Weak Scalability Analysis of the Distributed-Memory Parallel MLFMA

Bart Michiels, Jan Fostier, *Member, IEEE,* Ignace Bogaert, *Member, IEEE,* and Daniël De Zutter, *Fellow, IEEE*

***Abstract*—Distributed-memory parallelization of the Multilevel Fast Multipole Algorithm (MLFMA) relies on the partitioning of the internal data structures of the MLFMA among the local memories of networked machines. For three existing data partitioning schemes (spatial, hybrid and hierarchical partitioning), the weak scalability, i.e. the asymptotic behavior for proportionally increasing problem size and number of parallel processes, is analyzed. It is demonstrated that none of these schemes are weakly scalable. A non-trivial change to the hierarchical scheme is proposed, yielding a parallel MLFMA that does exhibit weak scalability. It is shown that, even for modest problem sizes and a modest number of parallel processes, the memory requirements of the proposed scheme are already significantly lower, compared to existing schemes. Additionally, the proposed scheme is used to perform full-wave simulations of a canonical example, where the number of unknowns and CPU-cores are proportionally increased up to more than 200 millions of unknowns and 1024 CPU-cores. The time per matrix-vector multiplication for an increasing number of unknowns and CPU-cores corresponds very well to the theoretical time complexity.**

***Index Terms*—MLFMA, parallelization, weak scalability**

I. Introduction

ARGUABLY, the use of boundary integral equations is one of the most powerful and popular methods to solve large electromagnetic scattering problems in piecewise homogeneous media. A Method of Moments (MoM) discretization gives rise to a dense system of N linear equations and N unknowns which can be solved iteratively. The Multilevel Fast Multipole Algorithm (MLFMA) reduces the computational complexity of the matrix-vector multiplication in this iterative scheme from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$ [1], allowing simulations with a large number of unknowns. To tackle problems that exhibit memory requirements beyond what can be provided by a typical workstation, the development of an efficient distributed-memory parallel MLFMA is warranted. The data structures associated with the MLFMA are then distributed over the local memories of several nodes in a computational cluster. Each node performs only a fraction of the total computations and relies on network communication to access data stored in the memory of another machine. Besides the ability to handle larger problems, parallel algorithms usually exhibit an important reduction in runtime.

In the past years, several distributed-memory parallel MLFMA implementations have been proposed in literature, aimed at high-frequency (i.e. geometry size $\gg \lambda$) three-dimensional scattering problems. They can be categorized according to how the data structures of the MLFMA are partitioned over the different processes, namely *spatial* [2], [3], [4], [5], *hybrid* [6], [7], [8], [9], [10] and *hierarchical* [11], [12], [13] partitioning.

Two scalability measures are important in the assessment of a particular parallel algorithm. In a *strong scaling* analysis, the speedup as a function of the number of parallel processes is observed for a *fixed* problem size. In the ideal case, this speedup S is equal to the number of processes P and the parallel efficiency (i.e. the ratio of S to P) is 100%. However, because of e.g. communication overhead and load imbalance, such speedups are rarely observed in reality. In the asymptotic case of a very large number of processes, the speedup is always bounded (cfr. Amdahl's law) and the efficiency tends to zero. The maximum speedup that can be attained depends on the problem size, implementation quality, speed of CPUs and interconnection network, the ability to overlap communications and computations, load balancing, etc.

Alternatively, in a *weak scaling* analysis, the ability to handle *larger* problems using a proportionally higher number of parallel processes is investigated. In other words, the problem size *per process* is fixed. Suppose a problem of size N can be handled using P processes with a certain parallel efficiency. An algorithm is then said to be weakly scalable if a problem twice the original size can be handled on twice the number of processes, with the same efficiency. Clearly, weak scalability is a very beneficial property. As opposed to strong scalability, weak scalability is an intrinsic property of a parallel algorithm, i.e. it is not related to the implementation quality or the parallel architecture used.

Most authors only investigate the strong scaling behavior of their algorithms. The term *scalable* then denotes that, for a specific problem size, high parallel efficiencies can be obtained using a certain number of processes. However, a strong scaling analysis does not reveal whether or not these efficiencies can be attained for larger problems to be solved on a (future) larger cluster. In this work, we investigate the weak scaling behavior of the spatial, hybrid and hierarchical partitioning scheme, by assessing the asymptotic behavior of these algorithms for large N and P. It turns out that these schemes are not weakly scalable, i.e. the parallel efficiency will tend to zero for sufficiently large N and P. We propose a change to the hierarchical scheme, yielding a parallel MLFMA that does exhibit weak scalability. Numerical experiments with actual implementations of each of the four schemes confirm our theoretical findings.

We motivate our work as follows. First, since the introduction of the multi-core CPU in 2003, progress in computational

The authors are with the Department of Information Technology (INTEC), Ghent University, Sint-Pietersnieuwstraat 41, B-9000 Ghent, Belgium (e-mail: bart.michiels@intec.ugent.be).

Manuscript received December 12, 2012; revised April 3, 2013.

power of CPUs is mainly achieved by incorporating more and more CPU-cores. Second, more powerful clusters are built by assembling an increasing number of networked machines, each machine typically containing a number of multi-core CPUs. Clusters containing several thousands of CPU-cores are nowadays widespread. However, the speed of a *single* core and the available memory *per core* has progressed at a much slower pace. This trend is likely to continue in the future. In order to take advantage of current and future infrastructures, an efficient parallel algorithm is required that exhibits weak scalability.

This paper is organized as follows: in Section II, the weak scaling bottlenecks are identified for three existing data partitioning schemes using an asymptotic analysis. A fourth scheme is proposed that exhibits weak scalability. Next, in Section III, implementations of each of the four schemes are numerically compared. In Section IV, we apply our weakly scalable parallel solver to simulate a large problem with more than 200 millions of unknowns. Finally, our conclusions are presented in Section V. Parts of the ideas in this work have been presented in [17] and [18]. Here, a much more comprehensive and detailed analysis is put forward together with an actual implementation of our proposed algorithm.

