Progress In Electromagnetics Research, PIER 101, 241–256, 2010

SUPERCOMPUTER AWARE APPROACH FOR THE SOLUTION OF CHALLENGING ELECTROMAGNETIC PROBLEMS

M. G. Araújo

Department Teoría do Sinal e Comunicacións
E.T.S.E. Telecomunicación
Universidade de Vigo
Vigo (Pontevedra) 36310, Spain

J. M. Taboada

Department Tecnologías de los Computadores y de las Comunicaciones
Escuela Politécnica
Universidad de Extremadura
Cáceres 10071, Spain

F. Obelleiro and J. M. Bértolo

Department Teoría do Sinal e Comunicacións
E.T.S.E. Telecomunicación
Universidade de Vigo
Vigo (Pontevedra) 36310, Spain

L. Landesa and J. Rivero

Department Tecnologías de los Computadores y de las Comunicaciones
Escuela Politécnica
Universidad de Extremadura
Cáceres 10071, Spain

J. L. Rodríguez

Department Teoría do Sinal e Comunicacións
E.T.S.E. Telecomunicación
Universidade de Vigo
Vigo (Pontevedra) 36310, Spain

Corresponding author: M. G. Araújo (martaga@com.uvigo.es).

Abstract—It is a proven fact that The Fast Fourier Transform (FFT) extension of the conventional Fast Multipole Method (FMM) reduces the matrix vector product (MVP) complexity and preserves the propensity for parallel scaling of the single level FMM. In this paper, an efficient parallel strategy of a nested variation of the FMM-FFT algorithm that reduces the memory requirements is presented. The solution provided by this parallel implementation for a challenging problem with more than 0.5 billion unknowns has constituted the world record in computational electromagnetics (CEM) at the beginning of 2009.

1. INTRODUCTION

Recent years have seen an increasing effort in the development of fast and efficient electromagnetic solutions with a reduced computational cost regarding the conventional Method of Moments. Among others, the Fast Multipole Method (FMM) [1] and its multilevel version, the MLFMA [2, 3] have constituted one of the most important advances in that context.

This development of fast electromagnetic solvers has gone hand in hand with the constant advances in computer technology. Due to this simultaneous growth, overcoming the limits in the scalability of the available codes became a priority in order to take advantage of the large amount of computational resources and capabilities that are available in modern High Performance Computer (HPC) systems. For this reason, works focused on the parallelization improvement of the Multilevel Fast Multipole Algorithm (MLFMA) [4–13] have gained interest in last years.

Besides, the FMM-Fast Fourier Transform (FMM-FFT) deserves be taken into account as an alternative to benefit from massively parallel distributed computers. This variation of the single-level FMM was first proposed in [14] as an acceleration technique applied to almost planar surfaces. Later on, a parallelized implementation was applied to general three-dimensional geometries [15]. The method uses the FFT to speedup the translation stage resulting in a dramatic reduction of the matrix-vector product (MVP) time requirement with respect to the FMM. Although in general the FMM-FFT is not algorithmically as efficient as the MLFMA, it has the advantage of preserving the natural parallel scaling propensity of the single-level FMM in the spectral (k-space) domain.

A large-scale problem involving more than 150 million unknowns has been successfully solved by the authors using a careful parallel implementation of the FMM-FFT in [16]. In this work, a hybrid

MPI/OpenMP parallel implementation of a "nested" version of the FMM-FFT is presented. In contrast with the FMM-FFT, it shows a slightly worse parallel performance in exchange for lower memory consumption. These features allow to deal with very large problems of hundreds of millions of unknowns, where the memory consumption may be a critical issue. The solution of a problem with more than a half billion of unknowns, included in the results of this paper, demonstrates that the nested algorithm constitutes a suitable tool for this kind of analysis when using modern parallel high performance supercomputers.

This paper is organized as follows: Section 2 reviews the main aspects of the FMM-FFT and its parallel implementation. Section 3 outlines the nested algorithm, some details about its parallelization and an assessment of the computational complexity. Section 4 presents some numerical results, among them a challenging one with more than 0.5 billion unknowns, and finally, the summary and conclusions are given in Section 5.

