Analysis of a scalable, parallel, 2D MLFMA solver.

Jan Fostier[1] and Daniël De Zutter[1]

[1]Department of Information Technology (INTEC), Ghent University,
Sint-Pietersnieuwstraat 41, B-9000 Ghent, Belgium.
jan.fostier@intec.ugent.be, daniel.dezutter@intec.ugent.be

Abstract: Recent advances in the parallelization of the Multilevel Fast Multipole Algorithm (MLFMA) have resulted into fully scalable parallelization schemes. By means of an asynchronous implementation of the hierarchical partitioning scheme, we demonstrate this ability by employing up to 512 processor cores. Furthermore, we demonstrate the capabilities of the solver through the simulation of a very large 2D canonical example with a diameter of three million wavelengths.

Keywords: Multilevel Fast Multipole Algorithm, scalable parallelization, bistatic RCS.

1. Introduction

The Multilevel Fast Multipole Algorithm (MLFMA) is arguably the most successful algorithm to accelerate the matrix-vector multiplication that arises during the iterative Method of Moments (MoM) solution of boundary integral equations. Indeed, the algorithm lowers the complexity from $O(N^2)$ to only $O(N \log N)$, where N denotes the number of unknowns. This close to linear increase of computational resources with respect to the number of unknowns makes the algorithm a good candidate for parallelization. In the past decade, a lot of research has been invested into this topic [1, 2, 3, 4, 5, 6, 7, 8, 9]. The key focus in these efforts has been the reduction of the amount of communication between the computational elements on the one hand and methods to ensure a good load balancing on the other hand.

The initial efforts to parallelize the MLFMA were based on techniques that existed for the solution of Laplace equation problems with the Fast Multipole Method (FMM). They relied on the distribution of the boxes in the tree data structure of the FMM algorithm. Because the amount of data stored inside a box is constant for Laplace problems, this method proved fruitful. For Helmholtz kind of problems, the amount of data inside a box increases significantly when traveling upwards in the tree, and the simple distribution of boxes led to a non-scalable algorithm and poor parallel efficiencies.

The first attempts to resolve the bottleneck at the top levels relied on a hybrid distribution of the workload [1, 2, 3]. For the lower levels, where the content of the boxes is limited, distribution of boxes (called 'spatial partitioning') is used. For the higher levels, the boxes are shared between the nodes, and the content of the boxes (i.e. radiation pattern sampling points or so-called 'k-space partitioning') is distributed. This method could significantly improve parallel efficiencies [1, 3], but could not resolve the scalability breakdown.

Recently, the hierarchical partitioning algorithm was introduced [4]. It provides for a gradual transition between spatial and full k-space partitioning. The algorithm distributes the data at every level of the MLFMA tree evenly among all participating nodes, and hence provides for a good load balancing and again delivers improved parallel efficiencies [3, 9]. Furthermore, under certain conditions, the algorithm can provide for a scalable MLFMA [7]. This means that larger problems can be handled on proportionally larger parallel machines, without loss of efficiency ($P = O(N)$, with P the number of processors).

In this contribution, we review a two-dimensional implementation of the MLFMA that has been parallelized using the hierarchical approach. Using a large cluster consisting of 512 computational cores, the

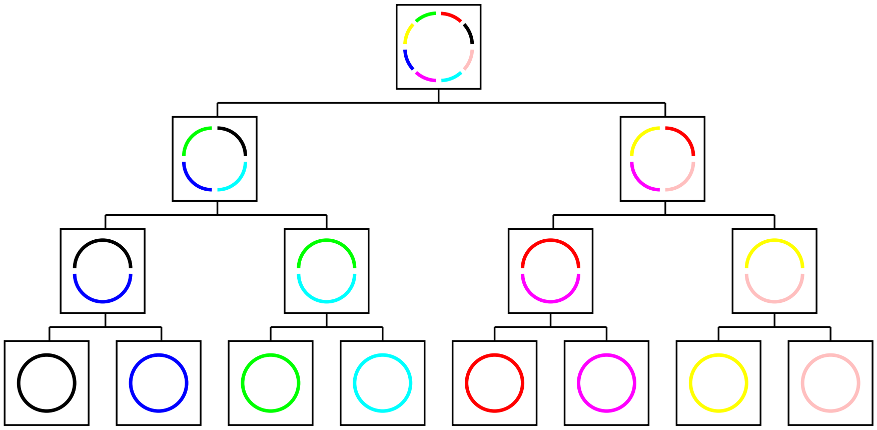

Figure 1: Schematic representation of the hierarchical *k*-space partitioning: the squares represent a set of boxes in the MLFMA tree; the circle segments represent a portion of the radiation pattern samples that are stored in the box; the different colors indicate the processor in which the samples are stored.

