Efficient Parallelization of the Multilevel Fast Multipole Algorithm for the Solution of Large-Scale Scattering Problems

Özgür Ergül, *Student Member, IEEE*, and Levent Gürel, *Senior Member, IEEE*

***Abstract*—We present fast and accurate solutions of large-scale scattering problems involving three-dimensional closed conductors with arbitrary shapes using the multilevel fast multipole algorithm (MLFMA). With an efficient parallelization of MLFMA, scattering problems that are discretized with tens of millions of unknowns are easily solved on a cluster of computers. We extensively investigate the parallelization of MLFMA, identify the bottlenecks, and provide remedial procedures to improve the efficiency of the implementations. The accuracy of the solutions is demonstrated on a scattering problem involving a sphere of radius 110λ discretized with 41 883 638 unknowns, the largest integral-equation problem solved to date. In addition to canonical problems, we also present the solution of real-life problems involving complicated targets with large dimensions.**

***Index Terms*—Electromagnetic scattering, fast solvers, integral equations, multilevel fast multipole algorithm (MLFMA), parallel algorithms.**

I. Introduction

Surface integral equations are commonly used to formulate scattering problems involving three-dimensional conducting bodies with arbitrary shapes [1]. These formulations provide accurate results when they are discretized appropriately by using small elements with respect to wavelength. Simultaneous discretizations of the scatterer and the integral equations lead to dense matrix equations, which can be solved iteratively using efficient acceleration methods, such as the multilevel fast multipole algorithm (MLFMA) [2]. However, accurate solutions of many real-life problems require discretizations with millions of elements leading to matrix equations with millions of unknowns. To solve these large problems, it is helpful to increase computational resources by assembling parallel computing platforms and at the same time by parallelizing the solvers.

Of the various parallelization schemes for MLFMA, the most popular use distributed-memory architectures by constructing clusters of computers with local memories connected via fast networks [3]–[11]. Parallelization tools are available, such as the message passing interface (MPI). Such tools provide many communication protocols to organize parallel solutions. However, parallelization of MLFMA is not trivial because of the complicated structure of this algorithm [11]. Simple parallelization strategies usually fail to provide efficient solutions because of the communications between the processors and the unavoidable duplication of some of the computations over multiple processors. Consequently, there have been many efforts to improve the parallelization of MLFMA by minimizing duplications and communications [7]–[12]. Thanks to these efforts, it has become possible to solve 20–30 million unknowns on relatively inexpensive computing platforms [8], [9], [13].

In this paper, we present the details of a parallel MLFMA implementation for the efficient solution of scattering problems involving tens of millions of unknowns. We extensively investigate the parallelization procedure by focusing on different parts of the algorithm and identifying the obstacles to parallelization efficiency. Our approach involves load-balancing and partitioning techniques to distribute the tasks equally among the processors and to minimize the interprocessor communications. We demonstrate the accuracy and efficiency of our implementations on canonical problems involving sphere geometries of various sizes. Specifically, we are able to solve problems with more than 40 million unknowns on relatively inexpensive platforms. In addition to canonical problems, we also solve real-life problems involving complicated geometries discretized with large numbers of unknowns.

The scattering problems considered in this paper involve closed surfaces, which can be formulated with the combined-field integral equation (CFIE) [1]. CFIE provides better-conditioned matrix equations than the electric-field integral equation (EFIE) and the magnetic-field integral equation (MFIE) [14]–[16]. Using CFIE, iterative convergence is achieved rapidly and it can be further accelerated by employing simple and efficient preconditioners.

The rest of the paper is organized as follows. In Section II, we examine the MLFMA solutions, focusing on the computational requirements. Section III explores efficient parallelization of MLFMA by investigating each part of the algorithm in detail. Section IV presents the results, followed by our concluding remarks in Section V.

II. Solution of Integral Equations by MLFMA

For the solution of scattering problems involving three-dimensional conducting bodies with arbitrary shapes, discretization of the surface integral equations leads to $N \times N$ dense

Manuscript received June 24, 2007; revised November 8, 2007. Published August 6, 2008 (projected). This work was supported in part by the Scientific and Technical Research Council of Turkey (TUBITAK) under Research Grants 105E172 and 107E136, in part by the Turkish Academy of Sciences in the framework of the Young Scientist Award Program (LG/TUBA-GEBIP/2002-1-12), and in part by contracts from ASELSAN and SSM. Computer time was provided in part by a generous allocation from Intel Corporation.

The authors are with the Department of Electrical and Electronics Engineering and the Computational Electromagnetics Research Center (BiLCEM), Bilkent University, TR-06800 Bilkent, Ankara, Turkey (e-mail: ergul@ee.bilkent.edu.tr; lgurel@bilkent.edu.tr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAP.2008.926757

matrix equations

$$\sum_{n=1}^{N} Z_{mn}^{E,M,C} a_n = v_m^{E,M,C}, \quad m = 1, 2, \ldots, N \tag{1}$$

where a_n represents the unknown coefficients of the basis functions $\boldsymbol{b}_n(\boldsymbol{r})$ for $n = 1, 2, \ldots, N$ to model the surface current density, i.e.,

$$\boldsymbol{J}(\boldsymbol{r}) \approx \sum_{n=1}^{N} a_n \boldsymbol{b}_n(\boldsymbol{r}). \tag{2}$$

Expressions for the matrix elements (Z_{mn}^E, Z_{mn}^M, and Z_{mn}^C) and the elements of the right-hand side vector (v_m^E, v_m^M, and v_m^C) for EFIE, MFIE, and CFIE, respectively, are presented in [17]. For the solution of problems involving closed surfaces, CFIE is preferable since it is free of the internal-resonance problem [18] and provides better-conditioned matrix equations than EFIE and MFIE [14]–[16]. This favorable quality of CFIE is crucial for the rapid convergence of iterative solutions. In this paper, CFIE is discretized by employing Rao-Wilton-Glisson (RWG) [19] functions defined on planar triangles for numerical solutions.

A. Solutions With MLFMA

MLFMA splits the matrix-vector multiplications (MVMs) required by the iterative solvers as

$$\bar{\boldsymbol{Z}} \cdot \boldsymbol{x} = \bar{\boldsymbol{Z}}_{NF} \cdot \boldsymbol{x} + \bar{\boldsymbol{Z}}_{FF} \cdot \boldsymbol{x}. \tag{3}$$

In (3), the near-field interactions denoted by $\bar{\boldsymbol{Z}}_{NF}$ are calculated directly and stored in memory, while the far-field interactions ($\bar{\boldsymbol{Z}}_{FF}$) are computed approximately in a group-by-group manner. For a single-level fast multipole algorithm, we calculate the far-field interactions as presented in [20]. In MLFMA, those interactions are calculated in a multilevel scheme using a tree structure constructed by including the scatterer in a cubic box and recursively dividing the computational domain into subboxes. The tree structure of MLFMA includes $L = O(\log N)$ levels. At level l from 1 to L, the number of nonempty boxes (clusters)[1] is N_l, where $N_1 = O(N)$ and $N_L = O(1)$. Each MVM involves four main stages.

- Near-field interactions: In MLFMA, near-field interactions are used directly to perform the multiplication

$$\boldsymbol{y} = \bar{\boldsymbol{Z}}_{NF} \cdot \boldsymbol{x}. \tag{4}$$

  The number of near-field interactions is proportional to N^2/N_1 and the near-field matrix $\bar{\boldsymbol{Z}}_{NF}$ has a sparsity of $1/N_1$.
- Aggregation: Radiated fields at the centers of the clusters are calculated from the bottom of the tree structure to the highest level.