II. Weak scaling analysis: theory

A. General considerations

We consider a high-frequency (i.e. geometry size $\gg \lambda$), three-dimensional scattering problem that is formulated using boundary integral equations. The mesh size is inversely proportional to the frequency, e.g. $\lambda/10$. In the MLFMA, the N unknowns are recursively subdivided in a tree-like structure of boxes with $\mathcal{O}(\log N)$ levels. At the lowest level, there are $\mathcal{O}(N)$ boxes, each holding a radiation pattern consisting of a constant number (i.e. independent of N, or $\mathcal{O}(1)$) of sampling points. When going up one level in the MLFMA tree, the number of boxes decreases roughly by a factor of four, whereas the size of the radiation patterns increases roughly by the same factor (see Table I). Hence, the top levels contain $\mathcal{O}(1)$ boxes, each holding a radiation pattern of size $\mathcal{O}(N)$. Each level contains $\mathcal{O}(N)$ sampling points in total. Because only a constant amount of work is required per sampling point, the amount of calculations to perform on each level is also $\mathcal{O}(N)$. Taking the $\mathcal{O}(\log N)$ levels into account yields a total complexity of $\mathcal{O}(N \log N)$ for the sequential MLFMA.

To assess the weak scalability, the asymptotic behavior for a proportionally increasing problem size N and number of processes P will be investigated, i.e. $P = \mathcal{O}(N)$. Note that this does not impose a strict linear dependency of P on N, but rather an asymptotic upper bound of how fast the number of processes *can* grow as a function of N. Following the assumption that the $P = \mathcal{O}(N)$ processes operate *concurrently*, the complexity *per process* should not exceed $\mathcal{O}(\log N)$. Because of inherent data dependencies between the radiation patterns on different levels, concurrency can only be achieved by distributing the $\mathcal{O}(N)$ work at each level among all processes. In other words, the computational complexity per process and per level should be $\mathcal{O}(1)$.

TABLE I
Number of radiation pattern sampling points Q_l and number of boxes B_l as a function of the MLFMA-level l for problem S_6 as defined in Section III.

level l	box size	Q_l	$\frac{Q_l}{Q_{l-1}}$	B_l	$\frac{B_{l-1}}{B_l}$
0	0.5λ	1 200	n/a	4 024 568	n/a
1	1λ	2 380	1.98	1 003 688	4.01
2	2λ	3 280	1.38	249 698	4.02
3	4λ	8 844	2.70	62 426	4.00
4	8λ	27 144	3.07	15 608	4.00
5	16λ	88 620	3.26	3 752	4.16
6	32λ	309 684	3.49	866	4.33
7	64λ	1 135 524	3.67	218	3.97
8	128λ	4 295 380	3.78	56	3.89
9	256λ	16 571 524	3.86	8	7
10	512λ	64 740 820	3.91	1	8

The time to send a message of size n between two processes is modeled as $\alpha+\beta n$, where α denotes the *latency* (i.e. the time to send an empty message) and $\frac{1}{\beta}$ the bandwidth. Therefore, weak scalability implies that also the communication volume per process and per level should be bounded by $\mathcal{O}(1)$. Note that a non-blocking communication model is assumed where two processes can communicate at full speed, regardless of any ongoing communication between other processes.

In the following sections, we investigate the computational, memory and communication complexity of three data partitioning strategies (spatial, hybrid and hierarchical partitioning) and show that they exceed $\mathcal{O}(1)$ per level and per process. Next, an augmented hierarchical partitioning scheme is proposed that is weakly scalable. In what follows, the term *scalable* always refers to weak scalability.

B. Spatial partitioning

The earliest attempts at parallelizing the MLFMA were based on the distribution of boxes (spatial partitioning (SP), sometimes referred to as *simple* partitioning) [2], [3], [4], [5]. Only at a constant number of lowest levels, the $\mathcal{O}(N)$ boxes can be evenly divided among $P = \mathcal{O}(N)$ processes, yielding a complexity of $\mathcal{O}(1)$ per process. On all other levels, the number of boxes grows slower than linear as a function of N. For increasing N, the number of processes P will eventually become larger than the number of boxes, which means that certain processes will not be attributed a box, rendering them idle and yielding an unfavorable load balancing.

From a different perspective, consider the complexity of a process that is attributed a top-level box. Because such a box contains $\mathcal{O}(N)$ sampling points, the computational complexity for that process is also $\mathcal{O}(N)$. Also, if such radiation patterns need to be communicated to another process (e.g. during the translation phase), the communication complexity is $\mathcal{O}(N)$. Clearly, spatial partitioning is not scalable.

C. Hybrid partitioning

Velamparambil *et al.* [6], [7] recognized this bottleneck and proposed the hybrid partitioning (HyP) scheme to alleviate the poor load balancing at the top levels. For the lower half of the tree, spatial partitioning is used as described above. For the upper half of the tree, the k-space partitioning (KP)

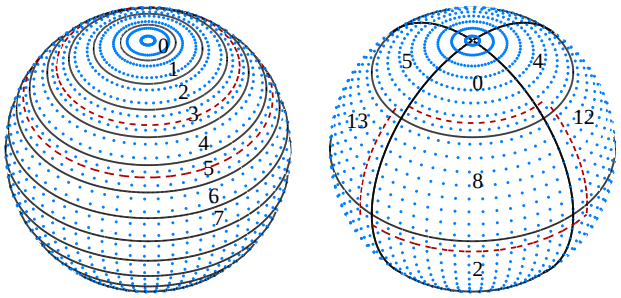


Fig. 1. Stripwise (left) vs. blockwise (right) partitioning of radiation pattern samples (blue dots). The solid lines mark the different partition boundaries while the numbers denote the process to which the partition is attributed. The (red) dashed line encompasses all sampling points required for a local interpolation of that partition.

