Fast and Accurate Solutions of Large-Scale Scattering Problems with Parallel Multilevel Fast Multipole Algorithm[†]

Özgür Ergül[1] and Levent Gürel*[1,2]

[1] Department of Electrical and Electronics Engineering
[2] Computational Electromagnetics Research Center (BiLCEM)
Bilkent University, TR-06800, Bilkent, Ankara, Turkey
E-mail: {ergul,lgurel}@ee.bilkent.edu.tr

Introduction

We consider the fast and accurate solution of large-scale scattering problems obtained by integral-equation formulations for conducting surfaces. By employing a parallel implementation of the multilevel fast multipole algorithm (MLFMA) [1] on relatively inexpensive platforms, we are able to solve problems with tens of millions of unknowns. Specifically, we report the solution of a scattering problem with 33,791,232 unknowns, which is even larger than the 20-million unknown problem reported recently [2]. Indeed, this 33-million-unknown problem is the largest integral-equation problem solved in computational electromagnetics, to the best of our knowledge.

MLFMA reduces the complexity of the matrix-vector multiplications related to an $N \times N$ dense matrix equation from $O(N^2)$ to $O(N \log N)$. Therefore, it provides the solution of large radiation and scattering problems in electromagnetics. However, it is also desirable to parallelize MLFMA for the solutions of many real-life problems that are not easily handled by sequential implementations. On the other hand, due to its complex structure, parallelization of MLFMA is not trivial. Communications between the processors and duplications of the computations reduce the efficiency of the parallelization. In this manner, our implementations involve various parallelization strategies, load-balancing algorithms, optimizations, and many other techniques to improve the efficiency without sacrificing the accuracy. In this paper, we present our efforts to develop a robust solver for the solution of large scattering problems both efficiently and accurately.

Data Structures and Primary Operations in Parallel MLFMA

The operations performed in MLFMA can be divided into setup and solution parts. In the setup of the implementation, the required data for the matrix-vector multiplications are computed and stored in the memory. In the second part, the solution is performed by an iterative algorithm, where the required matrix-vector multiplications are achieved in $O(N \log N)$ time using $O(N \log N)$ memory.

Setup Part: The data prepared during the setup of the program includes the tree structure and clustering of MLFMA, near-field interactions that are calculated directly, radiation and receiving patterns of the basis and testing functions, and translation matrices for the cluster-cluster interactions.

Input and Clustering: In our implementations, we apply a discretization on the geometry by using planar triangles. To obtain accurate results, the size of the triangles are set to approximately $\lambda/10$, where λ is the wavelength. Then, the unknown surface current density is expanded in a series of Rao-Wilton-Glisson (RWG) [3] or linear-linear (LL) [4] basis functions. The triangulation data is read from a file and the tree structure formed in $O(N)$ time. The geometry and clustering data with size $O(N)$ is stored in the memory to be used in the next stages.

Calculation of Near-Field Interactions: In MLFMA, there are $O(N)$ near-field interactions that are calculated directly and stored in the memory. Calculation of these interactions require the evaluation of double integrals, which can be performed by using low-order Gaussian quadratures with the aid of singularity extraction methods [5]–[7]. In the parallel implementation, we apply a load-balancing algorithm to distribute the near-field interactions equally among the processors [8].

Calculation of Radiation and Receiving Patterns: For each basis and testing functions, integrals in the form of

$$\boldsymbol{I} = \int_{S_m} d\boldsymbol{r} \exp{(ik\hat{\boldsymbol{k}} \cdot \boldsymbol{r})} \boldsymbol{f}_m(\boldsymbol{r}) \tag{1}$$

are need to be evaluated [1], where k is the wavenumber and $\boldsymbol{f}_m$ represents distribution of the basis or testing function with the spatial support of S_m. In (1), $\hat{\boldsymbol{k}}$ represents the angular directions according to the sampling scheme for the lowest-level clusters. For both RWG and LL functions, the integrals in the form of (1) can be calculated analytically. The calculation of the radiation and receiving patterns requires $O(N)$ time and the memory requirement is also $O(N)$. In the parallelization of the patterns, we distribute the the basis and testing functions among the processors according to the far-field partitioning, which is different from the distribution of the near-field interactions for efficiency.

Calculation of Translation Matrices: To evaluate the cluster-cluster interactions, translation operators are formed and stored in the memory. For each level, there are $O(1)$ different translation operators using the symmetrical properties of the cubic clustering scheme [9]. In addition, we use interpolation methods to efficiently fill the translation matrices in $O(N)$ time [10]. For high accuracy, the truncation numbers are selected by using the excess bandwidth formula [11] for the worst case scenario according to one-box-buffer scheme. In the parallel implementation, only the required translation operators are calculated and stored in each processor. Therefore, for the upper levels of the tree structure, where the fields are distributed among the processors [12], the translation matrices are partitioned and also distributed among the processors.