[1]In this paper, the term "cluster" is used in two different contexts. Its meanings in "clusters of computers" and to indicate the nonempty boxes in the MLFMA tree should be distinguishable from the context.

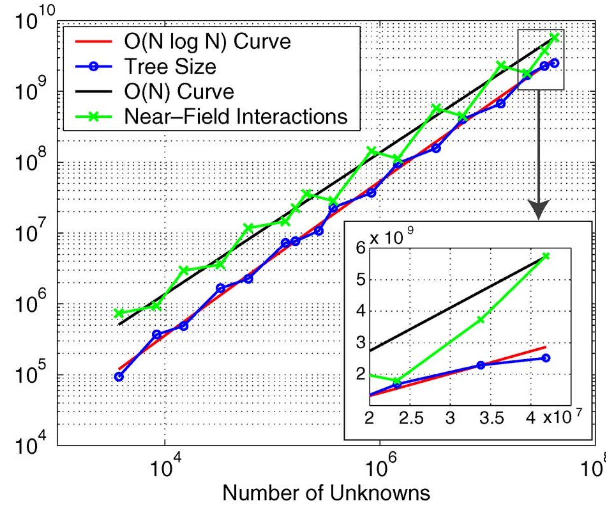

Fig. 1. Tree size and the number of near-field interactions for the solutions of the sphere problems using top-down strategy to construct the multilevel tree.

- Translation: Radiated fields are translated into incoming fields. For a basis cluster at any level, there are $O(1)$ testing clusters to translate the radiated field.
- Disaggregation: The incoming fields at the centers of the clusters are calculated from the top of the tree structure to the lowest level. At the lowest level, the incoming fields are multiplied by the receiving patterns of the testing functions and angular integrations are performed to complete the MVM.

In our MLFMA implementations, radiated and incoming fields are sampled uniformly in the ϕ direction, while we use the Gauss-Legendre quadrature in the θ direction [21]. There are a total of $(T_l + 1) \times (2T_l + 2)$ samples required for a cluster in level l, where T_l is the truncation number, i.e., the number of harmonics used to calculate the translation operators. To determine the value of T_l for each level, we use the excess bandwidth formula considering the worst-case scenario according to a one-box-buffer scheme [22], i.e.,

$$T_l \approx 1.73 k a_l + 2.16 (d_0)^{2/3} (k a_l)^{1/3} \tag{5}$$

where a_l is the box size at level l and d_0 is the desired digits of accuracy. Oscillatory nature of the Helmholtz solutions requires that the truncation number T_l and the sampling rate for the radiated and incoming fields depend on cluster size as measured by the wavelength ($\lambda = 2\pi/k$). During the aggregation and disaggregation stages, we employ local Lagrange interpolation and anterpolation methods to match the different sampling rates of the consecutive levels [23], [24].

B. Computational Requirements of MLFMA

When MLFMA is used, memory requirement for a MVM (M_{MVM}) is proportional to the tree size S_T, i.e.,

$$M_{MVM} \propto S_T = \sum_{l=1}^{L} 2 N_l (T_l + 1)^2. \tag{6}$$

TABLE I
MAJOR PARTS OF MLFMA AND THEIR COMPUTATIONAL REQUIREMENTS

	MEMORY		
PART	PROPORTIONAL TO	COMPLEXITY	SIGNIFICANCE
MVM	$\sum_{l=1}^{L} N_l(T_l+1)^2$	$O(N\log N)$	Significant
Radiation and Receiving Patterns	$N(T_1+1)^2$	$O(N)$	Significant
Translation Operators	$\sum_{l=1}^{L}(T_l+1)^2$	$O(N)$	Insignificant
Near-Field Interactions	N^2/N_1	$O(N)$	Significant
	PROCESSING TIME		
PART	PROPORTIONAL TO	COMPLEXITY	SIGNIFICANCE
MVM	$\sum_{l=1}^{L} c_l N_l(T_l+1)^2$	$O(N\log N)$	Significant
Radiation and Receiving Patterns	$N(T_1+1)^2$	$O(N)$	Insignificant
Translation Operators	$\sum_{l=1}^{L}(T_l+1)^2$	$O(N)$	Insignificant
Near-Field Interactions	N^2/N_1	$O(N)$	Significant

The processing time (T_{MVM}) is also related to the tree size as

$$T_{MVM} \propto \sum_{l=1}^{L} c_l N_l (T_l+1)^2 \tag{7}$$

where c_l represents relative weights for levels $l=1,2,\ldots,L$. Asymptotically, as N increases, $N_l(T_l+1)^2$ becomes $O(N)$ and the complexity of the MVM is $O(N\log N)$. Although this is true in general, measurements may present deviations from the ideal case depending on the construction technique for the tree structure, even when N is very large. For example, we usually employ a top-down strategy to build the multilevel tree for large problems. In this strategy, the smallest possible cubic box is used to enclose the target completely. Then, the computational domain is recursively divided into subdomains until the size of the clusters in the lowest level is in the 0.15λ–0.30λ range. In Fig. 1, tree size (S_T) is plotted as a function of the number of unknowns for the solution of scattering problems involving sphere geometries of various sizes, when the top-down strategy is used to construct the multilevel tree and the number of accurate digits d_0 is 2. The radius of the sphere changes from λ to 110λ corresponding to 3723 and 41 883 648 unknowns (edges), respectively, using $\lambda/10$ triangulation. We observe that the tree size oscillates around the $O(N\log N)$ curve. Due to such local variations, processing time and memory requirement for the MVMs with respect to N cannot be strictly proportional to $N\log N$. As an example, the tree size grows only by 50% when the number of unknowns increases from 23 405 664 to 41 883 648. Then, the memory requirement for the MVMs increases by about 50%, which is below the asymptotical estimation of 85%.

The radiation and receiving patterns of the basis and testing functions are sampled according to the sampling rate of the lowest level clusters. Using the RWG functions, these patterns are calculated analytically and stored in memory before the iterative solutions. Applying a Galerkin scheme and using the same sets of basis and testing functions, CFIE implementations require only two sets of patterns for each RWG function [25]. We also reduce the number of samples to $(T_1/2+1)\times(2T_1+2)$ using the symmetry of the patterns. Although the processing time to calculate the radiation and receiving patterns is negligible, significant amount of memory is required to store them.

Similar to the radiation and receiving patterns, translation operators are also calculated and stored in memory before the iterations. Using cubic (identical) clusters, there is a maximum of $7^3-3^3=316$ different translations in each level, independent of the number of clusters [7]. Although using cubic clusters reduces the number of translation operators significantly, we also need interpolation methods to calculate these operators in $O(N)$ time [26], [27]. With the optimization of the interpolations, both calculation time and memory for the translation operators are insignificant compared to the other parts of the implementation, especially when the problem size is large.

Processing time for the initial setup of MLFMA (prior to the iterative solution) is dominated by calculating near-field interactions and it is proportional to N^2/N_1. The amount of memory to store the near-field interactions is also significant and comparable to the memory used for the radiation and receiving patterns. Asymptotically, $N_1=O(N)$ and the near-field interactions has a complexity of $O(N)$. However, similar to the MVMs, local variations in the processing time and memory requirement for the near-field interactions may exhibit behavior different than the asymptotical estimation. This is because, as depicted in Fig. 1, the number of near-field interactions oscillates around the $O(N)$ curve when a top-down strategy is used to construct the tree structure. Consequently, variation in processing time and memory with respect to N can be higher or lower than the asymptotically linear estimate.

As a summary, Table I lists the major parts of MLFMA and their computation requirements for the solution of large problems.