2. PARALLEL FMM-FFT ALGORITHM

2.1. FMM-FFT Algorithm

As it is shown in [14, 15], the FFT extension of the conventional FMM method allows to obtain a great reduction of the MVP CPU time with respect to the FMM. The method consists of employing the Fast Fourier Transform to speedup the translation stage in the framework of the FMM. The translation is the hardest stage with a computational cost of $O(M^2K)$, with M being the number of non-empty groups of the oct-tree decomposition and K the number of samples in the k-space. The FMM-FFT algorithm takes advantage of the regular group spacing provided by the oct-tree decomposition of the geometry. Since the radiation centers of the groups lie on a regular three-dimensional (3D) lattice, the translation operator between groups for a given direction in the k-space can be seen as a circular 3D convolution. Consequently, it can be evaluated simultaneously for all groups using a 3D FFT, hence reducing the computational complexity of the translation stage to $O(KQ\log Q)$. Q is the total number of groups (including empty groups), which fulfills the relationship $Q \propto M^{3/2}$ for "volumetric" arbitrary shaped geometries.

2.2. Parallelization Issues

Regarding the parallelization, as it is reported in our previous work of [16], we have concerned with hybrid parallel algorithms using the Message Passing Interface (MPI) for message passing paradigm

between distributed nodes, and OpenMP standard for threads inside each shared-memory node. This hybrid parallel programming allows to fit the architecture characteristics of large mixed memory computers (distributed clusters of shared-memory nodes). Four important issues must be considered to get a high scalability code with this approach: Work-load balancing among processors, data locality, memory footprint, and communication requirements. Using the original FMM or the FMM-FFT extension, the work-load can be equally distributed among processors while keeping good data locality (without memory footprint), by applying a k-space parallelization strategy. This efficient distribution is based on the fact that, both in FMM and FMM-FFT, each sample in k-space is completely independent of each other. Hence, the work-load for the far interactions of the MVP can be optimally distributed by splitting the far-fields among processors, instead of partitioning the spatial oct-tree. A similar distribution strategy is applied in [7, 12] for the parallelization of the shared levels of the MLFMA. For the near-field interactions and the iterative solver parallelization, however, the usual physical domain decomposition must be used. Then, the three stage parallelization strategy considered for the FMM-FFT in [16] is recalled here: (i) Distribution of far-fields among processors to account for the far-field interactions; (ii) distribution of oct-tree groups for near-field interactions; (iii) distribution of unknowns for the iterative solver (in our case we have used GMRES [17]).

Mainly, the strong points of this implementation are the clean and efficient way of addressing the translation stage and the fact that only a single communication step at the end of the MVP is needed, which indeed can be performed very efficiently in an almost negligible time (details can be looked up in [16]).

To test the parallel scalability of this implementation, a relatively small problem consisting of the scattering of a PEC sphere with about 10 million unknowns has been solved. In Fig. 1 we show the speed-up using up to 1,024 parallel processors. A high scalability behavior can be observed, demonstrating the ability of the FMM-FFT algorithm to take advantage of the availability of a large amount of distributed computational resources.

3. PARALLEL NESTED FMM-FFT ALGORITHM

For very large problems with several tens or hundreds of millions of unknowns, the memory requirements of the FMM-FFT algorithm can become a critical factor, especially the memory which is required to store the near-coupling blocks of the impedance matrix and the

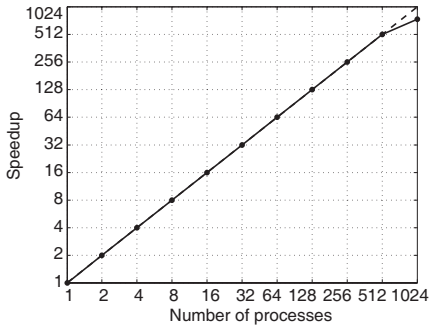

Figure 1. Parallel speed-up for the FMM-FFT algorithm. The problem is a PEC sphere with 10 million unknowns.

aggregation/disaggregation matrices containing the fields. With the aim of reducing the memory consumption, a variation of the FMM-FFT algorithm is proposed: the *nested* FMM-FFT.