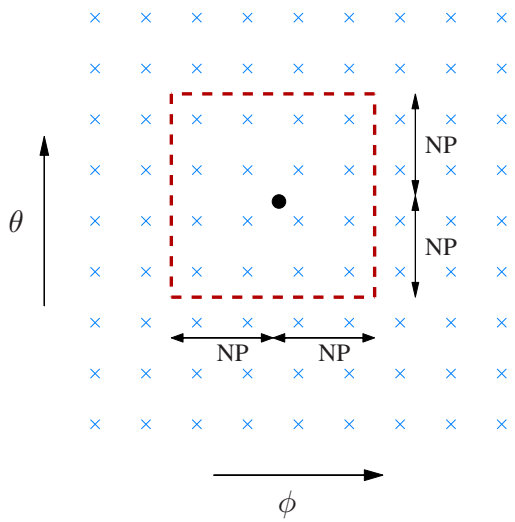
scheme was proposed. Instead of distributing the boxes among all processes, the sampling points *within* a box are distributed among all processes. Because the top-level radiation patterns contain $\mathcal{O}(N)$ sampling points, k-space partitioning attributes $\mathcal{O}(1)$ sampling points to each process for these levels. The hybrid scheme requires the transition from spatial to k-space partitioning at some level. The optimal level depends on the specific number of boxes and sampling points. From a complexity analysis point of view, the middle level is appropriate. At this transition level, i.e. the lowest level with k-space partitioning, there are $\mathcal{O}(\sqrt{N})$ boxes each containing $\mathcal{O}(\sqrt{N})$ sampling points.

Even in this improved scheme, bottlenecks continue to exist, as also pointed out in [6], [7]. The highest level that is partitioned using SP contains only $\mathcal{O}(\sqrt{N})$ boxes. For increasing N and P, the number of processes will again become larger than the number of available boxes. Processes that are attributed a box have a computational complexity of $\mathcal{O}(\sqrt{N})$. Similarly, at the lowest level that is partitioned using KP, the boxes contain only $\mathcal{O}(\sqrt{N})$ sampling points which can not be evenly partitioned among $\mathcal{O}(N)$ processes. Even though the HyP scheme reduces the worst-case complexity per process and per level from $\mathcal{O}(N)$ to $\mathcal{O}(\sqrt{N})$ compared to SP, the HyP scheme is also not scalable. However, this bottleneck in HyP will only become apparent for a higher number of processes than is the case for SP.

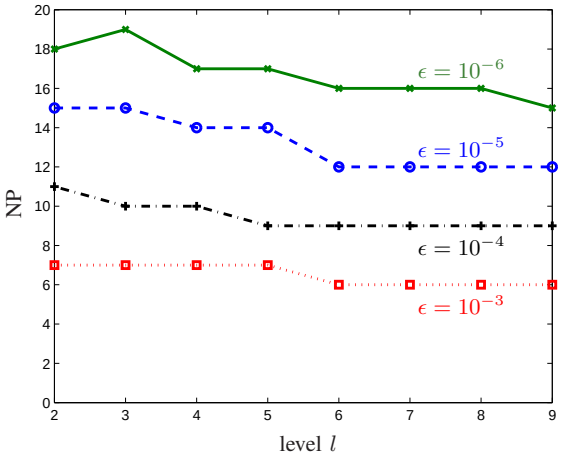
D. Hierarchical partitioning

Hierarchical partitioning (HiP), introduced in [11], [12], [13], uses a gradual transition between spatial and k-space partitioning. At the lowest level(s), the boxes are distributed using SP. At the next level, each box is shared among four processes, however, each process now only holds a quarter of the sampling points. At every next level, the radiation patterns are further repartitioned into an increasing number of $4, 16, 64, \ldots, P$ partitions, until eventually, full k-space partitioning is obtained at the top levels. Note that we assume for simplicity that P is a power of four.

Hierarchical partitioning *can* result in a scalable parallelization. For the two-dimensional MLFMA, this has been shown in [14], [15], [16]. In three dimensions however, special care



(a) Local interpolation of a source radiation pattern (the input for the interpolation, denoted by the blue crosses) in a point of a destination radiation pattern (one output point of the interpolation, denoted by the black dot). The (red) dashed line encompasses all sampling points required to calculate the value in the black dot. Two source points (NP) on each side in the θ and ϕ-direction are needed.



(b) Number of neighboring source points (NP), on each side in the θ and ϕ direction, required for local interpolation as a function of the MLFMA level l for different target precisions ϵ for problem S_6. For level 0 and 1 an FFT-interpolator is used, as their target level still uses spatial partitioning (SP).

Fig. 2. Local interpolation of the radiation patterns.

needs to be taken of *how* the radiation pattern sampling points are distributed among the processes. We consider two scenarios, denoted the *stripwise* and *blockwise* (see Fig. 1) approach. At first glance, this choice may seem to be an implementation detail, however, it follows from the complexity analysis that the former does not lead to a scalable algorithm whereas the latter does.

1) Stripwise scheme: In [11], [12], [13] the radiation patterns are partitioned *stripwise* (S-HiP): the values of the θ-range (elevation) are distributed among the different processes, irrespective of the ϕ-values (azimuth), as shown in Fig. 1