III. EFFICIENT PARALLELIZATION OF MLFMA

Because of its complicated structure, parallelization of MLFMA is not trivial. Simple parallelization schemes usually lead to inefficient solutions due to dense communications between the processors, duplication of computations, and unbalanced distribution of the workload among processors. Several issues must be carefully considered to obtain an efficient parallelization of MLFMA [7]–[12].

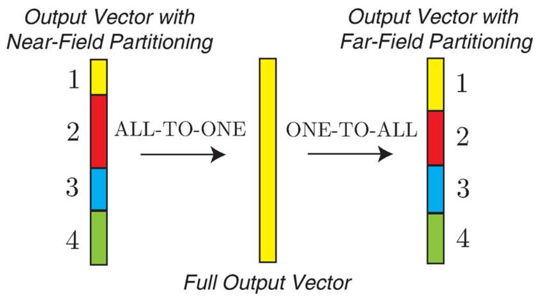

Fig. 2. Communications performed in each MVM to match the near-field and far-field partitioning schemes.

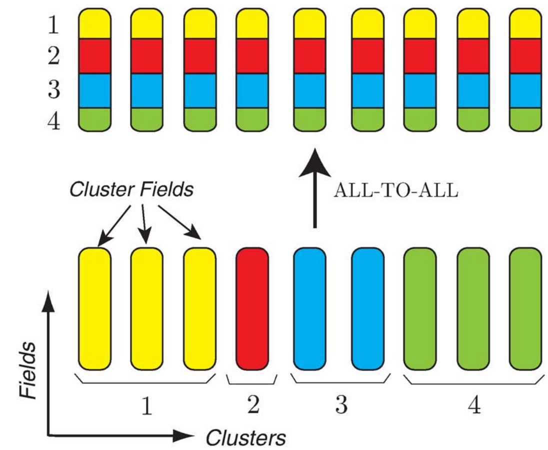

Fig. 3. All-to-all communications performed at LoD to change the far-field partitioning scheme from the distributed levels to the shared levels.

- Partitioning: For high efficiency, it is essential to distribute the tree structure among the processors with minimal duplication. This is achieved by using different partitioning strategies for the lower and higher levels of the tree structure [11]. In the lower levels (distributed levels), there are many clusters with small numbers of samples for the radiated and incoming fields. Therefore, it is appropriate to distribute the clusters in these levels by assigning each of them to a single processor. In higher levels (shared levels), however, it is easier to distribute the fields among the processors by assigning each cluster to all processors, since there are a few clusters in these levels with large numbers of samples. Calculation of the far-field interactions are organized according to the partitioning of the tree structure (far-field partitioning).
- Load-balancing: Parallelization cannot be achieved efficiently without distributing the tasks equally among the processors. We apply load-balancing for both the distributed and shared levels to improve the parallelization of the far-field interactions. For high efficiency, it is also essential to distribute the near-field interactions using a load-balancing algorithm [12].
- Communications: In parallel MLFMA, processors need to communicate with each other to transfer data. Using appropriate partitioning schemes and load-balancing algorithms significantly reduces the data traffic. However, the remaining communications must be organized carefully. For high efficiency, it is also essential to use high-speed networks to connect the processors.

In the following subsections, we provide the details of the efficient parallelization of MLFMA.

A. Setup Part

The setup part consists of preparing the near-field interactions, radiation and receiving patterns, translation operators, and preconditioners for the iterative solutions.

1) Near-Field Interactions: Near-field interactions should be distributed among the processors using a load-balancing algorithm. Considering the sparse near-field matrix, the rows $m = 1, 2, \ldots, N$ are assigned to the processors in such a way that all processors have approximately equal numbers of near-field interactions (near-field partitioning). Distributing the rows equally among the processors usually fails to provide good load-balancing, even for the solution of problems involving symmetrical geometries, such as a sphere. After distribution, the near-field interactions are calculated in each processor without any communication.

2) Radiation and Receiving Patterns: According to the far-field partitioning of the tree structure, the lowest-level clusters are distributed among the processors. Then, each processor calculates and stores the radiation and receiving patterns of the basis and testing functions included in its local tree.

3) Translation Operators: In the setup of MLFMA, each processor is tasked with calculating a set of translation operators that will be required during the MVMs. For a translation at a distributed level, where each cluster is assigned to a single processor, the operator is calculated by the processor working on the testing cluster. Due to symmetry, a translation operator can be used for many interactions in a level. Therefore, in the distributed levels, some of the translation operators are duplicated and included in more than one processor; this is allowable because of the negligible cost of the operators at the low levels. There is no duplication in the shared levels, where the fields are distributed and the translation operators are also partitioned among the processors.

4) Preconditioner: With CFIE, iterative solvers can be easily accelerated by employing simple and efficient preconditioners [15]. We use the block-diagonal preconditioner (BDP) [2] based on the self interactions of the lowest level clusters. The construction of BDP requires negligible time and memory, and its efficient parallelization is relatively easy to achieve.

B. Solution Part

For the iterative solutions, we employ Krylov subspace algorithms that are parallelized efficiently [28]. These algorithms require MVMs and the solutions of a sparse equation involving the preconditioner matrix $\bar{\boldsymbol{M}}$, i.e.,

$$\bar{\boldsymbol{Z}} \cdot \boldsymbol{x} = \boldsymbol{y} \tag{8}$$

$$\bar{\boldsymbol{M}} \cdot \boldsymbol{y} = \boldsymbol{x} \tag{9}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}$ are the input and output vectors, respectively; both are distributed according to the far-field partitioning. Before an MVM or a preconditioner solution, the partitions of the

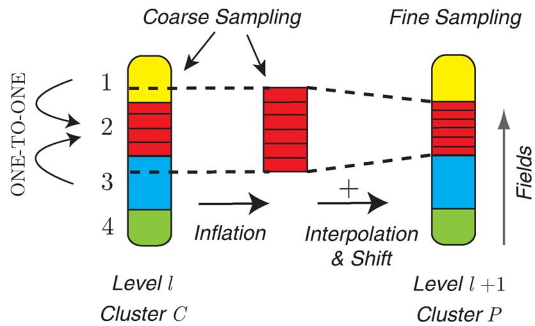

Fig. 4. Interpolations in the shared levels involving one-to-one communications.

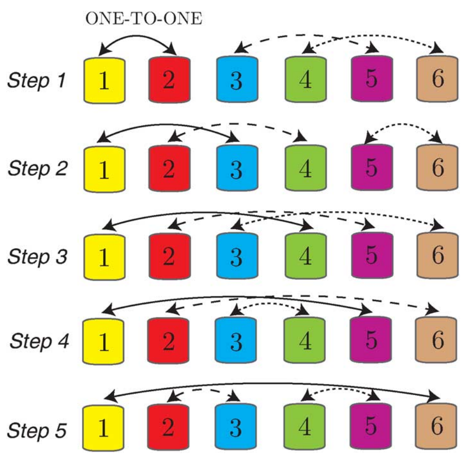

Fig. 5. Processor pairing for the translations in the distributed levels.

input vector $\boldsymbol{x}$ are combined together using the "gather" operation of MPI. Each MVM involves the use of near-field interactions, as well as the calculation of the far-field interactions via the aggregation, translation, and disaggregation stages.

1) Near-Field Stage: To match the near-field and far-field partitioning schemes during the MVMs, all-to-one and one-to-all communications are required, as depicted in Fig. 2. After the near-field computations are performed in negligible time, the partitioning of the output vector is modified for the iterative solver. The processing time for these communications is also negligible.