3.1. Nested FMM-FFT

The nested algorithm begins by applying one or more refinement steps to the hierarchical oct-tree decomposition. The far-field interactions are still obtained at the coarsest level of the geometry partition. As it is indicated in the previous section, a global distributed FMM-FFT algorithm is employed to accomplish these far-field contributions with a parallelization strategy based on the distribution by k-space samples. However, the key element here is the treatment of the near-field interactions in the MVP, which leads to a drastic reduction of the memory consumption. Instead of employing the coarsest level, the near contributions are obtained at the finest oct-tree level by using one or more local shared memory FMM-FFT algorithms inside each computing node. This procedure is depicted in Fig. 2 for a bi-dimensional case. It can be observed that the near-field contributions in the coarsest level are obtained at the finest level by means of a nested FMM-FFT algorithm (with the usual stages of far, near and self-coupling contributions). Since the near-coupling blocks of the impedance matrix are now calculated and stored at the finest level of the oct-tree, a strong reduction of the required memory can be achieved with respect to the original FMM-FFT.

At this point, it must be noted that the pursued memory reduction is obtained using the above nested scheme without any penalty on the solver scalability. On the other hand, the nested scheme can be also applied to calculate the aggregation and disaggregation matrices

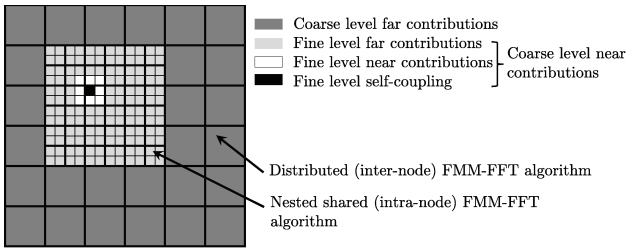

Figure 2. Graphic description of the nested FMM-FFT algorithm.

(as well as the radiated fields) at the finest level of the hierarchical decomposition, and then interpolate them to the coarsest level. This provides a further reduction of the memory consumption, but this reduction is obtained at the expense of additional communications during the MVP. The impact of this aspect in the scalability and the parallelization strategy of the nested algorithm are tackled next.

3.2. Parallelization Issues

The use of the nested algorithm for the computation of the aggregation and disaggregation matrices requires additional communications for the interpolation and anterpolation of fields between the finest and the coarsest level of the oct-tree. These communications have some impact on the scalability, as it will be shown next. However, we have addressed these all to all communications in a very efficient way by means of the asymmetric MPI collective communication operation `Alltoallw`.

Looking for maintaining the aforementioned objectives of workload balancing among processors, data locality, memory footprint and communication requirements, the parallelization strategy considered for the nested FMM-FFT includes the following stages: (i) For the far interactions, a mixed approach is applied by (i.1) distributing by groups at the finest level; and (i.2) distributing by field samples at the coarsest level. (ii) For the near interactions, a partition of work by groups at the finest level is applied. A well-balanced load distribution scheme has been considered [16]. (iii) The iterative solver is distributed by equal number of unknowns per processor. The use of a two-level scheme to obtain the far interactions requires the interpolation/anterpolation of outcoming/incoming fields across the two levels. On the other hand, the different partition of work, by groups and by fields, implies inter-node communications during the MVP.