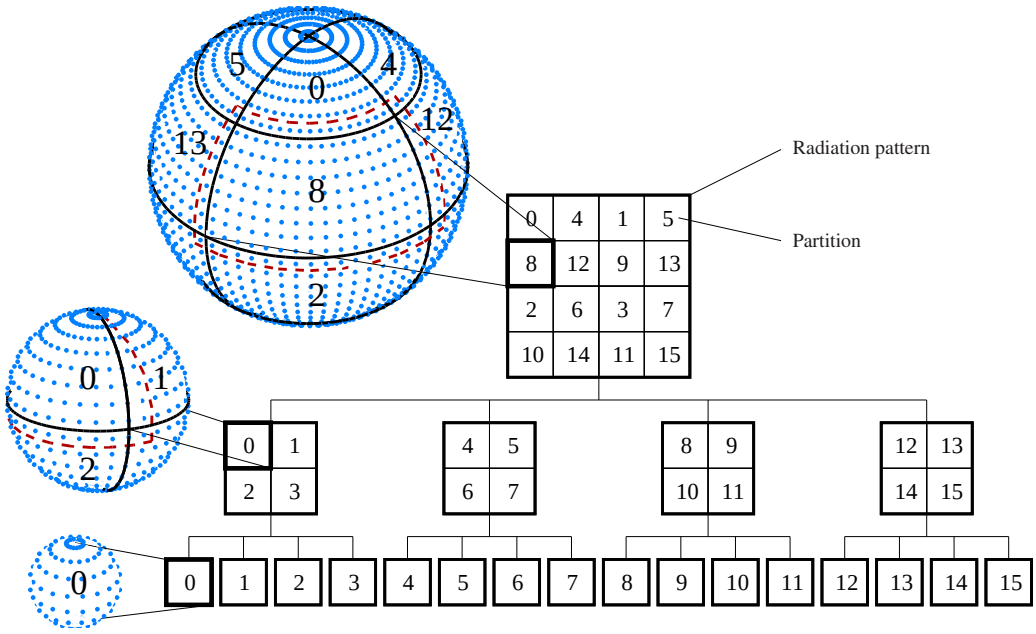


Fig. 3. Hierarchical scheme with a blockwise partitioning of the radiation pattern sampling points (B-HiP). MLFMA tree (right) and physical layout of the radiation pattern partitions on the sphere (left). Similar as Fig. 1, the (blue) dots denote sampling points, the solid lines mark the boundaries of the partitions and the numbers denote the process they are attributed to. Partitions held by the same process overlap as much as possible, reducing the required communication during repartitioning. The dashed (red) rectangle encompasses the sampling points that are required for local interpolation of that partition.

(left). This scheme again imposes a bottleneck. The top-level radiation patterns consist of $\mathcal{O}(N)$ sampling points, i.e. $\mathcal{O}(\sqrt{N})$ points along the azimuth times $\mathcal{O}(\sqrt{N})$ along the elevation direction. Clearly, for $P = \mathcal{O}(N)$ processes, distributing the radiation pattern along one dimension (i.e. elevation) only fails to attribute $\mathcal{O}(1)$ sampling points to each process. Indeed, eventually, P will exceed the number of sampling points along the elevation direction. Some processes will be attributed $\mathcal{O}(\sqrt{N})$ sampling points, whereas others will be attributed none. Hence, the hierarchical scheme with stripwise partitioning does not improve the worst-case complexity per process, compared to hybrid partitioning.

2) Blockwise: We propose a modification to the hierarchical scheme, where the radiation patterns are partitioned blockwise (B-HiP), i.e. *both* in azimuth (ϕ) and elevation (θ), as schematically shown in Fig. 1 (right). The partitions then consist of rectangular patches in the (θ,ϕ)-plane. Fig. 3 demonstrates the hierarchical blockwise scheme for three MLFMA levels.

The radiation patterns are uniformly sampled in θ and ϕ [19]. This yields a Cartesian grid of sampling points, which facilitates their partitioning in two dimensions. Because the number of partitions grows proportionally to the number of sampling points, each partition consists of $\mathcal{O}(1)$ sampling points.

At every level in the tree, the blockwise hierarchical scheme attributes $\mathcal{O}(1)$ sampling points to each process. Hence, the memory and computational complexity per level and per process is also $\mathcal{O}(1)$. We now prove that the communication per level and per process is also $\mathcal{O}(1)$.

- During the aggregation phase, the radiation patterns are repartitioned at every level. This means that approximately $\frac{3}{4}$ of the locally contained points are sent to other processes, yielding $\mathcal{O}(1)$ communication per process and per level. Similarly, the communication during the disaggregation phase is $\mathcal{O}(1)$.
- During the translation phase, interactions between boxes are evaluated. If the corresponding radiation patterns (or their partitions) are held by different processes, they need to be communicated. Because each process contains only $\mathcal{O}(1)$ boxes per level, and because the number of possible interactions for a box is bounded, the required communication per level and per process is $\mathcal{O}(1)$.
- In order to perform accurate local interpolation and anterpolation, sampling points near the boundaries of neighboring partitions (eight in the case of the blockwise partitioning) are required (see Fig. 1 right and Fig. 2(a)). Fig. 2(b) illustrates that the number of required neighboring source points (i.e. the input for the interpolation or anterpolation), on each side in the θ and ϕ direction, for a local interpolator is constant on every level. Hence, again only $\mathcal{O}(1)$ communication is required per process and per level.

III. Weak Scaling Analysis: Numerical Validation

In the previous section, we have theoretically investigated the weak scalability for the four data distribution schemes (SP, HyP, S-HiP and B-HiP) based on their asymptotic behavior for a high number of unknowns N and parallel processes P. In this section, we wish to a) validate the theoretically derived bounds and b) quantitatively assess each of the schemes for a *realistic* problem size and number of processes.

The previously described data partitioning schemes have been implemented in a generic parallel MLFMA framework [20] written in C/C++. Communication between the different processes is handled using the Message Passing Interface (MPI). To investigate the weak scalability, a sequence of

TABLE II
SIMULATION DETAILS: INCREASINGLY LARGER CUBES ARE HANDLED USING A PROPORTIONALLY INCREASING NUMBER OF PARALLEL PROCESSES.

simulation	number of processes P	cube edge size	number of unknowns N	number of levels L
S_1	4	$128 \cdot \lambda/10$	294 912	6
S_2	16	$256 \cdot \lambda/10$	1 179 648	7
S_3	64	$512 \cdot \lambda/10$	4 718 592	8
S_4	256	$1024 \cdot \lambda/10$	18 874 368	9
S_5	1024	$2048 \cdot \lambda/10$	75 497 4724	10
S_6	4096	$4096 \cdot \lambda/10$	301 989 888	11

six increasingly larger simulations (denoted as $S_i, i = 1 \ldots 6$) is considered. Each problem S_i contains exactly four times as many unknowns as S_{i-1}, while the number of parallel processes is also increased by a factor of four. The geometry consists of a perfectly electrically conducting (PEC) cube, illuminated by an incident plane wave (although it should be added that the type of excitation does not influence the weak scalability analysis). The details for each simulation are listed in Table II.

For all simulations, the relative precision for local interpolation was set to $\epsilon = 10^{-6}$, the size of the lowest-level box was 0.5λ. Single-precision calculations were used. For the HyP, the transition level was $\lceil \frac{L}{2} \rceil$, with L the number of MLFMA levels. For the S-HiP and B-HiP, spatial partitioning was used for the three lowest levels. For every next level, the number of partitions was increased by a factor of four.

