Multi-level Fast Multipole Algorithm for 3-D Homogeneous Dielectric Objects Using MPI-CUDA on GPU Cluster

Tuan Phan, Nghia Tran, and Ozlem Kilic

Department of Electrical Engineering and Computer Science
The Catholic University of America, Washington, DC, 20064, USA
30phan@cua.edu, 16tran@cua.edu, kilic@cua.edu

Abstract ─ The implementation of Multi-level Fast Multipole Algorithm (MLFMA) on a 13-node Graphical Processing Unit (GPU) cluster using Message Passing Interface (MPI) and CUDA programming is presented. The performance achievements are investigated in terms of accuracy, speed up, and scalability. The experimental results demonstrate that our MLFMA implementation on GPUs is much faster than (up to 66x) that of the CPU implementation without trading off the accuracy.

Index Terms ─ Graphics Processing Unit (GPU), Multilevel Fast Multipole Algorithm (MLFMA).

I. INTRODUCTION

In the last two decades, many authors have been investigating solving large scale electromagnetics problems using numerical techniques such as Method of Moments (MoM), Fast Multipole Method (FMM) and Multi-level Fast Multipole Algorithm (MLFMA). Modeling large-scale objects requires large memory resources and computational time. Among these methods, the MLFMA has the least computational complexity *O(NlogN)*, while MoM and FMM have the complexity of $O(N^3)$ and $O(N^{3/2})$, respectively.

MLFMA has successfully been implemented in parallel on CPU clusters to solve up to few hundreds millions of unknowns [1]. The CPU cluster-based parallel implementation has advantages of large memory resources, but their speed is relatively slow in comparison with GPU cluster-based implementations. In the past, our group has implemented a parallel version of MLFMA on GPUs clusters to solve for perfect electric conductor (PEC) objects [2]. This paper continues our efforts to investigate the implementation of MLFMA on GPU cluster platform for solving large scale dielectric objects. The platform we employ is a 13-node GPU cluster, which utilizes NVidia Tesla M2090 GPU. An MVAPICH2 implementation of Message Passing Interface (MPI) is used for parallel programming.

In this work, a workload partitioning technique, namely group-based distribution is investigated among the 13 computing nodes. This technique is applied for the tree structure in MLFMA as will be discussed in details in the implementation section. The rest of the paper is organized as follows. An overview of MLFMA for homogeneous dielectric objects is provided in Section 2. Section 3 presents the parallel implementation of MLFMA on GPU clusters. Simulation results are discussed in Section 4, followed by the conclusions in Section 5.

II. OVERVIEW OF THE MULTILEVEL FAST MULTIPOLE ALGORITHM ON DIELECTRIC OBJECTS

In this section, we provide a brief overview to help our discussion on the parallel implementation of dielectric MLFMA, which is presented in Section III. Numerical techniques such as MoM, FMM, and MLFMA are invented to solve for the linear equation system ZI = V, where I represents the unknown currents, V depends on the incident field, and Z is the impedance matrix. For an arbitrary structure meshed with M-edges the conventional MoM requires the computation of all direct interactions among the edges (MxM), while FMM accelerates the matrix-vector product by using an approximate multiple expansion of the fields to divide structure into near and far group interaction concept [3]. MLFMA is based on FMM, but it relies on forming hierarchical groupings to render reactions with far groups more efficiently. The main idea of the grouping concept of MLFMA is shown in Fig. 1, where M edges are categorized into an N-level tree structure. For the sake of simplicity and convenience, the oct-tree structure is used for grouping in MLFMA [4].

The near interactions among edges in spatially nearby groups are computed and stored using the conventional MoM [5], while the far interactions are calculated in a group-by-group manner consisting of three stages, namely, aggregation, translation, and disaggregation.