Figure 3 shows the computation procedure adopted for the far-

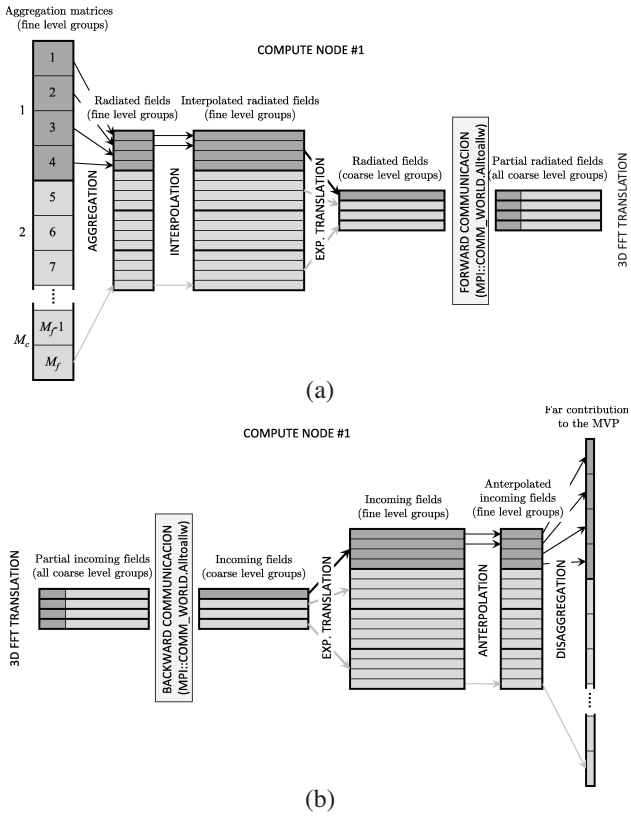

Figure 3. Far-field computation stages inside a shared memory node.

field interactions during a MVP inside each shared memory node. In light gray it is shown the total work of the MVP, while in dark gray it is represented the partition of work which is assigned to the first node (rank 0, other nodes are similar). Looking at Fig. 3(a), the work is distributed as follows:

(i) From the aggregation matrices at the finest level, we obtain the K_f samples of the outgoing radiated fields. These fields are calculated for the M_f/n groups assigned to this node (distribution by groups), where n is the number of nodes and the subscript f refers to the finest level.

(ii) The K_c samples of the outgoing radiated fields are obtained for the respective M_c/n groups, with the subscript c referring to the coarsest level. This is done throughout interpolation and shift (exponential translation) of the children groups to the center of

their parent groups. At this point, each node has the complete set of directions, K_c, for its assigned M_c/n groups.

(iii) An all to all communication is performed in order to obtain the partial K_c/n samples assigned to this node for all the M_c groups at the coarsest level (distribution by fields). This partial all to all communication is efficiently carried out in a single step by using the asymmetric MPI `Alltoallw` operation. Therein the management of the point to point communications is left to the MPI library, instead of using a rather complicated (and maybe inefficient) low-level communication scheme between all the distributed processes.

(iv) After the communication, each node performs the translation of its assigned field samples simultaneously for all groups using the 3D FFT convolution. As a result, each node has the partial K_c/n samples of the incoming fields into all the M_c groups.

The complementary procedure consisting of steps (iv) to (i) is applied to obtain the contributions to the MVP from the incoming fields. In this case, as it is represented in Fig. 3(b), the additional communications are required first, followed by the exponential translation to the corresponding "child" group centers, the anterpolation and the disaggregation stages. The `Alltoallw` high-level command makes possible to accomplish all the required communications between nodes without latency periods or explicit synchronization because of the efficient management provided by the MPI library.

Figure 4 helps to assess the scalability performance of the nested

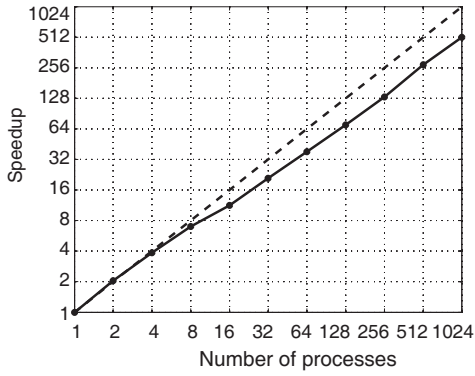

Figure 4. Parallel speed-up for the Nested FMM-FFT algorithm. The problem is a PEC sphere with 10 million unknowns.

method regarding the original FMM-FFT. It shows the speed-up obtained for the previous example of 10 million unknowns using the nested FMM-FFT algorithm. It can be observed that the scalability is not as good as in the FMM-FFT, but it still shows a good parallel performance. Partly, this is due to the fact that only two levels are needed for the application of the Nested FMM-FFT algorithm, in contrast with MLFMA which suffers from poor parallel scaling and requires sophisticated parallelization strategies.

3.3. Computational Complexity

To analyze the complexity of the nested FMM-FFT algorithm, the total number of fine and coarse cells, Q_f and Q_c, respectively, will be required additionally to the parameters K_f, K_c, M_f and M_c defined in the previous section. The objective is to estimate the computational cost and the memory requirements expressing them as a function of M_f and M_c, in order to later discuss the properly selection of both parameters.