The weak scalability is assessed by considering the *memory requirements* M_p for each process p individually. We excluded from M_p the memory required to store the matrices for the near interactions and lowest-level (dis)aggregations, because these contributions are identical for the four partitioning schemes. Among all processes, the process that has the highest amount of memory usage is selected:

$$M_{\max} = \max_{p=1\ldots P} M_p$$

Fig. 4 shows the average memory usage per MLFMA level (i.e. $M_{\max}/(L-2)$, as there are no translations, inter- and anterpolations at the two highest MLFMA-levels) for the different simulations and partitioning schemes. One can observe that for the spatial, hybrid and stripwise hierarchical scheme, certain processes exhibit a memory requirement that exceeds $\mathcal{O}(1)$. This is a manifestation of the fact that these schemes fail to attribute $\mathcal{O}(1)$ sampling points to each process at each level. For the blockwise hierarchical partitioning (B-HiP), however, the memory usage per process and per level, remains constant, which shows that the memory complexity per level and per process is indeed $\mathcal{O}(1)$, yielding a scalable data distribution scheme.

A few remarks are in order when interpreting the results. First, $M_{\max}$ only contains the contributions from the radiation patterns, translation operators, inter- and anterpolation matrices and communication buffers. The reason why near interactions and lowest-level (dis)aggregations were excluded from $M_{\max}$ is that they contribute in a significant, but constant way to the total memory requirements. The goal of this experiment is to validate the theoretically derived complexities

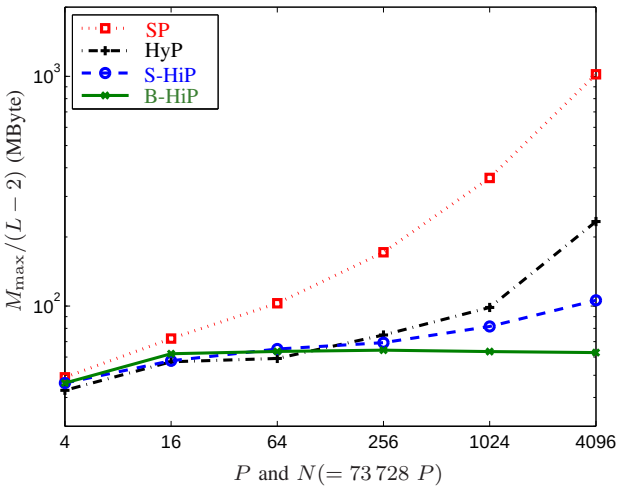


Fig. 4. Memory usage per level (maximum over all processes) as a function of the number of processes P and unknowns N.

from section II. A large constant contribution to a certain extent hides the presence of the higher-order terms.

Second, because a constant number of calculations are required per radiation pattern sampling point, the memory complexity is also representative for the computational complexity and hence the runtime. The time complexity cannot be lower than the memory complexity as every memory location has to be used at least once. Note that the largest cluster we have at our disposal contains 1024 CPU-cores and that the result on 4096 cores was obtained by oversubscribing the cluster, i.e. running 4 processes on a single core. Also note that in [17], the communication complexities were measured in a similar setup and shown to be $\mathcal{O}(1)$ as well.

Third, we make no statements as to which scheme has the highest parallel efficiency for a particular problem size and/or number of parallel processes. This depends on numerous factors, as listed in the introduction. However, the asymptotic analysis learns that for *sufficiently large* N and P, the B-HiP scheme will be most efficient. Algorithms with a lower computational complexity are usually more complex and their actual runtime can be dominated by fairly large prefactors. For example, the FFT-MLFMA algorithm has a higher computational complexity than the MLFMA [21], [22]. Nevertheless, the parallelization of the FFT-MLFMA algorithm is highly efficient (in a *strong scaling* sense) for *current* cluster sizes. Consequently, the largest integral equation problem so far was solved using a parallel FFT-MLFMA implementation.

We can now easily understand the bottlenecks in the different non-scalable schemes. The largest simulation S_6 was handled using $P = 4096$ processes. Table I reveals that only for level $l = 0$ to 4, the number of boxes $B_l > P$. Level 8 (i.e. the highest level that has actual MLFMA interactions) contains only 56 boxes. This means that only 56 processes out of 4096 actually contain a box on that level and that the other 98.6% of the processes are idle. In the spatial partitioning scheme, certain processes require 15 times more memory, compared to B-HiP.

For the HyP scheme, the transition level for simulation S_6 was $l = 6$. The highest level that uses SP (level $l = 5$) contains only 3 752 boxes, which can again not be uniformly distributed among $P = 4096$ processes. Even though this already significantly improves the load imbalance compared to pure SP, the transition level still imposes a bottleneck. Consequently, compared to the B-HiP scheme, the memory requirements are 4 times higher.

For the S-HiP scheme, a similar analysis can be made. A box on level 8 contains $Q_l = 4\,295\,380$ sampling points, or 1465 in elevation times 2932 in azimuth. Clearly, using the stripwise scheme from Fig. 1 (left), it is impossible to achieve a uniform partitioning. Roughly 35% of the processes are attributed a strip of 1×2932 sampling points, the other 65% are attributed none. From Fig. 2(b), it follows that 6 ($\epsilon = 10^{-3}$) to 16 ($\epsilon = 10^{-6}$) sampling points in elevation are required from adjacent partitions. This means that in order to perform accurate interpolations for a certain partition, data from *several* neighboring partitions are required, instead of only the two adjacent partitions as depicted in Fig. 1 (left). Clearly, such a communication pattern is undesirable. Even though the memory requirements are again lowered with respect to the HyP scheme, they are still approximately twice as high as for the B-HiP scheme when using 4096 processes.

For comparison, in B-HiP, each partition on level $l = 8$ contains roughly 1000 (23×46) sampling points. Every process contains a uniform amount of data and hence participates in the calculations. To perform interpolations and anterpolation on a certain partition, (portions of) no more than 8 neighboring boundary partitions are required.

We want to emphasize that the specific numbers attributed to the bottlenecks given above are specifically for problem S_6 using $P = 4096$ processes. For larger problem sizes and number of processes, these bottlenecks will become even more profound, and the relative difference to the proposed B-HiP scheme will become even larger.