In our previous work on FMM for a homogeneous dielectric object (permittivity ε_2, permeability μ_2)

Submitted On: July 16, 2016
Accepted On: October 4, 2016

immersed in an infinite homogeneous medium (permittivity ε_1, permeability μ_1), the basic formulas are given as:

$$Z^1_{ij,JJ} = \frac{\omega\mu k}{16\pi^2}\int d^2\hat{\boldsymbol{k}}\, \boldsymbol{T}^E_{\boldsymbol{r}_{im}}(\hat{\boldsymbol{k}})T_L(k,\hat{\boldsymbol{k}},\boldsymbol{r}_{ii'})\bullet R^E_{m'i'}(\hat{\boldsymbol{k}}), \quad (1)$$

$$Z^1_{ij,MM} = \frac{\omega\varepsilon k\eta^2}{16\pi^2}\int d^2\hat{\boldsymbol{k}}\, \boldsymbol{T}^E_{\boldsymbol{r}_{im}}(\hat{\boldsymbol{k}})T_L(k,\hat{\boldsymbol{k}},\boldsymbol{r}_{ii'})\bullet R^E_{m'i'}(\hat{\boldsymbol{k}}), \quad (2)$$

$$Z^1_{ij,MJ} = \frac{k\eta}{16\pi^2}\int d^2\hat{\boldsymbol{k}}\, \boldsymbol{T}^{ED}_{\boldsymbol{r}_{im}}{}'(\hat{\boldsymbol{k}})T_L(k,\hat{\boldsymbol{k}},\boldsymbol{r}_{ii'})\bullet R^E_{m'i'}(\hat{\boldsymbol{k}}) = -Z^1_{ij,JM}, \quad (3)$$

where

$$T^E_{\boldsymbol{r}_{im}} = \int_S (\boldsymbol{I} - \hat{\boldsymbol{k}}\hat{\boldsymbol{k}})\bullet \boldsymbol{f}_m(\boldsymbol{r}_{im})e^{-j\boldsymbol{k}\bullet\boldsymbol{r}_{im}}dS, \quad (4)$$

$$T^{ED}_{\boldsymbol{r}_{im}} = \int_S \hat{\boldsymbol{k}}\times \boldsymbol{f}_m(\boldsymbol{r}_{im})e^{-j\boldsymbol{k}\bullet\boldsymbol{r}_{im}}dS, \quad (5)$$

$$R^E_{\boldsymbol{r}_{m'i'}} = \int_S \boldsymbol{f}_{m'}(\boldsymbol{r}_{m'i'})e^{-j\boldsymbol{k}\bullet\boldsymbol{r}_{m'i'}}dS, \quad (6)$$

$$T_L = \sum_{l=0}^{L}(-j)^l(2l+1)h_l^{(2)}(\boldsymbol{k}\bullet\boldsymbol{r}_{ii'})P_l(\hat{\boldsymbol{k}}\bullet\boldsymbol{r}_{ii'}), \quad (7)$$

In the above equations, L denotes for multipole expansion number, $h_l^{(2)}$ identifies the second kind of Hankel function, P_l stands for Legendre polynomial of degree l terms. By the changing the sub and superscripts "1" to "2" in Equations (1) to (7), we can complete the 2N linear equations. The same idea applies for MLFMA to solve for dielectric object [6].

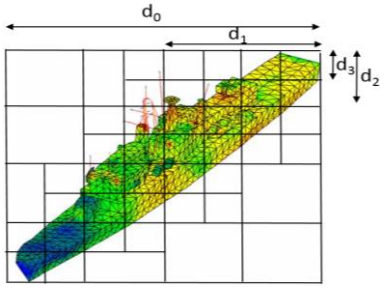


Fig. 1. MLFMA general grouping concepts.

III. GPU CLUSTER IMPLEMENTATION OF MLFMA

In this section, a detailed implementation of MLFMA is provided. The implementation is divided into three main blocks, which consist of pre-processing, processing and post-processing.

