Parallel Out-of-Core MLFMA on Distributed-Memory Computer Architectures

Mert Hidayetoğlu and Levent Gürel

ABAKUS Computing Technologies, Cyberpark, Bilkent, Ankara, TR-06800, Turkey
lgurel@gmail.com

***Abstract*—We introduce a parallel implementation of the out-of-core method for the multilevel fast multipole algorithm. This implementation utilizes disk space for storing large data structures, and it provides solutions of large-scale problems involving more than one billion unknowns within 2 TB memory.**

I. Introduction

The multilevel fast multipole algorithm (MLFMA) solves a system of equations formulated with the method of moments [1], i.e., $\overline{\boldsymbol{Z}} \cdot \boldsymbol{x} = \boldsymbol{v}$, iteratively with $\mathcal{O}(N Log\, N)$ computational complexity [2], where N is the number of unknowns. MLFMA partitions a matrix-vector multiplication (MVM) as $\overline{\boldsymbol{Z}} \cdot \boldsymbol{x}_i = \overline{\boldsymbol{Z}}_{NF} \cdot \boldsymbol{x}_i + \overline{\boldsymbol{Z}}_{FF} \cdot \boldsymbol{x}_i$, where $\overline{\boldsymbol{Z}}_{NF}$ is the near-field and $\overline{\boldsymbol{Z}}_{FF}$ is the far-field interaction matrices, and $\boldsymbol{x}_i$ is the evolving guess for the unknown vector in the i^{th} iteration. $\overline{\boldsymbol{Z}}_{NF}$ has $\mathcal{O}(N)$ non-zero elements and is stored in the memory, whereas $\overline{\boldsymbol{Z}}_{FF}$ is calculated on-the-fly for each MVM in the iterative solution via aggregation, translation, and disaggregation steps in a multilevel tree structure. Even though $\overline{\boldsymbol{Z}}_{FF}$ is not stored in the memory at all, MLFMA stores the far-field patterns of the basis (and testing) functions, which requires $\mathcal{O}(N)$ memory, in order to use them in the iterative solution.

For solving large-scale problems, MLFMA is parallelized on distributed-memory computer architectures [3]. We use the hierarchical partitioning strategy, which achieves good load-balancing performance by partitioning both clusters and field samples (as opposed to partitioning only one of them) in all levels [4]. Despite the excellent performance of the hierarchical partitioning strategy, the memory requirements for storing the near-field interactions and the far-field patterns grow immensely when extremely large geometries are involved. Hence, the amount of total memory available in a computer constitutes a limitation for solving large problems. As a remedy, we propose the out-of-core method on MLFMA, which uses the large data from disk [5], and this paper discusses its parallel implementation on distributed-memory computer architectures.

MLFMA involves pre-processing, setup, and iterative-solution stages. Inputing and processing the geomety data and the tree structure are performed in the pre-processing stage. Near-field interactions, far-field patterns, and the translation operators among the branches of the tree structure are calculated and stored in the setup stage. Finally, the iterative solution is performed.

II. Parallel Out-of-Core Method

The out-of-core MLFMA stores the near-field interactions and the far-field patterns not in the core memory, but instead, in the disk. Specifically, parallel processes write the out-of-core data on disk(s) during the setup stage, then read them back as needed from the disk(s) during each MVM [5]. The I/O operations are performed on-the-fly through small buffers and the space required for storing the whole portion of the out-of-core data is never allocated in memory.

Hierarchical partitioning strategy distributes the basis functions among processes in a load-balanced manner. Hence, each process has its own portion of basis functions and their corresponding near-field interactions. In the parallel implementation of the out-of-core MLFMA, the processes use the out-of-core data simultaneously from disk and independent from each other without requiring any communications. When multiple-node computer architectures are involved, each process uses the disk space on the node it belongs to, as depicted in Fig. 1.

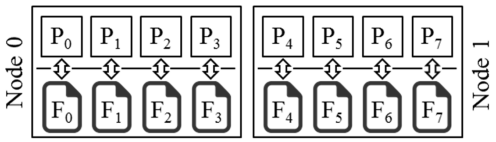

Fig. 1. As an example, the out-of-core MLFMA is depicted on a two-node computer cluster with 8 processes. In this case, each disk drive handles I/O jobs of four processes simultaneously. A process P_i names its own file F_i uniquely in order to access the file when needed. The arrows indicate I/O operations.

Once the processses store the out-of-core data on disks, they read the data multiple times during the iterative solution. In each MVM, the near-field interactions are read once for the near-field mutiplication, i.e., $\overline{\boldsymbol{Z}}_{NF} \cdot \boldsymbol{x}_i$, and the far-field patterns are read twice for the far-field multiplication, i.e., $\overline{\boldsymbol{Z}}_{FF} \cdot \boldsymbol{x}_i$, once for the aggregation and the second time for the dissaggregation steps.

III. Numerical Results

To demonstrate the performance of the out-of-core implementation, two scattering problems are solved, involving a conducting sphere and a conducting NASA Almond. The sphere has a diameter of 1000λ and the NASA Almond has a length of 2104λ, where λ is the wavelength of the illuminating plane wave in free space. The geometries are discretized with

978-1-4673-7197-1/15/$31.00 ©2015 IEEE

0.1λ mesh size, yielding approximately 1.1 billion unknowns for both problems. The combined-field integral equation with a combination factor of 0.5 is used to formulate the problems. The Bi-CGSTAB iterative solver is employed with a block-diagonal preconditioner [2] for the iterative solution to satisfy 1% residual error. 64 processes are employed on a 16-node cluster with a total memory of 2 TB. Each node is equipped with a single solid-state disk for storing the out-of-core data.