IV. Numerical example

In order to demonstrate the correctness of our B-HiP implementation and provide for a runtime analysis, the simulation results of a canonical example (a PEC sphere) are compared to the analytical solution (the Mie series [23]). Similar to the previous section, we increased the number of CPU-cores by a factor of four ($P = 1, 4, \ldots, 1024$) and the diameter d of the sphere by a factor of two, resulting in an increase of the number of unknowns by roughly a factor of four. This way, the weak scaling behavior of the implementation and the accuracy of the simulations can be validated.

We considered a plane wave impinging on the PEC sphere with a diameter $d = 14.41 \cdot \sqrt{P} \cdot \lambda$, using a $\lambda/10$-discretization. The largest simulation on $P = 1024$ CPU-cores, depicted in Fig. 6(a), contained 200 120 454 unknowns. For all simulations, the Combined Field Integral Equation (CFIE) [24] with the combination coefficient $\alpha = 0.5$ was used. For the construction of the MLFMA-tree, a smallest boxsize of 0.2λ was chosen, resulting in a tree of 13 MLFMA-levels for the largest simulation. The iterative convergence precision was set to 10^{-3}. Each simulation was performed in single-precision on a cluster consisting of 64 machines each containing two 8-core Intel Xeon E5-2670 processors (1024 CPU-cores in total), using 32 GByte of RAM (or 2 GByte per core). The machines were connected using an Infiniband network.

TABLE III
Runtime per iteration and obtained precision with respect to the analytical solution for a PEC sphere with an increasing diameter simulated on an increasing number of CPU-cores.

(a) Runtime per iteration

P	number of levels L	average time per iteration	average time divided by $L-2$
1	8	1m 39s	16.50s
4	9	2m 08s	18.29s
16	10	2m 31s	18.88s
64	11	2m 49s	18.78s
256	12	3m 06s	18.60s
1024	13	3m 23s	18.45s

(b) Obtained precision

P	sphere diameter d	number of unknowns N	error w.r.t. Mie series (%)
1	$14.41 \cdot \lambda$	195 426	1.20
4	$28.82 \cdot \lambda$	781 098	0.96
16	$57.64 \cdot \lambda$	3 112 850	0.99
64	$115.28 \cdot \lambda$	12 502 692	1.02
256	$230.56 \cdot \lambda$	50 032 914	1.06
1024	$461.12 \cdot \lambda$	200 120 454	1.11

Table III(a) displays the runtime per iteration (averaged over 20 iterations) for the different simulations. With every step, both N and P are increased by a factor of four and one can observe that the time per iteration grows with roughly a constant contribution of approximately 20 seconds. This corresponds to the time needed to handle one extra MLFMA level in the tree and shows that the runtime indeed grows with the number of levels, i.e. $\mathcal{O}(\log N)$. The last column of Table III(a) shows the average runtime per level (only $L-2$ levels of the tree actually have MLFMA interactions). This result corresponds very well to the goal of the scalable parallel algorithm to obtain a $\mathcal{O}(1)$ computational complexity per level per process.

Apart from the scalability of the B-HiP, it is interesting to take a look at the communication map of the largest simulation on $P = 1024$ CPU-cores. Fig. 5 shows the communication between the different processes. A dark spot denotes the presence of communication between two processes. The communication map is very sparse, only 77 744 of the 1024^2 data points or 7.4% are nonzero, which is the result of the hierarchical partitioning scheme and the blockwise partitioning of the radiation patterns, which limits the number of neighboring partitions. From Fig. 5 one can also distinguish square-like clusters of communication. These are the result of the hierarchical partitioning of the levels.

Table III(b) shows the relative error in the radar cross section (RCS) with respect to the analytical solution. The error is given by

$$\frac{||f_\theta(\theta, \phi = 0)_{\text{simulation}} - f_\theta(\theta, \phi = 0)_{\text{analytical}}||_2}{||f_\theta(\theta, \phi = 0)_{\text{analytical}}||_2}$$

with $f_\theta(\theta, \phi = 0)$ the θ-component of the radiation pattern in the $\phi = 0$ plane. The obtained precisions around 1% are a

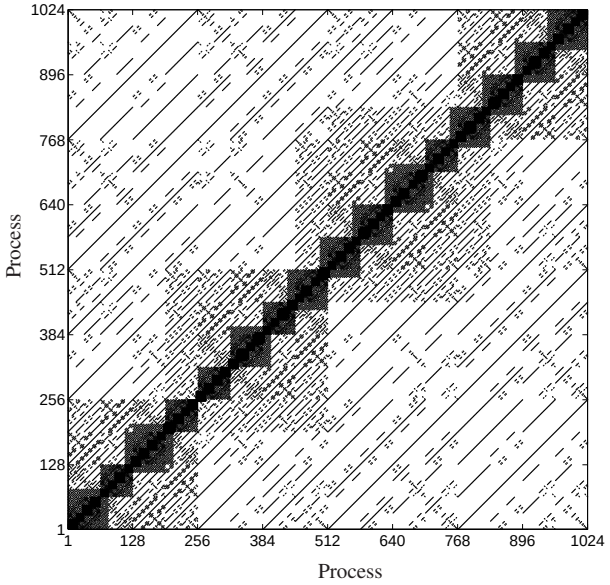
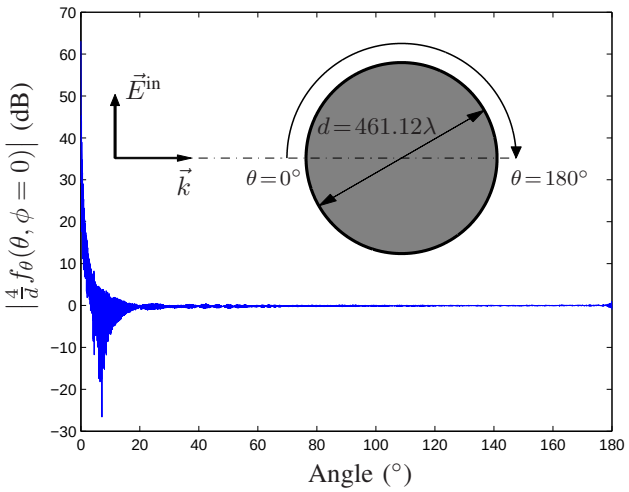