2) Aggregation Stage: In the highest distributed level, which we call the level of distribution (LoD), the clusters are distributed among the processors using a load-balancing algorithm that considers the combined load of all descendants (children, grandchildren, etc.) of each cluster at LoD. The combined load for a cluster is the size of the subtree attached to the cluster; we account for all descendants, each weighted by the number of field samples. The load-balancing algorithm assigns the whole branch of the tree starting at an LoD cluster to the same processor. Then, in the distributed levels, each cluster and all its subclusters are assigned to the same processor. In this way, the aggregation stage up to LoD can be performed without any communication. At LoD, the partitioning scheme is changed by employing an all-to-all communication, as shown in Fig. 3. For each cluster, the samples of the radiated field stored in a processor is distributed among all processors. In the shared levels

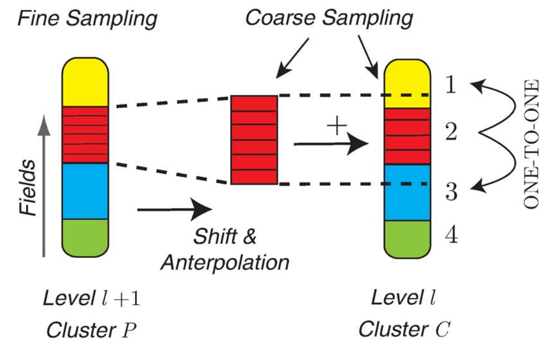

Fig. 6. Anterpolations (transpose interpolations) in the shared levels involving one-to-one communications.

TABLE II
COMMUNICATIONS REQUIRED IN THE MATRIX-VECTOR MULTIPLICATIONS BY PARALLEL MLFMA

PART	COMMUNICATION
Near-Field Stage	All-to-One and One-to-All
Distributed Aggregation/Disaggregation	None
Aggregation/Disaggregation in LoD	All-to-All
Shared Aggregation/Disaggregation	One-to-One
Distributed Translation	One-to-One
Shared Translation	None

above LoD, $(T_l+1)\times 2(T_l+1)$ samples on the θ - ϕ space are partitioned along the θ direction.

From LoD to the highest level L, the aggregation stage involves one-to-one communications that are required for the interpolation of the fields. This is illustrated in Fig. 4, where an interpolation is performed on the samples of cluster C. As an example, only the interpolation in processor 2 is depicted although similar operations are also performed in the other processors. To compute the data at each sample in the fine grid, a set of samples are used in the coarse grid. Even though a local interpolation method is used, some of those coarse samples may be located in other processors. Therefore, one-to-one communications are performed to provide the required data (inflation). After the data is prepared, interpolation and shifting operations are performed to include the contribution of the cluster C in the radiated field of its parent cluster P.

We note that the communications in the shared levels are mainly required between the processors located "close to each other." In other words, the processor with index n_p requires data from its "neighbors," i.e., n_p-1 and n_p+1. On the other hand, depending on the partitioning and the number of interpolation points, more data might be required from other processors next to the neighbors. We apply a load-balancing algorithm to distribute the fields appropriately so that the amount of the data transferred among all processors is minimized. However, as the number of processes increases and the fields are distributed over many processors, dense one-to-one communications cannot be avoided; this may reduce the efficiency of the parallelization.

Finally, for each problem, we carefully choose the number of distributed and shared levels by an optimization. For this purpose, we assign LoD to a series of possible levels and monitor

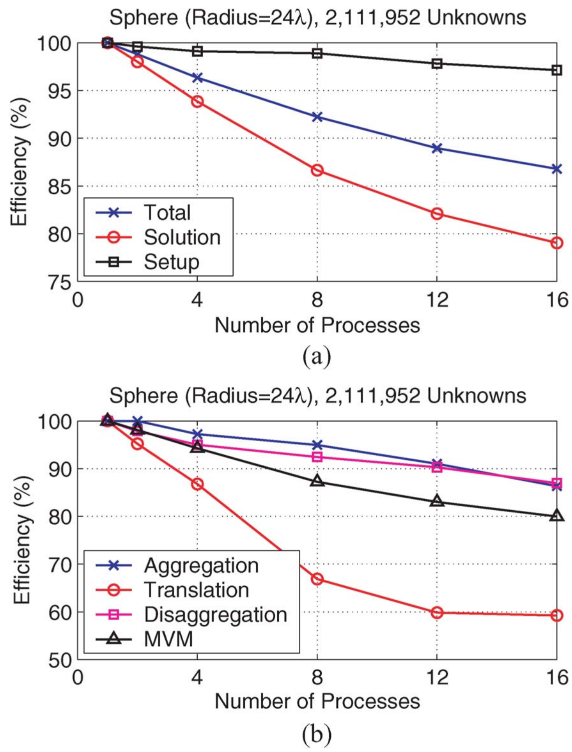

Fig. 7. Parallelization efficiency for the solution of a scattering problem involving a sphere of radius 24λ discretized with 2 111 952 unknowns.

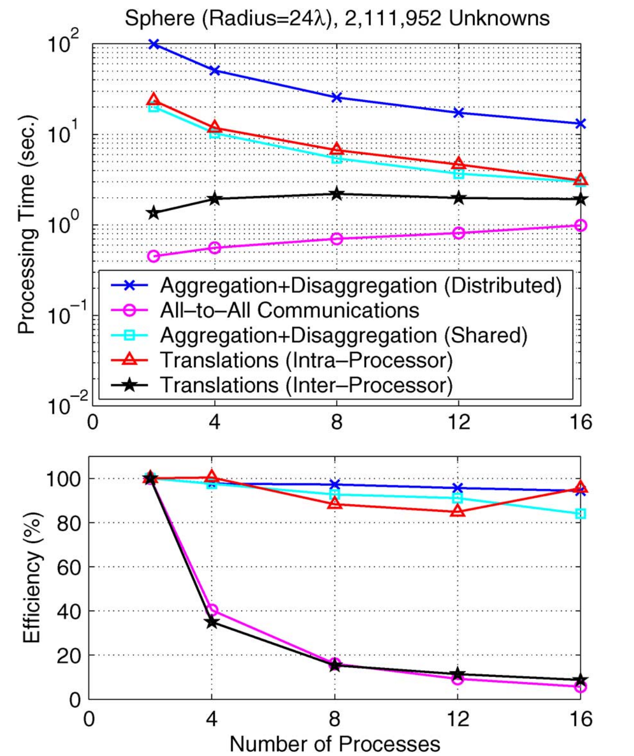

Fig. 8. Processing time and parallelization efficiency for various categorized parts of the MVMs for the solution of a scattering problem involving a sphere of radius 24λ discretized with 2 111 952 unknowns.

the distribution of the clusters and the fields. For some of the levels (higher levels), distribution of the fields is better than the distribution of the clusters, i.e., samples of the fields can be partitioned evenly among the processors, but not the clusters. For the others (lower levels), however, clusters can be partitioned easily, while it is difficult to partition the fields among the processors. Then, we choose LoD such that distributing the fields (clusters) is more preferable for all levels above (below) LoD. The choice of LoD depends on the tree structure (hence the geometry of the target) as well as the number of processors. However, our measurements show that, for a given problem, LoD is insensitive to the latter parameter if only a small number (e.g., 2 to 16) of processors are employed.

3) Translation Stage: The translation stage is one of the most critical parts for the efficiency of the parallelization. This is because dense one-to-one communications are required between the processors for the translations in the distributed levels. In general, each processor sends some data (radiated fields) to all other processors. We organize these communications using a communication map, which consists of interaction layers to match the processors. For p processors, it can be shown that the communications can be achieved in $p-1$ steps, as depicted in Fig. 5 for a 6-process case. After the processors are paired, the following operations are performed on the receiver and sender sides.