Let us consider first the FFT-based translations cost at the coarsest level. If the relation $K_c \propto N/M_c$ [18] is taken into account, this cost can be written as $\mathcal{O}\left(K_c Q_c \log Q_c\right) \approx \mathcal{O}\left(N/M_c\ Q_c \log Q_c\right)$. Given that for volumetric geometries $Q_c \approx M_c^{3/2}$, an $\mathcal{O}(N\sqrt{M_c}\log M_c)$ cost is obtained. Regarding the aggregations/disaggregations and the near couplings, they are obtained in the method under analysis by means of $\mathcal{O}(M_c)$ FMM-FFT nested algorithms involving each one $\mathcal{O}(M_f/M_c)$ fine level cells. Then, the cost for the finest level $\mathcal{O}(M_c)$ algorithms can be divided up as it is indicated in Table 1, where also the rest of the significant terms have been included. The parameter n_l represents the number of levels between the finest and coarsest level and the relationships $K_f \propto N/M_f$ and $Q_f/M_c \approx (M_f/M_c)^{3/2}$ have been considered.

Looking for minimizing the near couplings cost of Table 1, small fine cells must be selected to achieve $M_f \propto N$ and then an $\mathcal{O}(N)$ cost. Observing the rest of the involved terms it can be seen that the selection of $M_c \approx \sqrt{N}$ leads to a dominant cost of $\mathcal{C}_{nested} = \mathcal{O}(N^{5/4}\log N)$.

As regards the memory requirements, those associated to the coarsest level translations are $\mathcal{O}(Q_c K_c) \approx \mathcal{O}(N\sqrt{M_c})$. This requirement and the memory cost estimated for the main finest level stages are gathered in Table 2. Taking into account the previous choices for both M_f and M_c, the dominant memory requirement is $\mathcal{M}_{nested} = \mathcal{O}(N^{5/4})$.

Due to the use of two different refinement levels in the nested

Table 1. Summary of the nested FMM-FFT numerical complexity.

Aggregations/disaggregations (finest level)	$\mathcal{O}(K_f N) \approx \mathcal{O}(N^2/M_f)$
Near couplings (finest level)	$\mathcal{O}(N^2/M_f)$
FFT Translations (finest level)	$\mathcal{O}\left(K_f \frac{Q_f}{M_c} \log\left(\frac{Q_f}{M_c}\right) M_c\right)$ $\approx \mathcal{O}\left(N\sqrt{\frac{M_f}{M_c}} \log\left(\frac{M_f}{M_c}\right)\right)$
FFT Translations (coarsest level)	$\mathcal{O}(N\sqrt{M_c} \log M_c)$
Interpolations	$\mathcal{O}(N n_l) \approx \mathcal{O}(N \log(M_f/M_c))$

Table 2. Summary of the nested FMM-FFT memory requirements.

Aggregations/disaggregations (finest level)	$\mathcal{O}(N^2/M_f)$
Near couplings (finest level)	$\mathcal{O}(N^2/M_f)$
FFT Translations (finest level)	$\mathcal{O}\left(\frac{M_f}{M_c} K_f\right) \approx \mathcal{O}(N/M_c)$
FFT Translations (coarsest level)	$\mathcal{O}(N\sqrt{M_c})$
MVP	$\mathcal{O}(K_f M_f) + \mathcal{O}(K_c Q_c)$ $\approx \mathcal{O}(N) + \mathcal{O}(N\sqrt{M_c})$

algorithm, it is possible to do a properly selection of the two groups size to obtain an optimum balance of the computational cost and the memory requirements.

At this point, it is worth mentioning that if $M_c < \sqrt{N}$ is selected, the large size of the coarsest level cells implies a drastic reduction of the empty cells number, wich results in the relationship $Q_c \to M_c$. Under these conditions, an $\mathcal{O}(N)$ memory requirement can be achieved at the expense of the CPU time increase.

In order to illustrate the cost estimations detailed above, several executions of a sphere with different number of unknowns have been carried out using the nested FMM-FFT algorithm. Fig. 5 shows the computational cost result that fits the curve $\mathcal{O}(N^{5/4} \log N)$ according to the previous estimation for $\mathcal{C}_{nested}$. Regarding the memory consumption shown in Fig. 6, it can be observed that the nested method behavior is described by the $\mathcal{O}\left(N^{5/4}\right)$ curve, which was the estimation for $\mathcal{M}_{nested}$.