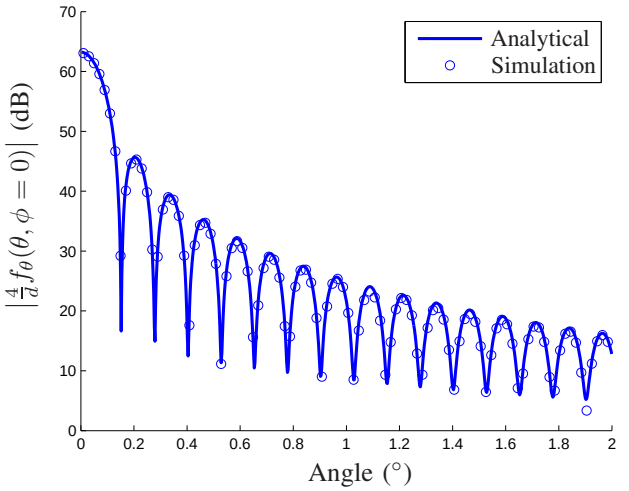
Fig. 5. Communication between the different processes. A dark spot means that there is communication between the processes, white corresponds to no communication.

typical result for a $\lambda/10$-discretization, similar as in [12], [13].

Fig. 6 shows the absolute value of $\frac{4}{d} f_\theta(\theta, \phi = 0)$, the θ-component of the normalized radiation pattern in the $\phi = 0$ plane, for the simulation of a PEC sphere with a diameter $d = 461.12\lambda$. Fig. 6(a) displays the full θ-range ($0° \ldots 180°$), discretized in 9026 sampling points or equivalently a resolution of approximately $0.02°$. Fig. 6(b), showing the backscattering direction for $\theta = 0° \ldots 2°$, confirms the good agreement between the computational values from our MoM-MLFMA implementation and the analytical solution of the Mie series, shown in Table III(b).



(a) Full θ-range ($0° \ldots 180°$) in 9026 sampling points.



(b) Backscattering direction ($\theta = 0° \ldots 2°$).

Fig. 6. The absolute value of the normalized radiation pattern $\frac{4}{d} f_\theta(\theta, \phi = 0)$ for a PEC sphere with a diameter $d = 461.12\lambda$.

V. Conclusion

In this paper a weak scaling analysis of the parallel MLFMA was performed, both theoretically and numerically. First, we examined three existing partitioning schemes, i.e. spatial (SP), hybrid (HyP) and hierarchical (S-HiP) and showed that they do not exhibit weak scalability. A modified hierarchical scheme was proposed, where the radiation patterns are partitioned blockwise (B-HiP) instead of stripwise. The complexity analysis shows that B-HiP does lead to a scalable algorithm. These theoretical results were experimentally verified for the different partitioning schemes. The results show that only the B-HiP scheme achieves an $\mathcal{O}(1)$ computational complexity per process and level, leading to a weakly scalable parallel MLFMA. Finally, a canonical example, where the number of unknowns and CPU-cores are proportionally increased up to more than 200 millions of unknowns and 1024 CPU-cores, was simulated using the B-HiP scheme. The time per matrix-vector multiplication per level also corresponded to an $\mathcal{O}(1)$ complexity and the results of the simulations were in agreement with the analytical solution.

Acknowledgment

The computational resources (Stevin Supercomputer Infrastructure) and services used in this work were provided by Ghent University, the Hercules Foundation and the Flemish Government – department EWI. The work of B. Michiels was supported by a doctoral grant from the Special Research Fund (BOF) at Ghent University. The work of I. Bogaert was supported by a post-doctoral grant from Research Foundation-Flanders (FWO-Vlaanderen).

References

[1] W.C. Chew, J. Jin, E. Michielssen and J. Song, "Fast and Efficient Algorithms in Computational Electromagnetics", Artech House, 2001.

[2] S. Velamparambil, J.M. Song, W.C. Chew and K. Gallivan, "ScaleME: a portable scaleable multipole engine for electromagnetic and acoustic integral equation solvers", IEEE Antennas and Propagation Society International Symposium, vol. 3, pp. 1774–1777, 1998.

[3] P. Havé, "A parallel implementation of the fast multipole method for Maxwell's equations", International Journal for Numerical Methods in Fluids, vol. 43, no. 8, pp. 839–864, Nov. 2003.

