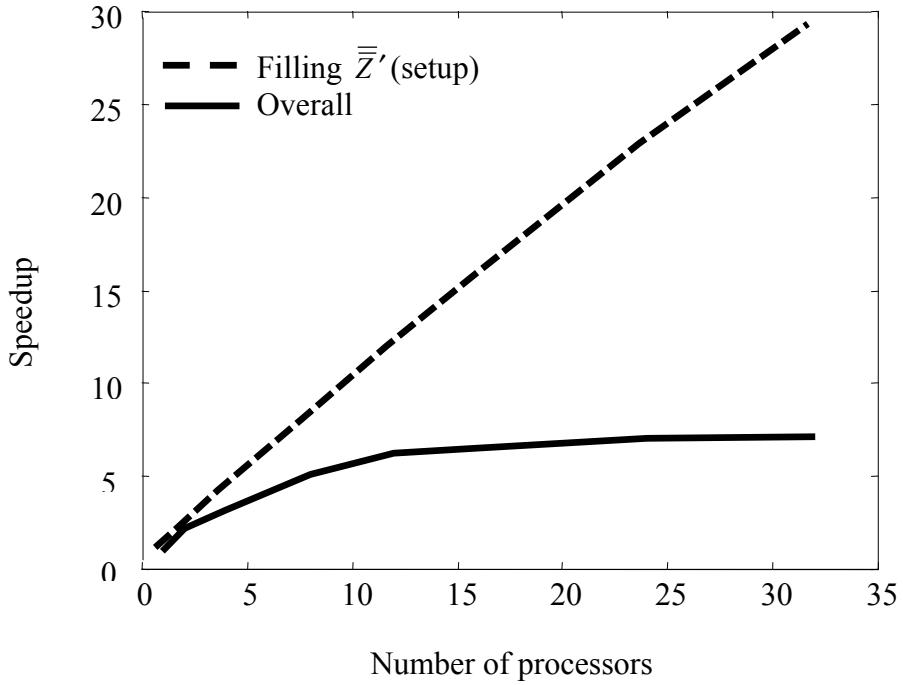
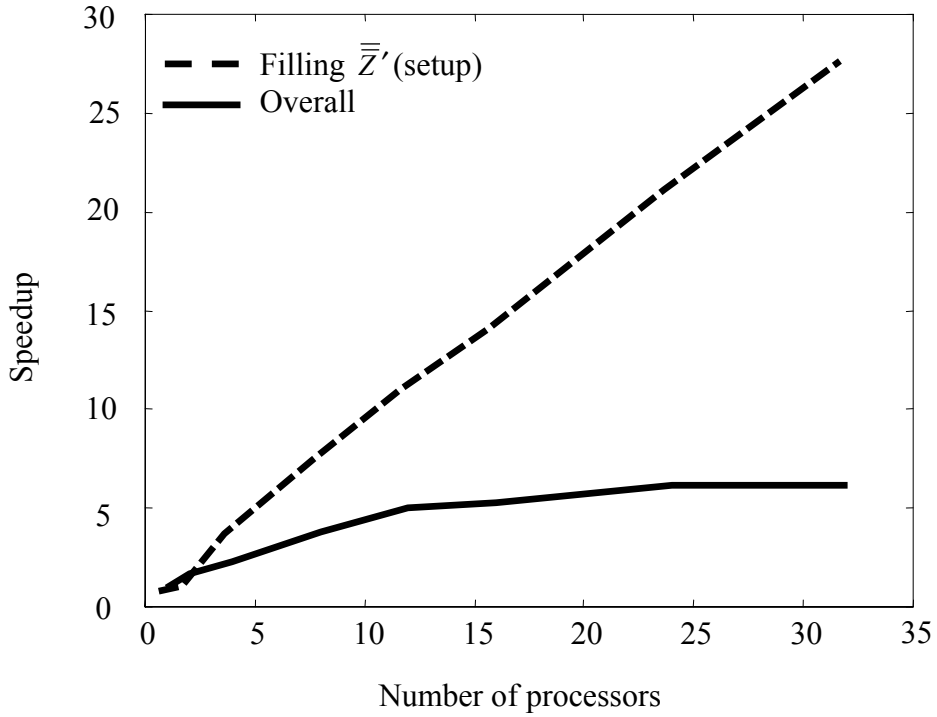


Fig. 2. Cross section of the object buried under the rough ground (3-D problem).