scattering at a perfectly electrically conducting (PEC) cylinder with a diameter of three million wavelengths is simulated and compared to the analytical solution. The problem is discretized in 94 247 780 unknowns. In terms of wavelengths, this problem is the largest scattering problem that has been solved up to date.

2. 2D Parallel MLFMA: hierarchical partitioning

In this Section, we briefly review the parallelization of the MLFMA tree and the hierarchical *k*-space partitioning. For an introduction to the MoM that is used, we refer to e.g. [6], for an introduction to the (sequential) MLFMA, we refer to [10].

First, each node constructs the global MLFMA tree for the geometry under consideration and the boxes at each level are ordered according to a Hilbert space filling curve. This is a sequential step. It is a known fact that each level contains $O(N)$ sampling points [10]. At the lowest levels, there are $O(N)$ boxes containing a constant number (or $O(1)$) sampling points, at the highest levels, there are $O(1)$ boxes containing $O(N)$ sampling points. Subsequently, $O(N/P)$ sampling points (with $P = O(N)$) are allocated to each node at each level in the following manner (see also Fig. 1). At the lowest level, each node is allocated $O(N/P) = O(1)$ boxes with their $O(1)$ sampling points as a whole (spatial partitioning). At the next level, the boxes are shared between two nodes. However, each node now holds only half of the radiation pattern samples. At the next level, the boxes are shared by four nodes, each containing only one fourth of the sampling points. This process continues until at the top level, the box is shared by all nodes, each node containing $1/P^{th}$ of the radiation pattern. The transition from spatial to *k*-space partitioning requires $\log_2(P)$ levels. For more information regarding this repartitioning process, we refer to [4, 5].

It is clear that every node contains $O(1)$ boxes and $O(1)$ samples at each level. Because the calculation time is proportional to the number of sampling points, it is also $O(1)$ per node and per level. It can be shown that the amount of communication per node and per level is also $O(1)$ [7], which means that this approach indeed allows for a number of processes $P = O(N)$. The hierarchical partitioning technique not only strongly reduces the amount of communication [5], it also reduces the number of communication events. Indeed, a certain node only has to communicate to $O(\log N)$ other nodes (i.e. $O(1)$ nodes per level).

The use of the hierarchical partitioning technique requires a local interpolator. In this case, a Dirichlet kernel with a Gaussian taper is used [11] although an interpolator based on periodic approximate prolate spheroidal (APS) functions [12] is slightly more performant. In a practical situation, the number of processes

Table 1: Simulation parameters for the canonical example.

# unknowns	94 247 780
# processor cores	512
smallest box size	0.25 λ
# MLFMA levels	25
precision FMM interactions	10^{-5}
# iterations	1 761
preconditioner	16 $\lambda\times$ 16 λ
setup time	28 min.
solution time	6h 40min.
time for matrix-vector product	6.49s
# RCS output points	74 547 200

P is much smaller than 2^L, with L the number of levels in the tree. This means that typically spatial partitioning is used for the first 6 or 7 levels, after which the hierarchical partitioning scheme is deployed. For these lowest levels, a global interpolator based on Fast Fourier Transforms (using the FFTW [13] package) is used because it is more accurate and faster. For the remaining top levels, if any, full *k*-space partitioning is used.

3. Example

We consider the Transverse Magnetic (TM) plane-wave scattering at a 2D PEC cylinder with a diameter of $3\,000\,000\lambda$. The problem was solved on a cluster consisting of 64 machines, each containing 2 quad-core Intel Xeon L5420 processors and 16 Gigabyte RAM (512 cores and 1 TByte RAM in total). The machines are connected through a fast 20 Gigabit/s Infiniband network and the proprietary 'Intel MPI' implementation of the Message Passing Interface (MPI) was used as a communication library. Double precision arithmetics were used for all calculations.

