Massively parallelization of multilevel fast multipole algorithm for 3-D electromagnetic scattering problems on SW26010 many-core Cluster

Xin-Duo Liu\textsuperscript{1}, Wei-Jia He\textsuperscript{1*}, Ming-Lin Yang\textsuperscript{1}, Xin-Qing Sheng\textsuperscript{1}

\textsuperscript{1}Center for Electromagnetic Simulation, Beijing Institute of Technology, Beijing, 100081, China.

\textsuperscript{*}Corresponding author(s). E-mail(s): heweijia@bit.edu.cn;

\textbf{Abstract}

This paper presents a massively parallel approach of the multilevel fast multipole algorithm (PMLFMA) on homegrown many-core SW26010 cluster of China, noted as (SW-PMLFMA), for 3-D electromagnetic scattering problems. In this approach, the multilevel fast multipole algorithm (MLFMA) octree is first partitioned among management processing elements (MPEs) of SW26010 processors following the ternary partitioning scheme using the message passing interface (MPI). Then the computationally intensive parts of the PMLFMA on each MPI process, matrix filling, aggregation and disaggregation, etc., are accelerated by using all the 64 computing processing elements (CPEs) in the same core group of the MPE via the Athread parallel programming model. Different parallelization strategies are designed for many-core accelerators to ensure a high computational throughput. In coincidence with the special characteristic of local Lagrange interpolation, the compressed sparse row (CSR) and the compressed sparse column (CSC) sparse matrix storage format is used for storing interpolation and antepolation matrices respectively, together with a specially designed cache mechanism of hybrid dynamic and static buffers using the scratchpad memory (SPM) to improve data access efficiency. Numerical results are included to demonstrate the efficiency and versatility of the proposed method. The proposed parallel scheme is shown to have excellent speedup.

\textbf{Keywords:} Multilevel fast multipole algorithm, distributed memory parallelization, many-core acceleration, electromagnetic scattering, sw26010 processor
1 Introduction

The multilevel fast multipole algorithm (MLFMA) is a powerful and essential tool in computational electromagnetics (CEM), particularly for large-scale objects in stealth and anti-stealth design, as documented in various studies [1–3]. To extend the solvable problem scale of MLFMA, researchers have proposed several approaches for the distributed memory parallelization of MLFMA over the last few decades. These methods involve the utilization of Message-Passing Interface (MPI) or hybrid MPI-OpenMP parallel implementations [4–18]. With these efficient parallel implementations of MLFMA, problems with over ten billion unknowns can now be addressed [17].

To improve the computational capability of pure CPU systems, which is limited by the power and heat dissipation, heterogeneous multi-core systems with many-core accelerator become the developing trend, the CPU and graphics processing unit (GPU), the CPU and Intel Xeon Phi Coprocessors (MIC), the homegrown many-core processor Sunway SW26010, etc. As the first super computer system with peak performance greater than 100P in the world, the computing performance and power efficiency of Sunway Taihulight are outstanding [19, 20]. Compare with MIC and GPU, the SW26010 processor of Sunway Taihulight also has several unique hardware features, such as the utilization of scratchpad memory (SPM) for each computing processing element (CPE), direct memory access (DMA) to transfer data between the main memory and the SPM, and the design of register communication channels. Due to the challenges in many-core parallel programming on SW26010 processor brought by its special architecture, compared with progresses on GPU accelerated parallel implementation of MLFMA have been made [21–26], fewer successful many-core implementation of the MLFMA in CEM community has been proposed [27].

In this article, we extend our prior work in [27] to address electrically large-scale scattering problems through the development of a massively parallel, distributed memory, many-core accelerated parallel MLFMA algorithm on SW26010 processors, denoted as SW-PMLFMA. The proposed SW-PMLFMA algorithm partitions the MLFMA octree tree among processes using a ternary partitioning scheme [17]. Each MPI process’s computationally intensive components are then accelerated using the CPEs in the same core groups (CG). We employ a hierarchical parallelization strategy that simultaneously partitions both boxes and planewaves to ensure even workload distribution among all CPEs on higher MLFMA levels. We use the compressed sparse row (CSR) and compressed sparse column (CSC) sparse matrix storage formats for interpolation and anterpolation matrices, which are well-suited to local Lagrange interpolation. To enhance data access efficiency in interpolation and anterpolation operations, we designed a hybrid dynamic and static buffer cache scheme. Further optimizations, such as the double buffering scheme and the structure of array (SoA) interface, are employed to improve the algorithm’s performance. We demonstrate the efficacy of the proposed SW-PMLFMA algorithm through various simulation results, showing that it achieves high efficiency without sacrificing accuracy.

The structure of this article is as follows: Section 2 provides an overview of the MLFMA and its ternary parallelization approach, while Section 3 describes the architecture of the SW26010 many-core processor and the Athread programming model.
Section 4 details the implementation of the SW-PMLFMA algorithm. In Section 5, we present the results of our numerical simulations, and Section 6 concludes the paper.