TABLE I
CPU TIMES AND DATA AMOUNTS FOR THE SOLUTIONS

	CPU Time (Hours)		Data Amount (TB)		
	Total	I/O	Total	Memory	Disk
NASA Almond	60.8	12.0	4.2	2.0	2.2
Sphere	71.3	10.7	4.0	1.9	2.1

Table I shows the CPU times along with the required memory and disk spaces for the solutions. The total data amount indicates the required memory space for solving the same problem in core memory, i.e., without using the out-of-core implementation. The out-of-core implementation decreases the required memory by 52% and 53%, but the I/O operations increase the overall CPU time by 12.0 and 10.7 hours, for the NASA Almond and sphere solutions, respectively. In Table I, the total solution times include the I/O times. With the same computer cluster, the largest sphere we can solve in core memory has a radius of 340λ and the problem involves approximately 540 million unknowns. That evidence shows that the out-of-core implementation is successful for decreasing the required memory space and providing solutions of extremely large problems.

Figure 2 shows the radar cross section (RCS) of the NASA Almond. The solution involves 1,126,503,936 unknowns. Involving two MVMs in each iteration, the iterative solution requires 39 iterations. The MVMs are performed through a 15-level tree structure, where the lowest three levels are "distributed" and the highest five levels are "shared" in the hierarchical partitioning scheme [4].

Figure 3 shows the RCS of the sphere. We observe that the computational values agree well with the analytical Mie-series solution. The computational RCS errors are 3.75%, 2.80%, and 0.94% in the 0°–30°, 0°–90°, and 0°–180° bistatic angle sectors, respectively. The error calculation method is

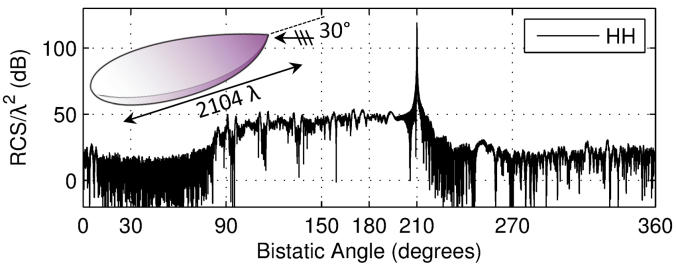

Fig. 2. Bistatic co-polar RCS of the NASA Almond. The geometry is illuminated at 30° from its sharp end on the azimuth plane with horizontal polarization. 30° and 210° are the backscattering and forward-scattering directions, respectively.

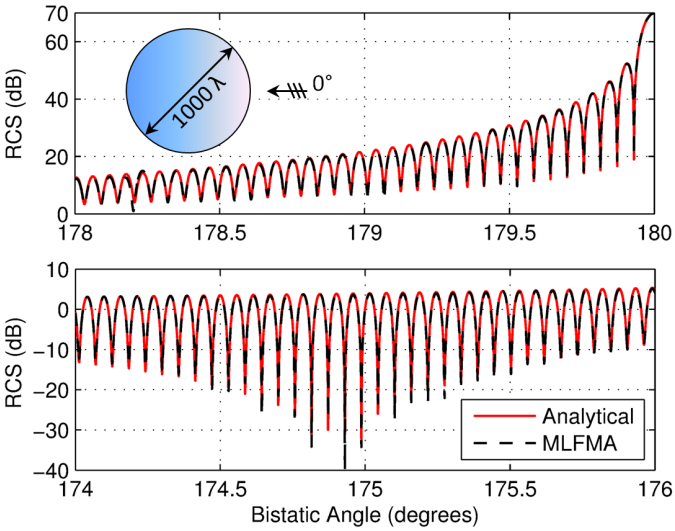

Fig. 3. RCS of the sphere. The MLFMA solution involves 1,109,280,768 unknowns. 180° and 0° are the forward-scattering and backscattering directions, respectively. The analytical solution is obtained with the Mie-series method.

provided in [6], where interested readers can compare their results with corresponding full-wave solutions of large-scale scattering problems including those presented in this paper.

IV. CONCLUSIONS

The out-of-core implementation of the parallel MLFMA is presented. Scattering results of a sphere and a NASA Almond with more than one billion unknowns are provided and the solution performance is demonstrated. Results show that the out-of-core method decreases the required memory for solutions and enables large-scale solutions in a limited memory, whereas it increases the required time by a reasonable amount due to the relatively slow disk operations.

ACKNOWLEDGMENT

This work was supported by Schlumberger-Doll Research (SDR). The authors would like to thank Intel Corporation and Jamie Wilcox for a generous allocation of parallel computer time.

REFERENCES

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

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

[3] 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. 2719–1727, Aug. 2005.

[4] Ö. Ergül and L. Gürel, "A hierarchical partitioning strategy for an efficient parallelization of the multilevel fast multipole algorithm," *IEEE Trans. Antennas Propag.*, vol. 57, no. 6, pp. 1740–1750, June 2009.

[5] M. Hidayetoğlu and L. Gürel "MLFMA memory reduction techniques for solving large-scale problems," *2014 IEEE Int. Symp. Antennas Propagation USNC-URSI Nat. Radio Science Meeting*, Memhpis, TN, USA, July 2014.

[6] ABAKUS Benchmarking Tool, Apr. 14, 2015. [Online]. Available: http://abakus.computing.technology/benchmark.