(a)


Filling $\overline{\overline{Z}}'$(setup)
Overall
Speedup
Number of processors
0
5
10
15
20
25
30
35

(b)


Filling $\overline{\overline{Z}}'$(setup)
Overall
Speedup
Number of processors
0
5
10
15
20
25
30
35

(c)

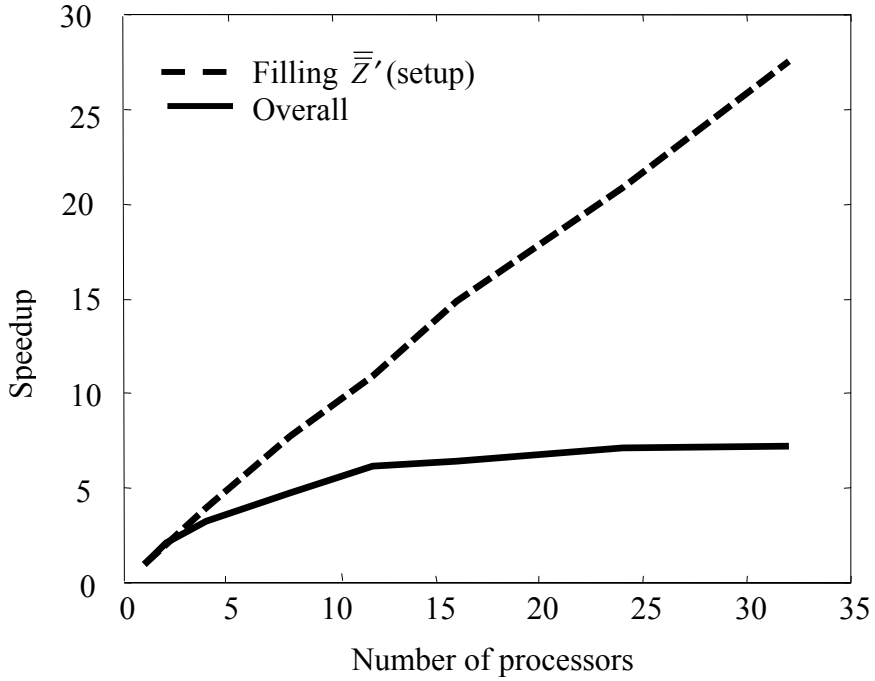


Fig. 3. Speedup of the Beowulf cluster: (a) Case 1, target-free rough ground, (b) Case 2, penetrable sphere buried under moderately rough ground surface, (c) Case 3, penetrable spheroid buried under slightly rough ground surface.