While the pre-processing and post processing processes utilize CPU, the processing are based on GPU cluster. The main purpose of the pre-processing step is to read the geometry mesh data, to set up the data structure, and to construct the oct-tree. Results from this process are transferred to the GPU memory, and the entire computation is performed on the GPU clusters. The user interested quantities such as scattered fields, radar cross section, are post-processing and handled on CPU. The processing step is the most time consuming in the algorithm. Hence, we focus our parallel programming of MLFMA on the most computationally intensive step, i.e., the processing. The details of this process is shown in Fig. 2.

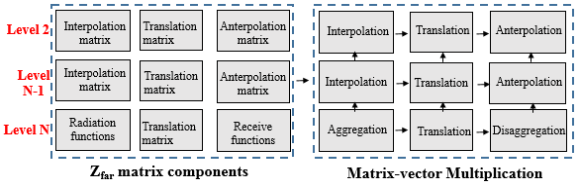


Fig. 2. A detail implementation of processing phase.

In the processing phase, the computational tasks are assigned to all computing nodes in a balanced manner such that each node holds the same amount of workload, and the inter-node communication is minimized. This is achieved by uniformly distributing the total number of groups of all levels except level 1 and 2, G, among the n computing nodes. We define this technique of data distribution among computing nodes as the group-based distribution. Two levels of parallelization are performed in this stage: among the n computing nodes using MPI library, and within the GPU per node using CUDA programming model. The CUDA thread-block model is utilized to calculate the assigned workload within a node. In this paper, only the far interactions is presented, and the near field and V vector calculation implementations can be found in [7].

The GPU cluster used for this work has 13 computing nodes. Each node has a dual 6-core 2.66 GHz Intel Xeon processor, 48 GB RAM along with one NVidia Tesla M2090 GPU running at 1.3 GHz supported with 6GB of GPU memory. The nodes are interconnected through the InfiniBand interconnection. The cluster populates CUDA v6.0 and MVAPICH2 v1.8.1 (an implementation of MPI).

A. Far interactions calculations

There are five main steps in this stage: radiation functions, receive function, interpolation, anterpolation and translation matrices. The group-based technique is performed to calculate the radiation functions, receive functions, and translation matrices.

(i) Radiation and Receive Function Calculations

The calculation of the radiation, T^E, and receive, R^E, functions for Z^{far} matrix are similar since R^E is the complex conjugate of T^E. Following the G group distribution as mentioned above, each computing node handles the calculation of K directions for G_{node} groups.

(ii) Translation Matrix Calculation

The workload for the T_L calculations is also distributed across the n nodes using the group-based

technique. In order to save memory, each CUDA block is assigned to compute one sparse row of the T_L matrix for a given direction.

(iii) Interpolation and Anterpolation Matrices

Due to the differences of sampling frequencies among the levels of the oct-tree structure, the interpolation and anterpolation are required for the aggregation and disaggregation stages. In this task, each node will handle the calculations of $K_{children/node}$ rows of the interpolation matrix $K_{children}$ * K_{parent}, where $K_{children}$ and K_{parent} are the number of directions of finer and coarser level, respectively. The blocks of a maximum of 1024 threads are utilized in the CUDA kernel once it is launched. The anterpolation is simply the transpose of the interpolation. Thus, their implementations are similar.

B. Matrix-vector multiplication

The matrix-vector multiplication (MVM) method is an important technique to accelerate the computational time, which can be found in detail in [8]. An iterative method; i.e., the biconjugate gradient stabilized method (BiCGSTAB), is used to solve for the linear system. The computation of $Z_{far}I$ is shown in Fig. 3, where the unknown current vector I is distributed among the 13 nodes using the group-based technique [9].