- The sender and receiver determine the cluster-cluster interactions involving the basis clusters on the sender side and testing clusters on the receiver side.
- The radiated fields of the basis clusters are sent one by one.
- When the radiated field of a basis cluster is received by the receiver, all of the translations involving this basis cluster and the testing clusters owned by the receiver are performed. This ensures that the same data is not transferred more than once.

To improve the efficiency of the translations, we use non-blocking send and receive operations of MPI to transfer the data. In the shared levels, all the translations are performed without any communication since the fields are distributed among the processors and a processor is assigned to the same portion of the radiated or incoming fields for all clusters.

4) Disaggregation Stage: The disaggregation stage is generally the inverse of the aggregation stage. The incoming fields are calculated at the center of each cluster from the top of the tree structure to the lowest level using the anterpolation and shift operations. For a cluster in level $l < L$, the incoming field is the combination of the translated field from the far-field clusters and the incoming field to the center of its parent. In the shared levels, anterpolation produces samples in the coarse grid, some of which should be sent to the "neighboring" processors. This is illustrated in Fig. 6, where processor 2 performs the anterpolation operation on the samples of cluster P for its subcluster C. Some of the resulting data in the coarse grid is used locally, while the rest are sent to other processors, i.e., exactly the reverse of the interpolation. As the disaggregation operation proceeds down to LoD, the partitioning is changed via an all-to-all communication. Then, the disaggregation is performed from LoD to the lowest level without any communication. In the lowest level, each processor performs the angular integrations and produces a partition of the output vector $\boldsymbol{y}$.

To sum up, Table II lists the communications required at each stage of the MVMs.

TABLE III
SOLUTIONS OF LARGE SPHERE PROBLEMS WITH MLFMA PARALLELIZED INTO 16 PROCESSES

Diameter	160λ	192λ	220λ
Unknowns	23,405,664	33,791,232	41,883,648
CLUSTERING			
Number of Levels	9		
Smallest Cluster Size	0.16λ	0.19λ	0.21λ
Number of Clusters	5,769,254	5,904,951	5,975,507
Lowest-Level Clusters	4,225,343	4,344,205	4,405,952
Near-Field Sparsity	3.27×10^{-6}	3.27×10^{-6}	3.28×10^{-6}
Truncation Number	5 to 457	6 to 546	6 to 623
Tree Size	1.68×10^{9}	2.29×10^{9}	2.51×10^{9}
PROCESSING TIME (Intel Xeon 5355 processors connected via an Infiniband network)			
Setup Times (minutes)	94	183	274
BiCGStab Iterations	17	21	19
MVM Time (seconds)	270	372	441
Solution Time (minutes)	155	264	290
MEMORY USAGE			
Translation Operators	1.5 GB	2.1 GB	2.7 GB
Radiation and Receiving Patterns	25.8 GB	57.4 GB	71.2 GB
Near-Field Interactions	13.4 GB	27.8 GB	42.9 GB
BD Preconditioner	1.3 GB	2.6 GB	4.0 GB
MVM	40.3 GB	55.9 GB	65.3 GB

IV. RESULTS

First, we demonstrate the efficiency of MLFMA parallelization for the solution of a scattering problem involving a sphere of radius 24λ. The problem is discretized with 2 111 952 unknowns and solved on a cluster of Intel Xeon processors connected via an Infiniband network. Fig. 7 depicts the efficiency (with respect to the solution with a single processor) when the solution is parallelized into 2, 4, 8, 12, and 16 processes. The parallelization efficiency is defined as

$$\varepsilon_p = \frac{T_1}{pT_p} \tag{10}$$

where T_p is the processing time of the solution with p processes. Fig. 7(a) shows that the overall efficiency (setup and iterative solution) is above 85% when the number of processes is 16. In this case, efficiency ratios for the setup and the solution parts are about 97% and 80%, respectively. We observe in Fig. 7(a) that the setup part is parallelized very efficiently, since this part is communication-free and the computations (especially the near-field interactions) are perfectly distributed to the processors using a load-balancing algorithm.

In Fig. 7(b), we present the parallelization efficiency for the aggregation, translation, and disaggregation stages, in addition to the overall efficiency for the MVMs. The near-field stage is not considered because of its negligible time. We observe that aggregation and disaggregation stages are parallelized with about 87% efficiency, while efficiency for the translation stage is 59% for the 16-process case. To further investigate the parallelization, Fig. 8 presents processing time and efficiency (with respect to the solution with 2 processors) for various categorized parts of the MVMs. Our observations are as follows.

- Aggregation and disaggregation stages in the distributed levels ($l = 1, 2, 3, 4$ for this problem) constitute the significant part of the processing time of MVM. These stages are perfectly parallelized, thanks to the load-balancing algorithm for distributed levels.
- The parallelization efficiency of the aggregation and disaggregation stages in the shared levels (from $l = 5$ to $l = L = 7$ in this problem) is also quite high. However, the efficiency drops to about 80% for the 16-process case. This is due to the increasing amount of one-to-one communications for interpolations and anterpolations.
- Parallelization efficiency of the communication-free (intraprocessor) translations is in the 80%–100% range. All of the translations in the shared levels and some of those in the distributed levels are communication-free.
- Translations that are performed with communications (interprocessor translations) and all-to-all communications performed at LoD exhibit reduced efficiency as the number of processes increases. Since they take longer processing time, the interprocessor translations affect the overall efficiency more than the all-to-all communications.

In general, interprocessor translations are the bottleneck of the parallelization. Since these translations are performed in the distributed levels, their negative contributions can be minimized by increasing the number of shared levels. However, aggregation and disaggregation in low levels cannot be performed efficiently by partitioning the coarsely sampled fields. As discussed in Section III, we carefully determine the number of distributed and shared levels to optimize the parallelization efficiency for the solution of each problem.

In Table III, we present the solutions of very large scattering problems involving spheres of radii 80λ, 96λ, and 110λ, which are discretized with 23 405 664, 33 791 232, and 41 883 648 unknowns, respectively. For all three problems, 9-level MLFMA is employed and parallelized into 16 processes. The numbers of distributed and shared levels are 6 and 3, respectively. Using a top-down strategy, the cluster size in the lowest level is 0.16λ–0.21λ. Each of the tree structures contains about six million clusters and most of them are used in the lowest level. The number of near-field interactions increases with the problem size and the sparsity of the near-field matrix is almost constant. The near-field interactions are calculated with 1% error. The smallest and largest truncation numbers are also listed in Table III when the far-field interactions are calculated with two digits of accuracy.

Table III shows that the setup time increases proportionally to N^2 since the sparsity of the near-field matrix is constant and the number of near-field interactions is proportional to N^2. On the other hand, the processing time for the MVMs, which is related to the tree size, increases more slowly than $O(N \log N)$. As discussed in Section II, these local deviations from the asymptotical estimates are expected depending on the clustering technique used for the tree structure. As depicted in Fig. 1, the tree size and the number of near-field interactions oscillate around the $O(N \log N)$ and $O(N)$ curves, respectively. Local variations of these quantities corresponding to the three large problems in Table III are magnified in the inset of Fig. 1. We observe that the tree size (hence the computational requirements for the MVMs) increases slower than the asymptotical estimate of $O(N \log N)$. On the other hand, due to a top-down clustering scheme, the number of near-field interactions (hence the computational requirements for the near-field part) grows faster than $O(N)$. We emphasize that this behavior is local and depends on the strategy to construct the tree structure, the overall complexity of MLFMA is still $O(N \log N)$.

We also observe in Table III that the maximum number of biconjugate-gradient-stabilized (BiCGStab) iterations to reduce the residual error below 10^{-3} is 21. Using the BDP, iterative solution of the 42-million-unknown problem requires only 290 min, while each MVM is performed in 441 s. Table III also lists the total memory usage for different parts of the algorithm using the single-precision representation for the complex numbers.