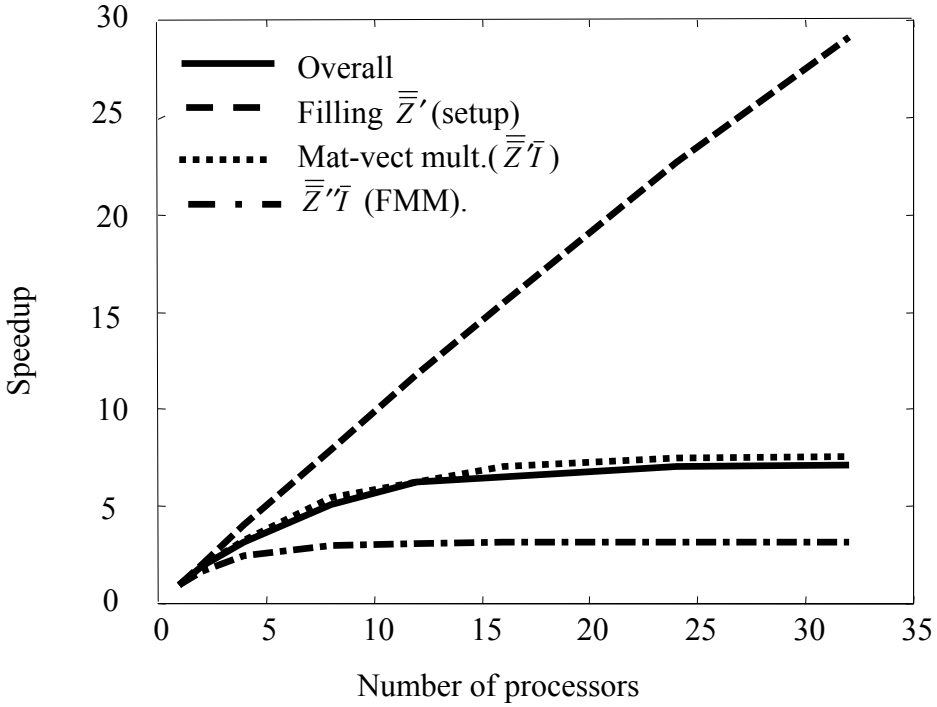


Fig. 4. Performance improvement for each of the separate component tasks and overall speedup of the SDFMM algorithm, as a function of the number processors in the Beowulf cluster.