[4] F. Wu, Y. Zhang, Z.Z. Oo and E. Li, "Parallel Multilevel Fast Multipole Method for Solving Large-Scale Problems", IEEE Transactions on Antennas and Propagation Magazine, vol. 47, no. 4, pp. 110–118, 2005.
[5] J. Fostier and F. Olyslager, "An Asynchronous Parallel MLFMA for Scattering at Multiple Dielectric Objects", IEEE Transactions on Antennas and Propagation, vol. 56, no. 8, pp. 2346–2355, 2008.
[6] S. Velamparambil and W.C. Chew, "10 Million Unknowns: Is it that big?", IEEE Antennas and Propagation Magazine, vol. 45, no. 2, pp. 43–58, 2003.
[7] S. Velamparambil and W.C. Chew, "Analysis and Performance of a Distributed Memory Multilevel Fast Multipole Algorithm", IEEE Transactions on Antennas and Propagation, vol. 53, no. 8, pp. 2719–2727, 2005.
[8] Ö. Ergül, and L. Gürel, "Efficient Parallelization of the Multilevel Fast Multipole Algorithm for the Solution of Large-Scale Scattering Problems", IEEE Transactions on Antennas and Propagation, vol. 56, no. 8, pp. 2335–2345, 2008.
[9] X.-M. Pan and X.-Q. Sheng, "A Sophisticated Parallel MLFMA for Scattering by Extremely Large Targets", IEEE Antennas and Propagation Magazine, vol. 50, no. 3, pp. 129–138, 2008.
[10] V. Melapudi, B. Shanker, S. Seal and S. Aluru, "A Scalable Parallel Wideband MLFMA for Efficient Electromagnetic Simulations on Large Scale Clusters", IEEE Transactions on Antennas and Propagation, vol. 59, no. 7, pp. 2565–2577, 2011.
[11] Ö. Ergül and L. Gürel, "Hierarchical Parallelisation Strategy for Multilevel Fast Multipole Algorithm in Computational Electromagnetics", Electronics Letters, vol. 44, no. 1, pp. 3–4, 2008.
[12] Ö. Ergül and L. Gürel, "A Hierarchical Partitioning Strategy for an Efficient Parallelization of the MultiLevel Fast Multipole Algorithm", IEEE Transactions on Antennas and Propagation, vol. 57, no. 6, pp. 1740–1750, June 2009.
[13] Ö. Ergül and L. Gürel, "Rigorous Solutions of Electromagnetic Problems Involving Hundreds of Millions of Unknowns", IEEE Antennas and Propagation Magazine, vol. 53, no. 1, pp. 18–27, 2011.
[14] J. Fostier and F. Olyslager, "Provably Scalable Parallel Multilevel Fast Multipole Algorithm", Electronics Letters vol. 44, no. 19, pp. 1111–1112, 2008.
[15] J. Fostier and F. Olyslager, "Full-Wave Electromagnetic Scattering at Extremely Large 2-D Objects", Electronics Letters, vol. 45, no. 5, 2009.
[16] J. Fostier and F. Olyslager, "An Open-Source Implementation for Full-Wave 2D Scattering by Million-Wavelength-Size Objects", IEEE Antennas and Propagation Magazine, vol. 52, no. 5, pp. 23–34, 2010.
[17] B. Michiels, J. Fostier, I. Bogaert, P. Demeester and D. De Zutter, "Towards a Scalable Parallel MLFMA in Three Dimensions", 2011 Computational Electromagnetics International Workshop (CEM'11), Izmir, August 2011.
[18] B. Michiels, J. Fostier, J. Peeters, I. Bogaert and D. De Zutter, "Towards an Asynchronous, Scalable MLFMA for Three-Dimensional Electromagnetic Problems", 2011 International Conference on Electromagnetics in Advanced Applications (ICEAA 2011), Torino, September 2011.
[19] J. Sarvas, "Performing Interpolation and Anterpolation entirely by Fast Fourier Transform in the 3-D Multilevel Fast Multipole Algorithm", SIAM Journal on Numerical Analysis, 41(6):2180–2196, 2003.
[20] B. Michiels, J. Fostier, I. Bogaert and D. De Zutter, "A Generic Framework for the Parallel MLFMA", 2013 Applied Computational Electromagnetic Society (ACES 2013), Monterey, March 2013.
[21] C. Waltz, K. Sertel, M.A. Carr, B.C. Usner and J.L. Volakis, "Massively Parallel Fast Multipole Method Solutions of Large Electromagnetic Scattering Problems", IEEE Transactions on Antennas and Propagation, vol. 55, no. 6, pp. 1810–1816, 2007.
[22] J.M. Taboada, L. Landesa, F. Obelleiro, J.L. Rodriguez, J.M. Bertolo, M.G. Araujo, J.C. Mouriño and A. Gomez, "High Scalability FMM-FFT Electromagnetic Solver for Supercomputer Systems", IEEE Antennas and Propagation Magazine, vol. 51, no. 6, pp. 20–28, 2009.
[23] G. Mie, "Beiträge zur Optik trüber Medien, speziell kolloidaler Metallösungen", Annalen der Physik, vol. 25, no. 3, pp. 377–445, 1908.
[24] J.R. Mautz and R.F. Harrington, "H-Field, E-Field, and Combined-Field Solutions for Conducting Bodies of Revolution", Archiv für Elektronik und Übertragungstechnik, vol. 32, no. 4, pp. 157–164, April 1978.

Bart Michiels was born on October 30, 1986. He received the M.S. degree in engineering physics from Ghent University, Ghent, Belgium, in 2009. His M.S. thesis dealt with the simulation of large, broadband two-dimensional electromagnetic problems, using the Multilevel Fast Multipole Method (MLFMA). He is currently pursuing the Ph.D. degree in engineering physics at the Department of Information Technology (INTEC), Ghent. His research interests are numerical techniques and fast algorithms to solve full-wave electromagnetic problems and his current research deals with the parallelization of the MLFMA.

Jan Fostier (M'10) was born in 1982. He received the M.S. degree in Physical Engineering in 2005 and the Ph.D. degree in applied physics in 2009, both from Ghent University, Ghent, Belgium. In 2005, he joined the electromagnetics research group at the Department of Information Technology (INTEC) and in 2011 he was appointed assistant professor at the Internet Based Communication Networks and Services (IBCN) research group at the same department. His current research interests are numerical techniques, fast algorithms and distributed computing for bio-informatics and electromagnetics problems.

Ignace Bogaert (M'12) was born in Ghent, Belgium, in 1981. He received the M.S. degree in engineering physics from Ghent University, Ghent, Belgium, in 2004. After graduating, he joined the Electromagnetics Group of the Department of Information Technology (INTEC) at Ghent University, where he received his Ph.D. in applied physics in 2008. His research is supported by a postdoctoral grant from the Research Foundation-Flanders (FWO-Vlaanderen). His research interests include optimization problems and the modeling of various physical systems, with the emphasis on robustness, efficiency and accuracy.

Daniël De Zutter (F'00) was born in 1953. He received the M.Sc. degree in electrical engineering from the University of Gent in 1976. In 1981 he obtained a Ph.D. degree and in 1984 he completed a thesis leading to a degree equivalent to the French Aggrégation or the German Habilitation (both at the University of Ghent). He is now a full professor of electromagnetics. His research focusses on all aspects of circuit and electromagnetic modelling of high-speed and high-frequency interconnections and packaging, on Electromagnetic Compatibility (EMC) and numerical solutions of Maxwell's equations. As author or co-author he has contributed to more than 200 international journal papers (cited in the Web of Science) and 200 papers in conference proceedings. In 2000 he was elected to the grade of Fellow of the IEEE. He was an Associate Editor for the IEEE Microwave Theory and Techniques Transactions. Between 2004 and 2008 he served as the Dean of the Faculty of Engineering of Ghent University and is now the head of the Department of Information Technology.