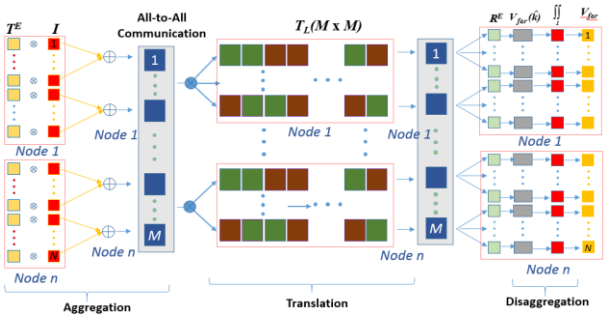


Fig. 3. The parallelization of matrix-vector multiplication for $Z_{far}I$.

First, in the aggregation stage, at level max, (N), each node computes the radiated fields for its assigned groups by multiplying the current I with the radiation functions, T^E, and accumulating within each group. Then, all-to-all communication is required to broadcast the data to all nodes. For the remaining levels (up to level 2), the radiated field is the result of multiplying interpolation matrices with radiated fields of its direct children groups.

In the translation stage, at each level (except levels 0 and l) the radiated fields for each group are calculated by multiplying the translation matrix with the radiated fields.

In the disaggregation stage, going down from level 2 to level N, the radiated fields at each group are added with the inherited fields from its parents using anterpolation. At the maximum level (N), the received fields are multiplied with their corresponding receive functions, and integrated over K directions. Then, the near components and far components of MVM are incorporated to complete the full matrix. In the end of this process, the results from all nodes are summed and updated.

IV. EXPERIMENTAL RESULTS

A. Accuracy

The accuracy of the method is verified by comparing the Radar Cross Sections (RCSs) of 9λ-diameter dielectric sphere with analytical technique, Mie scattering, and a 10λ-height by 4λ-radius dielectric cone with commercial simulation software, FEKO. In two cases, the results verify our method's accuracy, as observed in Fig. 4 and Fig. 5, respectively.

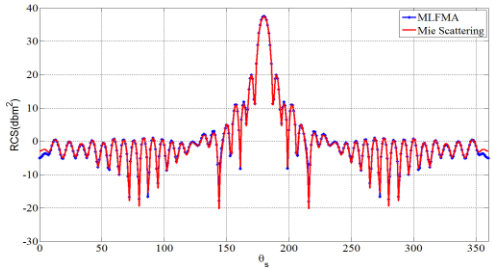


Fig. 4. RCS of a 9λ diameter dielectric sphere (ε = 4 – 0.1i) with 105,000 unknowns.

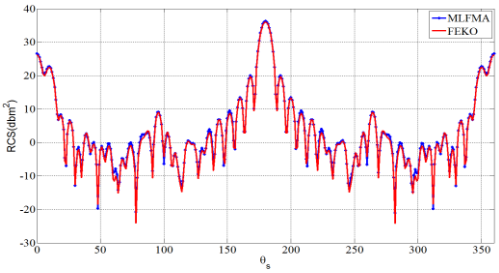


Fig. 5. RCS of 10λ height and 4λ radius dielectric cone (ε = 4 – 0.1i) with 109,000 unknowns.

B. Performance on GPU cluster

We conducted two experiments to investigate the speed-up, scalability using a fixed-workload model (Amdahl's Law) and maximum problem size. The speed-up is defined as the ratio of time required by multi-node GPU implementation with respect to the 8-node CPU implementation. The scalability is the normalized speed-up of multiple nodes in reference to the speed-up of 8 nodes. Finally, we fully utilized the memory available of 13 nodes to investigate the maximum number of unknowns we can handle.

In the first experiment, a 16.74 λ-diameter dielectric

sphere (320k unknowns) which requires the memory of at least 8 nodes is used. The results are evaluated in terms of speed-up and scalability. As shown in Fig. 6, the speed-up for process of matrix-vector products and matrix fill increases from 45.6 for 8 nodes to 66.4 for 13 nodes. The GPU execution time decreases as the number of nodes increases because of less workload per node.