Solution Part: In the solution part, each matrix-vector multiplication includes several operations such as aggregation, translation, and disaggregation that are performed in a multilevel scheme, as well as the near-field interactions.

Near-Field Interactions: The near-field interactions that are stored in the memory are used to compute the partial matrix-vector multiplication in $O(N)$ time. Since the near-field and far-field partitioning schemes are different, an all-to-all communication is required to switch between the two partitionings after the near-field interactions are performed [8].

Agggregation: During the aggregation phase, radiation patterns of the clusters are calculated from the bottom to the top of the tree structure. Using a local interpolation scheme, aggregation is achieved in $O(N \log N)$ time using $O(N \log N)$ memory to store the samples of the radiation patterns. To parallelize the aggregation process, we choose a level (LoD: Level of Distribution) to distribute the clusters among the processors using a load-balancing algorithm. Then, for the lower levels (below LoD), each sub-cluster is assigned to the same processor as its parent cluster. This way, the aggregation operations for the lower levels of the tree structure do not require any communication. On the other hand, in the upper levels of the tree structure (above LoD), radiation and receiving patterns of the clusters are distributed among the processors to improve the parallelization [12]. In this scheme, one-to-one communications are required between the processors to perform the interpolations accurately. To further improve the parallelization, we reduce the amount of these communications by carefully partitioning the patterns. Finally, we perform an all-to-all communication in LoD to pass between two different strategies applied in the lower and the upper levels of the tree structure.

Translation: During the translations, radiation patterns of the basis clusters are converted into incoming fields for the testing clusters in $O(N \log N)$ time. In the lower levels of the tree structure, some of the translations can be performed in each processor without any communication. However, there are many basis and testing clusters that assigned to different processors so that their interactions require dense communications. We control these one-to-one communications by carefully examining the data traffic [13] and using various communication algorithms. For the upper levels of the tree structure, the translations can be performed without any communication due to the favorable properties of the strategy involving the distribution of the fields among the processors [12].

Disaggregation and Receiving: In the disaggregation process, the incoming fields at the centers of the testing clusters are calculated. At the end, the fields are disaggregated towards the testing functions in the lowest level, where we perform a numerical integration to finalize the matrix-vector multiplications. The disaggregation process is generally the inverse of the aggregation process and its parallelization is very similar to the parallelization of the aggregation.

Results

As an example to the solution of large problems, we present the results of a scattering problem involving a sphere of radius 96λ. The discretization of the problem leads to 33,791,232 unknowns using the RWG functions. The problem is formulated by the combined-field integral equation and solved on a cluster of quad-core Intel Xeon 5355 processors connected via an Infiniband network. The solution is parallelized into 16 processes running on 16 processors. We select the smallest cluster size as 0.19λ leading to a 9-level tree structure, where we choose the LoD as the sixth level from the bottom of the tree structure. The far-field interactions are computed with 2 digits of accuracy while the near-field interactions are evaluated with at most 1% error. Using the biconjugate-gradient-stabilized (BiCGStab) algorithm and a block-diagonal preconditioner, the number of iterations is only 21 to reduce the residual error below 10^{-3}. As detailed in Fig. 1, the processing time for the setup and solution parts are completed in 3.0 and 4.4 hours, respectively, while each matrix-vector product is performed in 370 seconds. During the solution, maximum 13.4 GB of memory is used per processor. To demonstrate the accuracy of the solution, Fig. 2 presents the bistatic radar cross section of the sphere from 160° to 180°, where 180° corresponds to the forward-scattering direction. We observe that the computational values are very close to the analytical curve obtained by a Mie-series solution. As a result, by employing parallel MLFMA, we are able to solve very large scattering problems with tens of millions of unknowns both accurately and efficiently. Additional examples and details of the implementation will be provided during the presentation.

References

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

[2] M. L. Hastriter and W. C. Chew, "Role of numerical noise in ultra large-scale computing," in *Proc. IEEE Antennas and Propagation Soc. Int. Symp.*, vol. 3, 2004, pp. 3373–3376.

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

[4] Ö. Ergül and L. Gürel, "Improving the accuracy of the magnetic field integral equation with the linear-linear basis functions," *Radio Sci.*, vol. 41, RS4004, doi:10.1029/2005RS003307, 2006.

[5] R. D. Graglia, "On the numerical integration of the linear shape functions times the 3-D Green's function or its gradient on a plane triangle," *IEEE Trans. Antennas Propagat.*, vol. 41, no. 10, pp. 1448–1455, Oct. 1993.