In Fig. 9, we further present the details of the solution of the 23-million-unknown problem involving a sphere of radius 80λ. In Fig. 9(a), the total processing time is depicted for all processes from 1 to 16. After the input and the clustering part$^{(1)}$, computations of the translation operators$^{(2)}$ and the radiation/receiving patterns$^{(4)}$ require negligible time. Calculation of the near-field interactions$^{(3)}$ dominates the setup time, which is about 94 min. Then the solution part$^{(5)}$, involving a total of 34 MVMs, is performed in about 155 min. The processing time for a MVM is depicted in Fig. 9(b), including the near-field stage$^{(1)}$, aggregation/disaggregation in the distributed levels$^{(2,9)}$, all-to-all communications$^{(3,8)}$, aggregation/disaggregation in the shared levels$^{(4,7)}$, communication-free (intraprocessor) translations$^{(5)}$, and interprocessor translations$^{(6)}$. The most problematic parts in terms of parallelization efficiency, i.e., all-to-all communications and interprocessor translations, require negligible time compared to other parts of the MVM. This is commonly observed with large-sized problems and supports the conclusion that the parallelization efficiency for a fixed number of processes usually increases as the problem size grows.

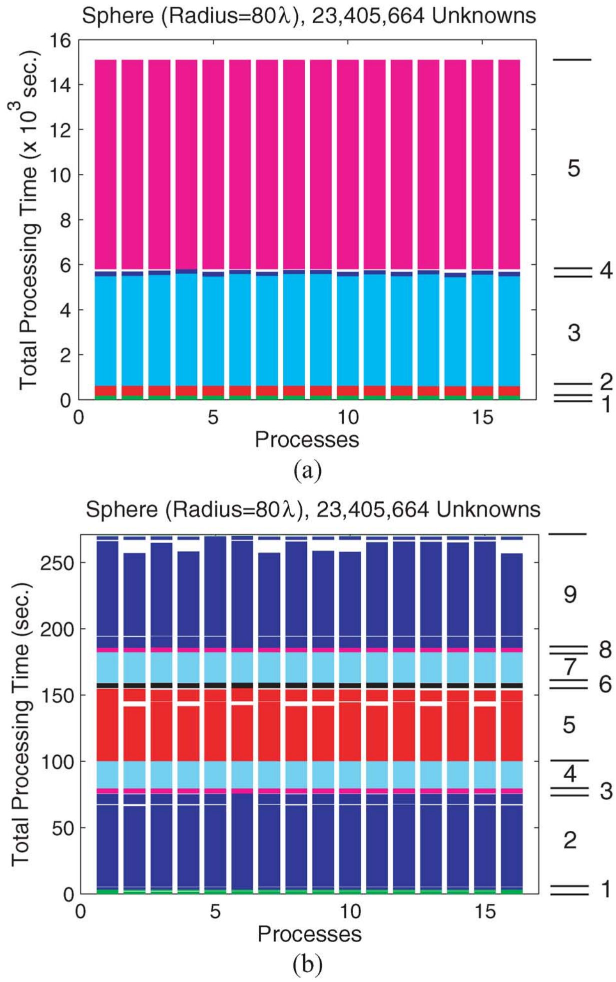

Fig. 9. Time diagrams for the solution of a scattering problem involving a sphere of radius 80λ discretized with 23 405 664 unknowns. (a) Overall time includes the input and clustering parts$^{(1)}$, calculation of the translation matrices$^{(2)}$, calculation of the near-field interactions$^{(3)}$, calculation of the radiation and receiving patterns$^{(4)}$, and the iterative solution$^{(5)}$. (b) Matrix-vector multiplications include the near-field stage$^{(1)}$, aggregation in the distributed levels$^{(2)}$, all-to-all communications in LoD$^{(3,8)}$, aggregation in the shared levels$^{(4)}$, translations without communications$^{(5)}$, translations with communications$^{(6)}$, disaggregation in the shared levels$^{(7)}$, and disaggregation in the distributed levels followed by the receiving operation$^{(9)}$. In the diagrams, white areas correspond to waits before the operations that require synchronization.

To present the accuracy of the solutions, Fig. 10 depicts the normalized bistatic radar cross section (RCS/λ^2) values in decibels (dB) for a sphere of radius 110λ discretized with 41 883 648 unknowns. We believe this is the solution of the largest integral-equation problem ever reported. Solutions of

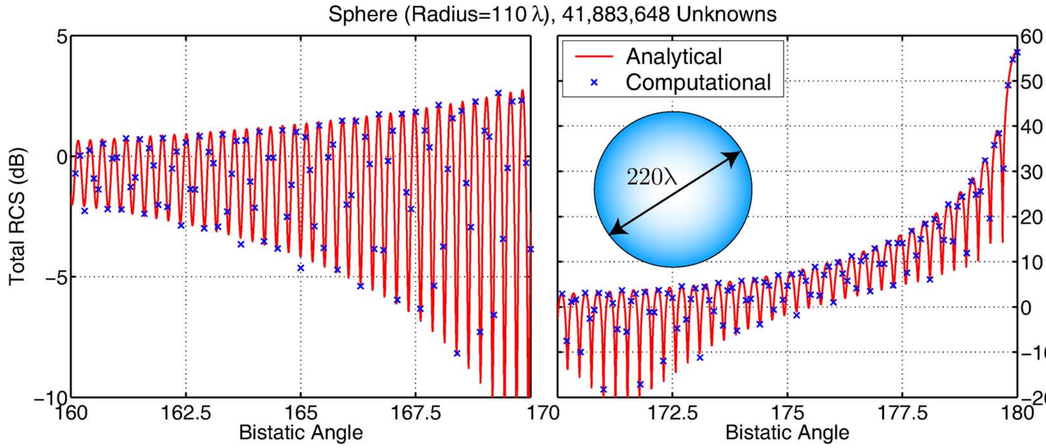

Fig. 10. Bistatic RCS (in dB) of a sphere of radius 110λ discretized with 41 883 648 unknowns from 160° to 180°, where 180° corresponds to the forward-scattering direction.

integral-equation problems with 20 million and 33 million unknowns were reported in [8] and [13], respectively. Analytical values obtained by a Mie-series solution is plotted as a reference from 160° to 180°, where 180° corresponds to the forward-scattering direction. Fig. 10 shows that the computational values sampled at 0.1° are in agreement with the analytical curve. For more quantitative information, we define a relative error as

$$e_R = \frac{||\boldsymbol{A} - \boldsymbol{C}||_2}{||\boldsymbol{A}||_2} \tag{11}$$

where $\boldsymbol{A}$ and $\boldsymbol{C}$ are the analytical and computational RCS values, respectively, $||.||_2$ is the l^2-norm defined as

$$||\boldsymbol{x}||_2 = \sqrt{\sum_{s=1}^{S} |\boldsymbol{x}[s]|^2} \tag{12}$$

and S is the number of samples. The relative error is 3.87%, 4.67%, and 4.67% in the 160°–170°, 170°–180°, and 0°–180° ranges, respectively. We note that the relative error in the RCS values is about 5%, although we calculate the near-field and far-field interactions with 1% error. The extra error is due to the low-order discretization of CFIE. For the same discretization of a scattering problem with the RWG functions, MFIE (thus, CFIE) is consistently inaccurate to calculate the scattered fields compared to EFIE, even MFIE (and CFIE) is better conditioned than EFIE [29]. A remedy to this accuracy problem is to use higher-order basis functions, such as the linear-linear basis functions discussed in [17].