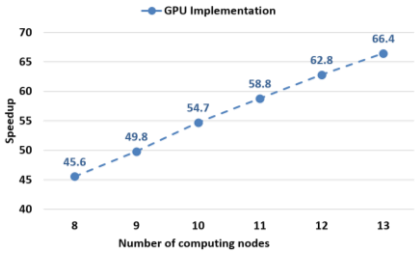


Fig. 6. Speedup analysis for the fixed-workload model (vs. 8 nodes CPU implementation, 100 iterations).

For the scalability, we keep the problem size constant and compare how the speed-up improves with increasing number of nodes, Fig. 7. It shows a good agreement between our implementation and the theoretical expectation.

In the second experiment, we try to solve for the largest problem size using the maximum memory available to us in each node. As the number of nodes increases, we increase the problem size to fully utilize the available memory. As shown in Fig. 8, we can process a maximum problem size of 439k unknowns with a speed-up of 46.

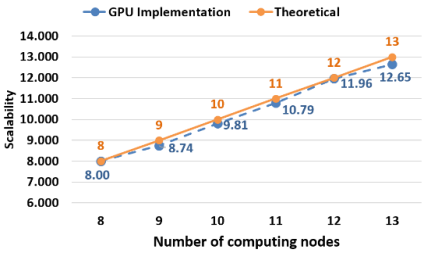
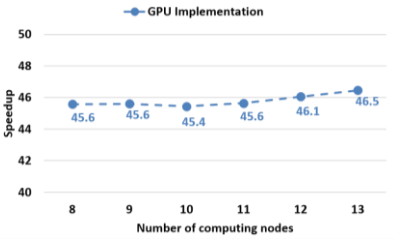


Fig. 7. Scalability analysis for the fixed-workload model.



Nodes	8	9	10	11	12	13
Size (Radius/Unknowns)	8.35λ/362K	8.5λ/375K	8.7λ/394K	8.9λ/413K	9.05λ/427K	9.15λ/439K

Fig. 8. Speed-up analysis for increasing number of nodes along with problem size increases.

V. CONCLUSION

In this paper,MLFMA for homogeneous dielectric objects has bee implemented using GPU clusters. Our 13-node GPU cluster is able to solve 426k unknowns utilizing the available on-board GPU memory. It demonstrates that the GPU implementation is much faster than CPU implementation while keeping a same degree of accuracy.

REFERENCES

[1] O. Ergul and L. Gurel, "Efficient parallelization of the multilevel fast multipole algorithm for the solution of large-scale scattering problems," *IEEE Trans. Antennas Propag.*, vol. 56, no. 8, pp. 2335-2345, August 2008.

[2] N. Tran and O. Kilic, "Parallel implementations of multilevel fast multipole algorithm on graphical processing unit cluster for large-scale electromagnetics objects," *ACES Express Journal*, vol. 1, no. 4, April 2016.

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

[4] H. Samet, *An Overview of Quadtrees, Octrees, and Related Hierarchical Data Structures*. NATO ASI Series, vol. F40, Springer-Verlag Berlin Heidelberg, 1988.

[5] 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.

[6] J.-Y. Li and L.-W. Li, "Characterizing scattering by 3-D arbitrarily shaped homogeneous dielectric objects using fast multipole method," *IEEE Antennas and Wireless Propagation Letters*, vol. 3, 2004.

[7] Q. M. Nguyen, V. Dang, O. Kilic, and E. El-Araby, "Parallelizing fast multipole method for large-scale electromagnetic problems using GPU clusters," *Antennas and Wireless Propagation Letters, IEEE*, vol. 12, pp. 868-871, 2013.

[8] V. Dang, Q. Nguyen, and O. Kilic, "Fast multipole method for large-scale electromagnetic scattering problems on GPU cluster and FPGA-accelerated platforms," *Applied Computational Electromagnetics Society Journal*, vol. 28, no. 12, 2013.

[9] V. Dang, Q. M. Nguyen, and O. Kilic, "GPU cluster implementation of FMM-FFT for large-scale electromagnetic problems," *IEEE Antennas and Wireless Propagation Letters*, vol. 13, 2014.