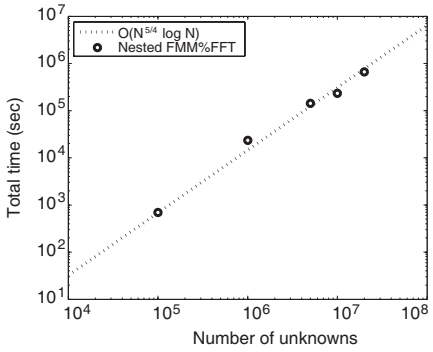

Figure 5. Nested FMM-FFT computational cost for a sphere with different number of unknowns.

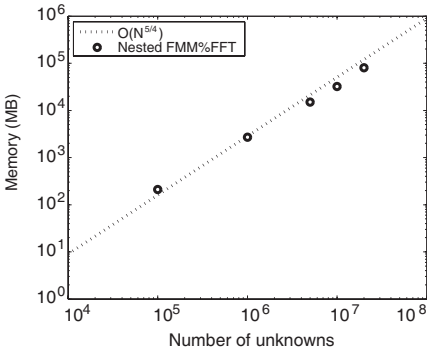

Figure 6. Nested FMM-FFT memory requirements for a sphere with different number of unknowns.

4. NUMERICAL EXAMPLES

The nested FMM-FFT algorithm has been used to solve two large-scale problems included in this section. Regarding the electromagnetic formulation, both examples have been addressed with an Electric Field Integral Equation (EFIE) based Method of Moments formulation, in which the well-known Rao-Wilton-Glisson (RWG) basis functions [19] have been applied both in the discretization of the geometry and the Galerkin's testing procedure. No preconditioning has been considered for the numerical examples of this work.

The first result of more than 150 millions of unknowns was carried out using the Lusitania supercomputer of the Centro Extremeño de Investigación, Innovación Tecnológica y Supercomputación (CENITS). Lusitania is made up of 2 HP Integrity SuperDome SX2000 nodes with 64 dual core Itanium2 Montvale processors at 1.6 GHz (18 MB cache). Altogether a RAM memory of 2,000 GB is available. The operating system is Linux SLES 10 and for all the results in this paper, we have used the Intel C_{++} Compiler version 11.0.069, and Intel MPI version 3.2.0.011 for communications. For matrix/vector linear algebra operations we have used the Intel Cluster MKL version 10.0.2.018. The first analysis consists of a car of 154,927,736 unknowns. A front incident plane wave horizontally polarized has been considered. The problem has been solved using 5 GMRES outer iterations and a restart parameter of 50, obtaining a residue lower than $3 \cdot 10^{-2}$. A total memory of 630 GB was required and the solution time was of 29 hours. The configuration parameters of the method and the technical data

Table 3. Technical data for the solution of a 150 million of unknowns problem.

Frequency	48 GHz
Number of unknowns	154, 927, 736
Groups dimensions (fine/coarse level)	$0.25\lambda/4\lambda$
Multipole terms (fine/coarse level)	5/45
Number of total/non-empty groups (fine level)	1, 935, 136, 755/7, 382, 859
Number of total/non-empty groups (coarse level)	487, 060/29, 794
Num. of nodes/processors per node	2/128
Min./max. peak memory in node	309 GB/321 GB
Total memory	630 GB
Num. of iterations/GMRES restart	5/50
Setup/solution time	1.8 h/29 h

corresponding to this result are gathered in Table 3. The numerical prediction obtained for the bistatic RCS is shown in Fig. 7.

The results for the second example were performed using the HPC supercomputer Finis Terrae, recently installed in the Supercomputing

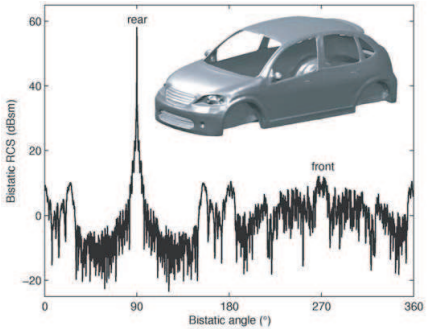

Figure 7. Bistatic RCS of more than 150 million of unknowns car.