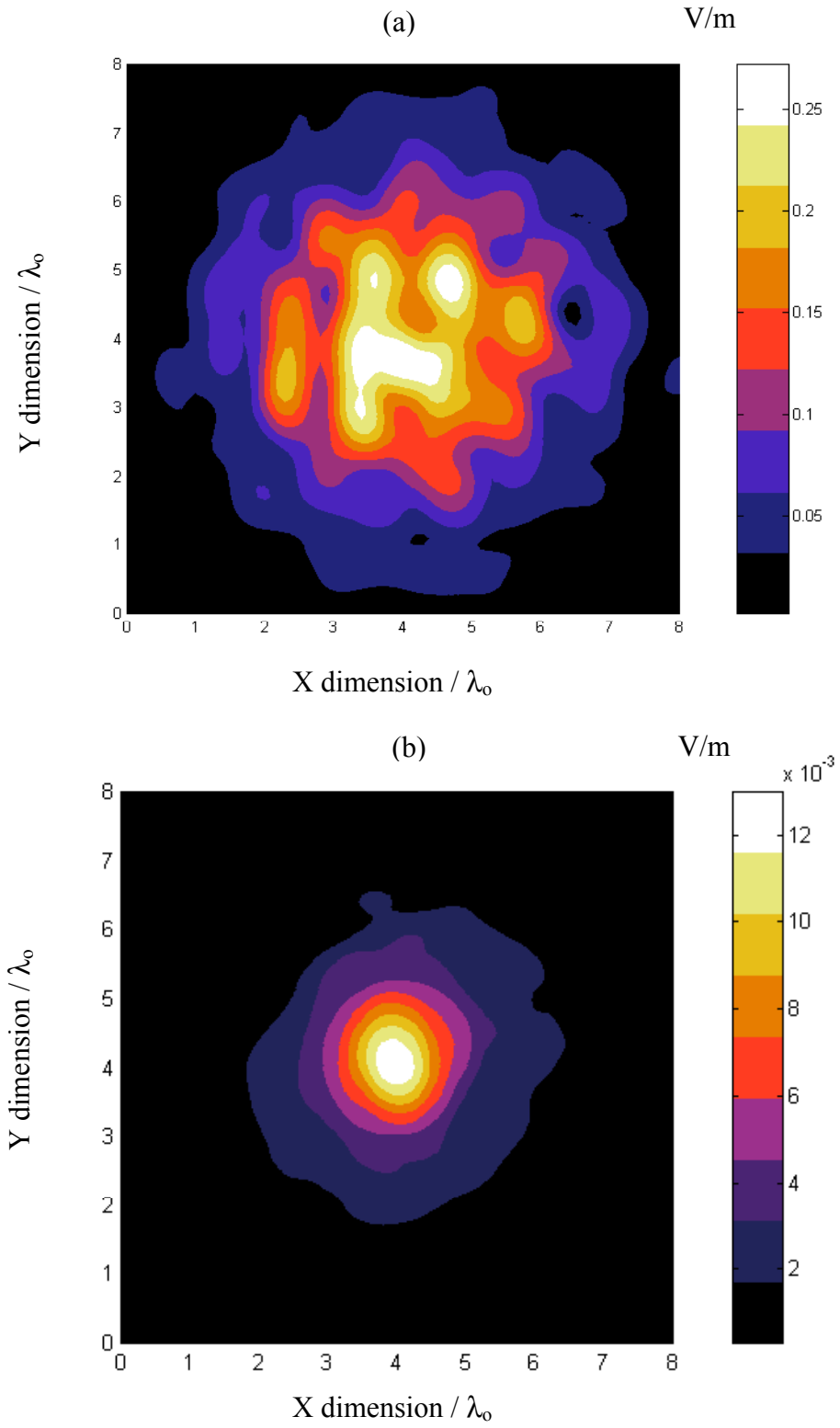


Fig. 5 The near electric field scattered above the rough ground at $z = 0.5\lambda_0$ for the spheroid of Case 4 ($a = 0.3\lambda_0$, $b = 0.15\lambda_0$, buried at $z = -0.3\lambda_0$ in conductive clay loam soil), computed using the 4-processor Alpha Server: (a) the rough ground with the buried spheroid (total field), (b) just the spheroid obtained by subtraction.

Magda El-Shenawee received the Ph.D. degree in electrical engineering from the University of Nebraska-Lincoln in 1991. In 1992, she worked as a Research Associate in the Center for Electro-Optics at the University of Nebraska. In 1997, she worked as Visiting Scholar at the University of Illinois at Urbana-Champaign and in 1999, she joined the Center for Electromagnetics Research at Northeastern University, Boston. Currently, Dr. El-Shenawee is an assistant professor in the Department of Electrical Engineering at the University of Arkansas, Fayetteville. Her research areas are rough surface scattering, computational electromagnetics, subsurface sensing of buried objects, breast cancer modeling, numerical methods, and micro-strip circuits. Dr. El-Shenawee is a member of Eta Kappa Nu electrical engineering honor society.

Carey M. Rappaport received five degrees from the Massachusetts Institute of Technology: the SB in Mathematics, the SB, SM, and EE in Electrical Engineering in June 1982, and the PhD in Electrical Engineering in June 1987. Prof. Rappaport has worked as a teaching and research assistant at MIT from 1981 until 1987, and during the summers at COMSAT Labs in Clarksburg, MD, and The Aerospace Corp. in El Segundo, CA. He joined the faculty at Northeastern University in Boston, MA in 1987. During Fall 1995, he was Visiting Professor of Electrical Engineering at the Electromagnetics Institute of the Technical University of Denmark, Lyngby, as part of the W. Fulbright International Scholar Program. He has consulted for Geo-Centers, Inc., PPG, Inc. on wave propagation and modeling, and microwave heating and safety. He is Principal Investigator of the ARO Multidisciplinary University Research Initiative in Humanitarian Demining and Co-Principal Investigator of the Center for Subsurface Sensing and Imaging Systems (CenSSIS) Engineering Research Center. Prof. Rappaport has authored over 180 technical journal and conference papers in the areas of microwave antenna design, electromagnetic scattering computation, and bio-electromagnetics, and has received two reflector antenna patents, two biomedical device patents and three subsurface sensing device patents. He was awarded the IEEE Antenna and Propagation Society's H.A. Wheeler Award for best applications paper of 1985. He is a member of Sigma Xi and Eta Kappa Nu professional honorary societies.

Desheng Jiang received the BS degree in Chemistry from Jiangxi Normal University in 1993, the MS in Oceanography from Texas A&M University in 1999, and the MS in Electrical Engineering from Northeastern University in 2001, respectively. He is currently a Software Engineer with Openwave Systems Inc.

Waleed M. Meleis received the BSE degree in electrical engineering from Princeton University in 1990 and the MS and PhD degrees in computer science and engineering from the University of Michigan in 1992 and 1996, respectively. He is an assistant professor in the Department of Electrical and Computer Engineering at Northeastern University. His research interests are in scheduling algorithms and bounds for modern processors and compilers.

David R. Kaeli received his B.S. in Electrical Engineering from Rutgers University, his M.S. in Computer Engineering from Syracuse University, and his PhD in Electrical Engineering from Rutgers University. He is currently an Associate Professor on the faculty of the Department of Electrical and Computer Engineering at Northeastern University. Prior to 1993, he spent 12 years at IBM, the last 7 at IBM T.J. Watson Research in Yorktown Heights, N.Y.. In 1996 he received an NSF CAREER Award. He currently directs the Northeastern University Computer Architecture Research Laboratory (NUCAR). Dr. Kaeli's research interests include computer architecture and organization, compiler optimization, VLSI design, trace-driven simulation and workload characterization. He is a member of the IEEE and ACM. He is presently an Associate Editor for IEEE Transactions on Computer and IEEE Computer Architecture Letters.