[6] P. Y.-Oijala and M. Taskinen, "Calculation of CFIE impedance matrix elements with RWG and $\hat{n} \times$RWG functions," *IEEE Trans. Antennas Propagat.*, vol. 51, no. 8, pp. 1837–1846, Aug. 2003.
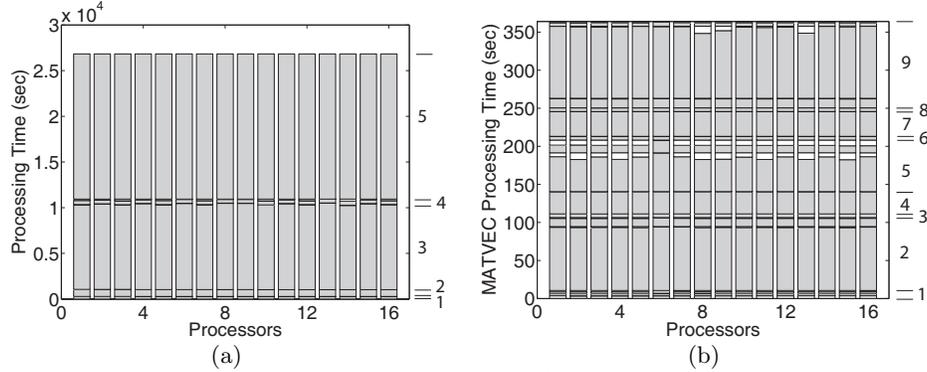
Figure 1. Time diagrams for the solution of a scattering problem involving a sphere of radius 96λ discretized with 33,791,232 unknowns. (a) Overall time includes the input and clustering part[1], calculation of translation matrices[2], calculation of near-field interactions[3], calculation of radiation and receiving patterns[4], and the iterative solution[5]. (b) Matrix-vector multiplications include the near-field interactions[1], aggregation in the lower levels[2], all-to-all communications in LoD[3,8], aggregation in the higher levels[4], translations without communications[5], translations with communications[6], disaggregation in the higher levels[7], and disaggregation in the lower levels followed by the receiving operation[9]. In the diagrams, white areas correspond to waits before the operations that require synchronization.
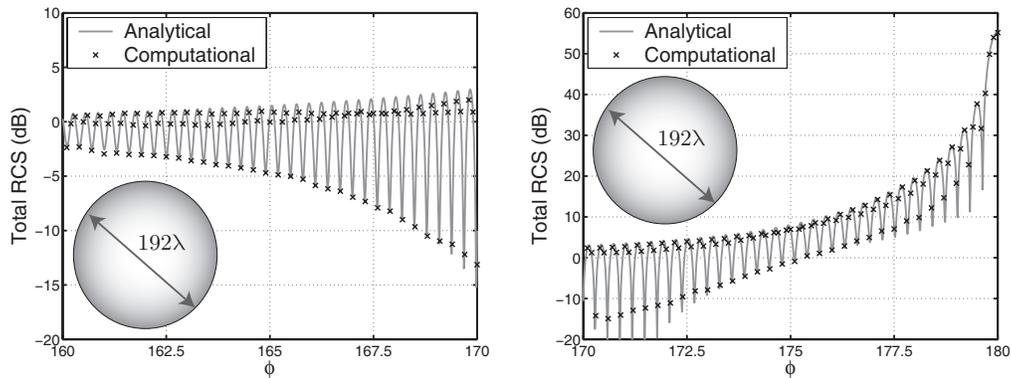

Figure 2. Bistatic radar cross section of a sphere of radius 96λ discretized with 33,791,232 unknowns from 160° to 180°, where 180° corresponds the forward-scattering direction.

[7] L. Gürel and Ö. Ergül, "Singularity of the magnetic-field integral equation and its extraction," *IEEE Antennas Wireless Propagat. Lett.*, vol. 4, pp. 229–232, 2005.

[8] L. Gürel and Ö. Ergül, "Investigations of load balancing, communications, and scalability in parallel MLFMA," *2006 IEEE AP-S International Symposium and USNC/URSI National Radio Science Meeting and AMEREM Meeting*, Albuquerque, New Mexico, USA, July 2006.

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

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

[11] 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.

[12] S. Velamparambil ve W. C. Chew, "Analysis and performance of a distributed memory multilevel fast multipole algorithm," *IEEE Trans. Antennas Propag.*, vol. 53, pp. 2719–2727, Aug. 2005.

[13] Ö. Ergül and L. Gürel, "Efficient parallelization of multilevel fast multipole algorithm," in *Proc. European Conference on Antennas and Propagation (EuCAP)*, 350094, 2006.