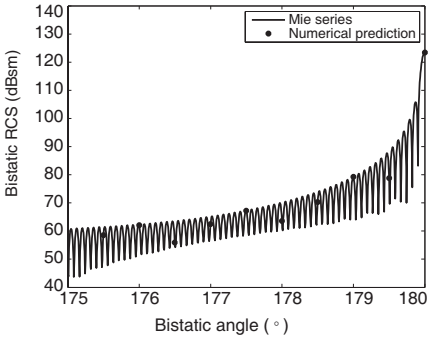

Figure 8. Bistatic RCS of more than 0.5 billion of unknowns PEC sphere.

Center of Galicia (CESGA). Finis Terrae consists of 142 cc-NUMA HP Integrity rx7640 with 8 dual core Intel Itanium2 Montvale processors at 1.6 GHz with 18 MB L_3 cache and 128 GB of memory. Totally, they sum more than 2,500 cores and 19,000 GB of memory, being one of the computers with the best ratio memory/processor in the world. The nodes are interconnected through a high efficiency Infiniband network (4xDDR), and the operating system is Linux SLES 10. The example that has been run in this supercomputer consists of the challenging analysis of a PEC sphere with 728.36λ diameter and 500,159,232 unknowns. The solution required the use of 64 nodes (involving a total of 1,024 processors) and 6 TB of memory. A total of 10 GMRES iterations with restart 10 have been required to obtain a residual error below $5 \cdot 10^{-2}$. The setup time was about 5 hours, while the iterative solution took less than 26 h. An excellent agreement between the numerical result obtained for the bistatic RCS of the sphere and the analytical solution provided by the Mie series can be observed in Fig. 8. Technical data of the problem and the solution are summarized in Table 4.

Table 4. Technical data for the solution of a challenging electromagnetic problem with 0.5 billion unknowns.

Sphere diameter	728.36λ
Frequency	300 MHz
Number of unknowns	500,159,232
Groups dimensions (fine/coarse level)	$0.5\lambda/4\lambda$
Multipole terms (fine/coarse level)	6/44
Number of total/non-empty groups (fine level)	3,092,990,993/9,558,160
Number of total/non-empty groups (coarse level)	6,128,487/155,391
Num. of nodes/processors per node	64/16
Min./max. peak memory in node	89.2 GB/99.9 GB
Total memory	6 TB
Num. of iterations/GMRES restart	10/10
Setup/solution time	5 h/26 h

5. CONCLUSIONS

An efficient parallelization of the FMM-FFT algorithm has been implemented, exploiting its natural high scaling properties to benefit from the availability of massively distributed supercomputers. In order to accomplish the analysis of very large-scale scattering problems, a nested configuration of the method that improves the memory requirements has been proposed. The computational cost and the memory requirements associated to the nested algorithm are $\mathcal{O}(N^{5/4} \log N)$ and $\mathcal{O}(N^{5/4})$, respectively. Optimal load balance and data locality have been obtained, while minimizing the memory footprint and inter-processor communication requirements. This guarantees the efficient use of large amounts of parallel processors. The aforementioned advances, combined with the computational resources provided by the supercomputer Finis Terrae, have allowed us to address the electromagnetic scattering of a 728.36λ diameter PEC sphere involving more than 500 million of unknowns at the beginning of 2009. This world record challenge has resulted in various international awards in supercomputing in 2009 (International PRACE Award and Itanium Innovation Award).

ACKNOWLEDGMENT

This work was supported by the Spanish Government (projects TEC2008-06714-C02-01, TEC2008-06714-C02-02, and CONSOLIDER-INGENIO2010 CSD2008-00068), and by Xunta de Galicia (project INCITE08PXIB322250PR). The authors would like to thank the Supercomputing Center of Galicia (CESGA) for their support in the use of Finis Terrae supercomputer and the Centro Extremeño de Investigación, Innovación Tecnológica y Supercomputación (CENITS) for their support in the use of Lusitania supercomputer, which were used to run the simulations.