Finally, we present the solution of a real-life problem involving the Flamme, which is a stealth airborne target, as detailed in [30]. The scattering problem is solved at 16 GHz and the maximum dimension of the Flamme is 6 m, corresponding to 320λ. Using $\lambda/10$ triangulation, the problem is discretized with 24 782 400 unknowns. Fig. 11 presents the bistatic RCS values in dBm^2 when the target is illuminated by a plane wave propagating in the x-y plane at a 30° angle from the x axis (from $\phi = 30°$). Both θ and ϕ polarizations are considered.

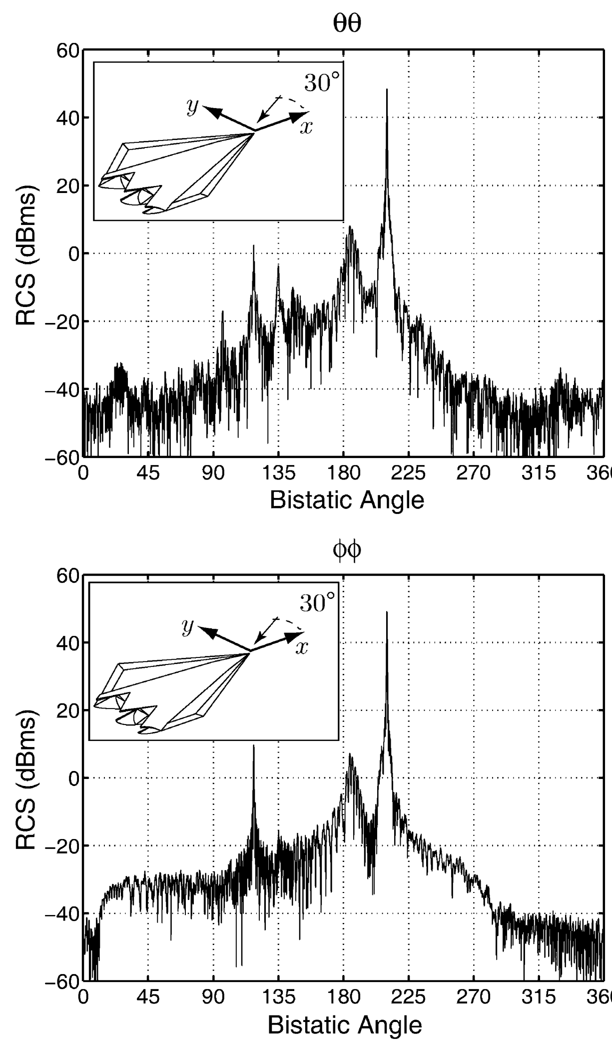

Fig. 11. Bistatic RCS (in dBm^2) of the stealth airborne target Flamme at 16 GHz. Maximum dimension of the Flamme is 6 m corresponding to 320λ. The target is illuminated by a plane wave propagating in the x-y plane at a 30° angle from the x axis, as also depicted in the inset.

The copolar RCS values are plotted on the x-y plane as a function of the bistatic angle ϕ. In the plots, 30° and 210° correspond to the back-scattering and forward-scattering directions, respectively. Solution of this problem is performed by a 10-level MLFMA (6 distributed and 4 shared levels) parallelized into 16 processes. After the setup, which takes about 104 min, the problem is solved twice (for two polarizations) in about 490 min. Using BiCGStab and BDP, the numbers of iterations to reduce the residual error below 10^{-3} are 42 and 35, respectively, for the θ and the ϕ polarizations of the plane-wave excitation. Both near-field and far-field interactions are calculated with 1% error and the total memory usage is 139 GB using the single-precision representation.

V. Concluding Remarks

In this paper, we consider fast and accurate solutions of large-scale scattering problems discretized with tens of millions of unknowns using a parallel MLFMA implementation. We investigate the parallelization of MLFMA and improve the efficiency of the implementations. Some of the major steps for the efficient parallelization of MLFMA are as follows.

- Distribute the near-field interactions equally among the processors using a load-balancing algorithm.
- Determine the shared and distributed levels appropriately by choosing an optimal LoD.
- Distribute the clusters in LoD among the processors by considering the combined load of all descendants of each cluster.
- Assign each cluster and its subclusters to the same processor for the levels below LoD.
- Distribute the samples of the fields among the processors using a load-balancing algorithm (to reduce one-to-one communications) for the levels above LoD.
- Use a communication map to pair the processors for the translations in the distributed levels. Transfer the required data using nonblocking send and receive operations.

We demonstrate the accuracy of our implementations by considering a canonical scattering problem involving a sphere of radius 110λ discretized with 41 883 638 unknowns. To the best of our knowledge, this is the largest integral-equation problem ever solved.[2] In addition to the solution of various extremely large canonical problems, we also demonstrate the effectiveness of our implementation on a real-life problem involving the Flamme geometry with a size larger than 300λ.

References

[1] A. J. Poggio and E. K. Miller, "Integral equation solutions of three-dimensional scattering problems," in *Computer Techniques for Electromagnetics*, R. Mittra, Ed. Oxford, U.K.: Pergamon, 1973, ch. 4.

[2] J. Song, C.-C. Lu, and W. C. Chew, "Multilevel fast multipole algorithm for electromagnetic scattering by large complex objects," *IEEE Trans. Antennas Propag.*, vol. 45, no. 10, pp. 1488–1493, Oct. 1997.

[3] S. 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," in *Proc. IEEE Antennas Propag. Soc. Int. Symp.*, 1999, vol. 1, pp. 636–639.

[4] S. Velamparambil, J. M. Song, and W. C. Chew, "A portable parallel multilevel fast multipole solver for scattering from perfectly conducting bodies," in *Proc. IEEE Antennas Propag. Soc. Int. Symp.*, 1999, vol. 1, pp. 648–651.

[2]Prior to the publication of this paper, the authors reported in January 2008 the solution of a larger scattering problem involving a sphere of radius 150λ discretized with 85 148 160 unknowns.

[5] K. C. Donupedi, J.-M. Jin, S. Velamparambil, J. Song, and W. C. Chew, "A higher order parallelized multilevel fast multipole algorithm for 3-D scattering," *IEEE Trans. Antennas Propag.*, vol. 49, no. 7, pp. 1069–1078, Jul. 2001.

[6] S. Velamparambil, W. C. Chew, and M. L. Hastriter, "Scalable electromagnetic scattering computations," in *Proc. IEEE Antennas Propag. Soc. Int. Symp.*, 2002, vol. 3, pp. 176–179.

[7] S. Velamparambil, W. C. Chew, and J. Song, "10 million unknowns: Is it that big?," *IEEE Antennas Propag. Mag.*, vol. 45, no. 2, pp. 43–58, Apr. 2003.

[8] M. L. Hastriter, "A study of MLFMA for large-scale scattering problems," Ph.D. dissertation, Univ. Illinois at Urbana-Champaign, , 2003.

[9] G. Sylvand, "Performance of a parallel implementation of the FMM for electromagnetics applications," *Int. J. Numer. Meth. Fluids*, vol. 43, pp. 865–879, 2003.

[10] J. Dong, S. L. Chai, and J. J. Mao, "An automatic load-balanced parallel multilevel fast multipole method for large scale electromagnetic scattering problem," in *Proc. Asia-Pacific Conf.*, 2005.

[11] S. Velamparambil and W. C. Chew, "Analysis and performance of a distributed memory multilevel fast multipole algorithm," *IEEE Trans. Antennas Propag.*, vol. 53, no. 8, pp. 2719–2727, Aug. 2005.

[12] Ö. Ergül and L. Gürel, "Efficient parallelization of multilevel fast multipole algorithm," in *Proc. Europ. Conf. Antennas Propag. (EuCAP)*, 2006, no. 350094.