Using a $\lambda/10$ segment size, the problem is discretized in 94 247 780 unknowns. Using the TFQMR iterative method, the problem was solved in 1 761 iterations to a relative residual error of 10^{-3}, using the 512 cores and 1 TByte of RAM memory in total. Other simulation parameters of interest are listed in Table 1.

The bistatic radar cross-section (RCS) σ_c was numerically calculated for N equidistant angles θ_i and compared to the analytical solution σ_a. The evaluation of the analytical solution can be accelerated using the fast cosine transform, allowing for an evaluation in only a few minutes. We determine the root mean square (RMS) error as follows

$$\text{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} |\sigma_a(\theta_i) - \sigma_c(\theta_i)|^2} \tag{1}$$

For the full [0°. . . 360°] range, the RMS error is only 0.165 dB, indeed yielding very accurate results. Fig. 2 both shows the full bistatic RCS and a detail around 0°.

Fig 3 shows the total amount of communication between the 512 different computation cores. One can notice that the hierarchical approach indeed leads to very sparse communication patterns, i.e. a certain node only has to communicate to a limited number of other nodes. The total amount of data communication by e.g. the first node is 104.5 MByte (outgoing) and 102.5 MByte (incoming). These data are transmitted or received by only 22 other nodes. The largest amounts of communications occur during the repartition events. The number of such events is equal to $2\log_2 P$ (both upward and downward pass), assuming that a

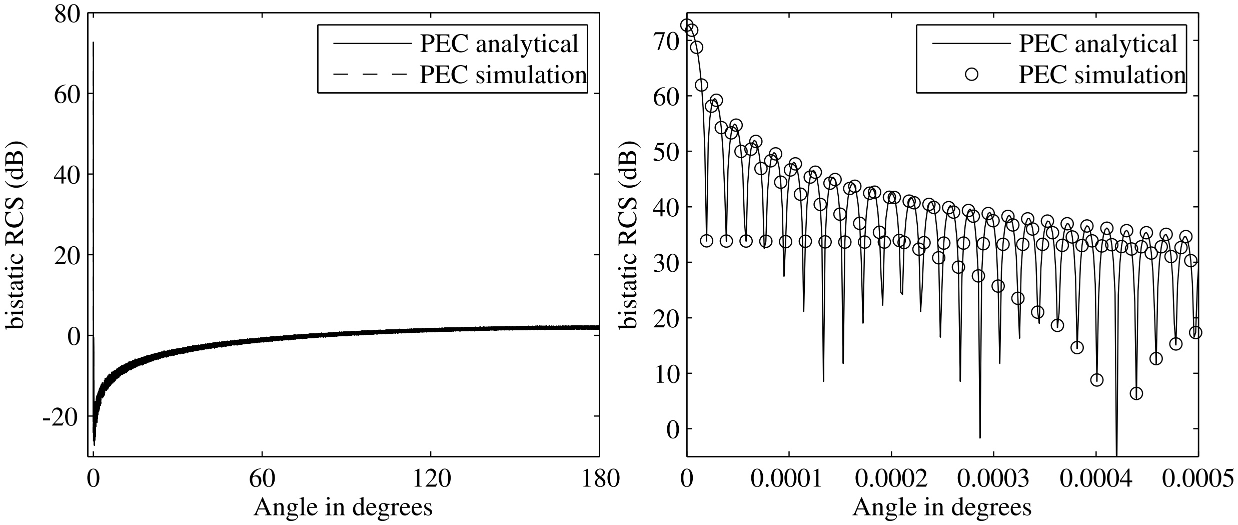

Figure 2: Comparison between the analytical and simulated bistatic RCS for the scattering at a PEC cylinder with a diameter of $3\,000\,000\lambda$. The full RCS $\theta \in [0° \ldots 180°]$ (left) and detail for θ around $0° \in [0° \ldots 0.0005°]$ is given.

complete transition between spacial and *k*-space partitioning is made. In the case of this example, there are 2×9 repartitioning events. The other communication events include the communication of *k*-space samples at the edges of the partitions for local interpolation, the communication of sampling points to complete translations and the communication of small portions of the unknown vector in order to complete the near interactions.