REFERENCES

1. Coifman, R., V. Rokhlin, and S. Wanzura, "The fast multipole method for the wave equation: A pedestrian prescription," *IEEE Antennas Propagat. Mag.*, Vol. 35, No. 3, 7–12, Jun. 1993.
2. Song, J. M. and W. C. Chew, "Multilevel fast multipole algorithm for solving combined eld integral equations of electromagnetic scattering," *Microw. Opt. Tech. Lett.*, Vol. 10, 14–19, Sep. 1995.
3. Song, J. M., C. C. Lu, W. C. Chew, and S. Lee, "Fast Illinois solver

code (FISC)," *IEEE Antennas Propag. Mag.*, Vol. 40, No. 3, 27–34, Jun. 1998.

4. Velamparambil, S., J. E. Schutt-Aine, J. G. Nickel, J. M. Song, and W. C. Chew, "Solving large scale electromagnetic problems using a linux cluster and parallel MLFMA," *IEEE Antennas Propag. Soc. Int. Symp.*, Vol. 1, 636–639, 1999.
5. Velamparambil, S., W. C. Chew, and J. M. Song, "10 million unknowns: Is it that big?," *IEEE Antennas Propagat. Mag.*, Vol. 45, No. 3, 43–58, Apr. 2003.
6. Sylvand, G., "Performance of a parallel implementation of the FMM for electromagnetics applications," *Int. J. Numer. Meth. Fluids*, Vol. 43, 865–879, 2003.
7. Velamparambil, S. and W. C. Chew, "Analysis and performance of a distributed memory multilevel fast multipole algorithm," *IEEE Trans. Antennas Propagt.*, Vol. 53, No. 8, 2719–2727, 2005.
8. Pan, X. M. and X. Q. Sheng, "A highly efficient parallel approach of multi-level fast multipole algorithm," *Journal of Electromagnetic Waves and Applications*, Vol. 20, No. 8, 1081–1092, 2006.
9. Wang, P. and Y. J. Xie, "Scattering and radiation problem of surface/surface junction structure with multilevel fast multipole algorithm," *Journal of Electromagnetic Waves and Applications*, Vol. 20, No. 15, 2189–2200, 2006.
10. Gürel, L. and Ö. Ergül, "Fast and accurate solutions of extremely large integral-equation problems discretised with tens of millions of unknowns," *Electronics Letters*, Vol. 43, No. 9, 499–500, Apr. 2007.
11. Pan, X.-M. and X. X.-Q. Sheng, "A sophisticated parallel MLFMA for scattering by extremely large targets," *IEEE Antennas Propagat. Mag.*, Vol. 50, 129–138, Jun. 2008.
12. Ergül, Ö. and L. Gürel, "Efficient parallelization of the multilevel fast multipole algorithm for the solution of large-scale scattering problems," *IEEE Trans. Antennas Propagt.*, Vol. 56, 2335–2345, Aug. 2008.
13. Gürel, L., Ö. Ergül, A. Ünal, and T. Malas, "Fast and accurate analysis of large metamaterial structures using the multilevel fast multipole algorithm," *Progress In Electromagnetics Research*, PIER 95, 179–198, 2009.
14. Wagner, R., J. M. Song, and W. C. Chew, "Montecarlo simulation of electromagnetic scattering from two-dimensional random rough surfaces," *IEEE Trans. Antennas Propagt.*, Vol. 45, No. 2, 235–

245, Feb. 1997.

15. Waltz, C., K. Sertel, M. A. Carr, B. C. Usner, and J. L. Volakis, "Massively parallel fast multipole method solutions of large electromagnetic scattering problems," *IEEE Trans. Antennas Propagt.*, Vol. 55, No. 6, 1810–1816, Jun. 2007.
16. Taboada, J. M., 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 Propagat. Mag.*, Dec. 2009.
17. Saad, Y. and M. Schultz, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems," *SIAMJ. Sci. Statist. Comput.*, Vol. 7, 856–869, 1986.
18. Rodriguez, J. L., M. G. Araujo, J. M. Taboada, L. Landesa, and F. Obelleiro, "On the use of the singular value decomposition in the fast multipole method," *IEEE Trans. Antennas Propagt.*, Vol. 56, No. 8, 2325–2334, Aug. 2008.
19. Rao, S. M., D. R. Wilton, and A. W. Glisson, "Electromagnetic scattering by surfaces of arbitrary shape," *IEEE Trans. Antennas Propagat.*, Vol. 30, 409–418, May 1982.