[13] L. Gürel and Ö. Ergül, "Fast and accurate solutions of integral-equation formulations discretised with tens of millions of unknowns," *Electron. Lett.*, vol. 43, no. 9, pp. 499–500, Apr. 2007.

[14] D. R. Wilton and J. E. Wheeler, III, "Comparison of convergence rates of the conjugate gradient method applied to various integral equation formulations," in *Progress in Electromagn. Res. PIER 05*, 1991, pp. 131–158.

[15] L. Gürel and Ö. Ergül, "Comparisons of FMM implementations employing different formulations and iterative solvers," in *Proc. IEEE Antennas Propag. Soc. Int. Symp.*, 2003, vol. 1, pp. 19–22.

[16] L. Gürel and Ö. Ergül, "Extending the applicability of the combined-field integral equation to geometries containing open surfaces," *IEEE Antennas Wireless Propag. Lett.*, vol. 5, pp. 515–516, 2006.

[17] Ö. Ergül and L. Gürel, "Linear-linear basis functions for MLFMA solutions of magnetic-field and combined-field integral equations," *IEEE Trans. Antennas Propag.*, vol. 55, no. 4, pp. 1103–1110, Apr. 2007.

[18] J. R. Mautz and R. F. Harrington, "H-field, E-field, and combined field solutions for conducting bodies of revolution," *AEÜ*, vol. 32, no. 4, pp. 157–164, Apr. 1978.

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

[20] Ö. Ergül and L. Gürel, "The use of curl-conforming basis functions for the magnetic-field integral equation," *IEEE Trans. Antennas Propag.*, vol. 54, no. 7, pp. 1917–1926, Jul. 2006.

[21] R. Coifman, V. Rokhlin, and S. Wandzura, "The fast multipole method for the wave equation: A pedestrian prescription," *IEEE Antennas Propag. Mag.*, vol. 35, no. 3, pp. 7–12, Jun. 1993.

[22] S. Koc, J. M. Song, and W. C. Chew, "Error analysis for the numerical evaluation of the diagonal forms of the scalar spherical addition theorem," *SIAM J. Numer. Anal.*, vol. 36, no. 3, pp. 906–921, 1999.

[23] Ö. Ergül and L. Gürel, "Enhancing the accuracy of the interpolations and anterpolations in MLFMA," *IEEE Antennas Wireless Propag. Lett.*, vol. 5, pp. 467–470, 2006.

[24] A. Brandt, "Multilevel computations of integral transforms and particle interactions with oscillatory kernels," *Comp. Phys. Comm.*, vol. 65, pp. 24–38, Apr. 1991.

[25] W. C. Chew, J.-M. Jin, E. Michielssen, and J. Song, *Fast and Efficient Algorithms in Computational Electromagnetics*. Boston, MA: Artech House, 2001.

[26] J. Song and W. C. Chew, "Interpolation of translation matrix in MLFMA," *Microw. Opt. Technol. Lett.*, vol. 30, no. 2, pp. 109–114, Jul. 2001.

[27] Ö. Ergül and L. Gürel, "Optimal interpolation of translation operator in multilevel fast multipole algorithm," *IEEE Trans. Antennas Propag.*, vol. 54, no. 12, pp. 3822–3826, Dec. 2006.

[28] S. Balay, K. Buschelman, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang, *PETSc Users Manual*. Argonne, IL: Argonne National Lab., 2004.

[29] Ö. Ergül and L. Gürel, "Investigation of the inaccuracy of the MFIE discretized with the RWG basis functions," in *Proc. IEEE Antennas Propag. Soc. Int. Symp.*, 2004, vol. 3, pp. 3393–3396.

[30] L. Gürel, H. Bağcı, J. C. Castelli, A. Cheraly, and F. Tardivel, "Validation through comparison: Measurement and calculation of the bistatic radar cross section (BRCS) of a stealth target," *Radio Sci.*, vol. 38, no. 3, Jun. 2003.

Özgür Ergül (S'98) was born in Yozgat, Turkey, in 1978. He received the B.Sc. and M.S. degrees in electrical and electronics engineering from Bilkent University, Ankara, Turkey, in 2001 and 2003, respectively, where he is currently working toward the Ph.D. degree.

Since 2001, he has served as a Teaching and Research Assistant in the Department of Electrical and Electronics Engineering, Bilkent University. He has been with the Computational Electromagnetics Group at Bilkent University from 2000 to 2005 and with the Computational Electromagnetics Research Center (BiLCEM) since 2005. His research interests include fast and accurate algorithms for the solutions of large and complicated structures, integral equations, parallel programming, and iterative techniques.

Mr. Ergül was a recipient of the 2007 IEEE Antennas and Propagation Society Graduate Fellowship and the 2007 Leopold B. Felsen Award for Excellence in Electrodynamics. He is the Secretary of Commission E (Electromagnetic Noise and Interference) of URSI Turkey National Committee. His academic endeavors are supported by the Scientific and Technical Research Council of Turkey (TUBITAK) through a Ph.D. scholarship.

Levent Gürel (S'87–M'92–SM'97) received the B.Sc. degree from the Middle East Technical University (METU), Ankara, Turkey, in 1986, and the M.S. and Ph.D. degrees from the University of Illinois at Urbana-Champaign (UIUC) in 1988 and 1991, respectively, all in electrical engineering.

He joined the Thomas J. Watson Research Center of the International Business Machines Corporation, Yorktown Heights, NY, in 1991, where he worked as a Research Staff Member on the electromagnetic compatibility (EMC) problems related to electronic packaging, on the use of microwave processes in the manufacturing and testing of electronic circuits, and on the development of fast solvers for interconnect modeling. Since 1994, he has been a faculty member with the Department of Electrical and Electronics Engineering, Bilkent University, Ankara, where he is currently a Professor. He was a Visiting Associate Professor with the Center for Computational Electromagnetics (CCEM) of the UIUC for one semester in 1997. He returned to the UIUC as a Visiting Professor during 2003–2005, and as an Adjunct Professor after 2005. He founded the Computational Electromagnetics Research Center (BiLCEM) at Bilkent University in 2005, where he is serving as the Director. His research interests include the development of fast algorithms for computational electromagnetics (CEM) and the application thereof to scattering and radiation problems involving large and complicated scatterers, antennas and radars, frequency-selective surfaces, high-speed electronic circuits, optical and imaging systems, nanostructures, and metamaterials. He is also interested in the theoretical and computational aspects of electromagnetic compatibility and interference analyses. Ground-penetrating radars and other subsurface scattering applications are also among his research interests. Since 2006, his research group has been breaking several world records by solving extremely large integral-equation problems, the largest involving as many as 85 million unknowns.

Prof. Gürel's accomplishments include two prestigious awards from the Turkish Academy of Sciences (TUBA) in 2002 and the Scientific and Technical Research Council of Turkey (TUBITAK) in 2003. He served as the Chairman of the AP/MTT/ED/EMC Chapter of the IEEE Turkey Section in 2000–2003. He founded the IEEE EMC Chapter in Turkey in 2000. He served as the Co-chairman of the 2003 IEEE International Symposium on Electromagnetic Compatibility. He is a member of the General Assembly of the European Microwave Association, a member of the USNC of the International Union of Radio Science (URSI), and the Chairman of Commission E (Electromagnetic Noise and Interference) of URSI Turkey National Committee. He organized and served as the General Chair of the CEM'07 Computational Electromagnetics International Workshop in 2007. He is currently serving as Associate Editor of the IEEE ANTENNAS AND WIRELESS PROPAGATION LETTERS and *Radio Science* .