Conclusion

We have reviewed and analyzed a scalable parallel implementation of a 2D Method of Moments solver that is accelerated through the Multilevel Fast Multipole Algorithm. An example containing over 94 million of unknowns was solved on a large cluster using 512 computational cores, demonstrating the ability to solve extremely large scattering problems. Because of the scalability of the parallel algorithm, it is to be expected that the size of the problems can be increased even further, provided that a sufficiently computational cluster is provided.

Acknowledgment

The computational resources and services used in this work were provided by Ghent University. The work of J. Fostier was supported by a grant of the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT-Vlaanderen).

References

[1] 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.

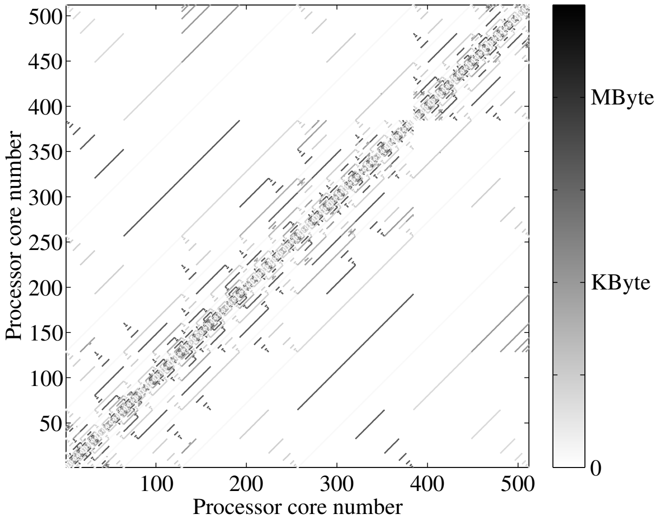

Figure 3: Communication map between the different processor cores. Each dot in the graph represents the total amount of communication (incoming + outgoing) between two cores for one matrix-vector multiplication.

[2] X.-M. Pan and X.-Q. Sheng, "A sophisticated parallel MLFMA for scattering by extremely large targets," *IEEE Antennas Propag. Mag.*, vol. 50, no. 3, pp. 129–138, Jun. 2008.

[3] Ö. Ergül and L. Gürel, "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. 2235–2345, Aug. 2008.

[4] ——, "Hierarchical parallelisation strategy for multilevel fast multipole algorithm in computational electromagnetics," *Electronics Letters*, vol. 44, no. 1, pp. 3–5, Jan. 2008.

[5] ——, "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, Jun. 2009.

[6] J. Fostier and F. Olyslager, "An asynchronous parallel MLFMA for scattering at multiple dielectric objects," *IEEE Trans. Antennas Propag.*, vol. 56, no. 8, pp. 2346–2355, Aug. 2008.

[7] ——, "Provably scalable parallel multilevel fast multipole algorithm," *IET Electronics Letters*, vol. 44, no. 19, pp. 1111–1113, Sep. 2008.

[8] ——, "Full-wave electromagnetic scattering at extremely large 2D objects," *IET Electronics Letters*, vol. 45, no. 5, pp. 245–246, Feb. 2009.

[9] ——, "An open source implementation for 2D full-wave scattering at million wavelength size objects," *Submitted to Antennas and Propagation Magazine*, 2009.

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

[11] S. Ohnuki and W. C. Chew, "Numerical analysis of local interpolation error for 2D-MLFMA," *Microw. and Opt. Techn. Lett.*, vol. 36, no. 1, pp. 8–12, Jan. 2003.

[12] M. Lu and E. Michielssen, “A local filtering scheme for FMM/PWTD algorithms,” in *Proc. IEEE Int. Symp. Antennas Propag.*, vol. 4, Jun. 2004, pp. 4523–4526.

[13] M. Frigo and S. G. Johnson, “The design and implementation of fftw3,” *Proceedings of the IEEE*, vol. 93, no. 2, pp. 216–231, Feb. 2005, www.fftw.org.