2 An outline of MLFMA for 3-D electromagnetic scattering and its ternary parallelization approach

The surface integral equation method and its multi-level fast multipole algorithm (MLFMA) acceleration are briefly reviewed in this section. For a general three-dimensional (3-D) object irradiated by an incident field \( \mathbf{E}_i, \mathbf{H}_i \), the combined field integral equation (CFIE) can be obtained as [3]:

\[
\alpha \hat{n} \times \eta_0 \mathcal{L}(\mathbf{J}) \times \hat{n} + (1 - \alpha) \hat{n} \times \eta_0 [\mathcal{K}(\mathbf{J}) + 0.5 \hat{n} \times \mathcal{I}] = -\alpha \hat{n} \times \mathbf{E}^{inc} \times \hat{n} - (1 - \alpha) \eta_0 \hat{n} \times \mathbf{H}^{inc}
\] (1)

where \( \alpha \) denotes the combined coefficient, \( \hat{n} \) denotes the unit normal vector, \( \eta_0 \) is the impedance in free space. \( \mathcal{L} \) and \( \mathcal{K} \) are integral-differential operators defined by:

\[
\mathcal{L}(\mathbf{J}) := -jk_0 \int_{S'} \left( \mathcal{I} + \frac{1}{k_0^2} \nabla \nabla \cdot \right) \mathbf{J}(r') \mathbf{G}(r, r') dS'
\] (2)

\[
\mathcal{K}(\mathbf{J}) := \text{P.V.} \int_{S'} \nabla \mathbf{G}(r, r') \times \mathbf{J}(r') dS'
\] (3)

where \( \mathbf{G}(r, r') = \left( e^{-jk_0 R} / 4\pi R \right) \) is the Green’s function in free space with \( R = |r - r'|, k_0 = 2\pi / \lambda_0 \). \( \mathcal{I} \) denotes the identity operator, \( \mathbf{J} \) represents the equivalent electric currents, and P.V. stands for the Cauchy principal value integration.

Following the procedure of method of moments (MoM), by using the Rao–Wilton–Glisson (RWG) functions as the basis functions, and applying the Galerkin’s method, a system of linear equations can be obtained:

\[
[Z_{mn}] \{J_n\} = \{f_m\}
\] (4)

with

\[
Z_{mn} = \int_S \mathbf{g}_m \cdot [\hat{n} \times \eta_0 \mathcal{L}(\mathbf{g}_n) \times \hat{n}] dS
\]

\[
+ (1 - \alpha) \int_S \mathbf{g}_m \cdot \hat{n} \times \eta_0 [\mathcal{K}(\mathbf{J}) + 0.5 \hat{n} \times \mathcal{I}] dS
\] (5)

\[
f_m = \int_S \mathbf{g}_m \cdot [-\alpha \hat{n} \times \mathbf{E}^{inc} \times \hat{n} - (1 - \alpha) \eta_0 \hat{n} \times \mathbf{H}^{inc}] dS
\] (6)

where \( \mathbf{g} \) represents the RWG basis function. The full matrix equation system (4) can be solved iteratively by using Krylov subspace methods such as the generalized minimal residual method (GMRES), with MLFMA to accelerate the matrix-vector multiplication with a complexity of \( O(N \log N) \).

When solving matrix equation system (4) iteratively, MLFMA can be employed to accelerate matrix-vector multiplications. The interaction can be separated into near field and far field interactions. The near field is calculated using conventional MoM, while the far field is computed in a group-wise manner by traversing the MLFMA.
The octree is constructed by recursively bisecting the dimensions of a cubic box until the size of the box on the lowest level is below a given threshold. The calculation of far field interaction involves three operators: aggregation, translation, and disaggregation. In contrast to FMM for Laplace equations, where the number of plane waves in a box remains constant, MLFMA for electromagnetic scattering problems solving Helmholtz equation has a twofold increase in the number of planewaves for a parent box at approximately the same rate in the $\theta$- and $\varphi$- directions as its child boxes. In the aggregation stage, plane waves from child boxes at the lower level are interpolated and centrally shifted to obtain higher level plane waves of their parent boxes. In the disaggregation stage, incoming plane waves at the centers of boxes are anterpolated and shifted to the centers of their child boxes at the lower level. During disaggregation, translation operations of plane waves at the same level are required. The translation operators used here are diagonal in nature [2].

The key in designing a scalable parallelization approach of MLFMA is how to partitioning the weighted MLFMA tree among processes properly. In distributed memory parallelization of MLFMA, workload of each MLFMA level can be partitioned among MPI processes by boxes, by planewave directions, or both two simultaneously. The use of different partitioning strategies results in different parallelization approaches, the simple, the hybrid [4], the hierarchical [8], and the ternary [17]. The ternary parallelization approach integrates the advantages of available partitioning strategies and resolves the disadvantage of the others. It achieves a parallel efficiency as high as the hierarchical parallelization approach, while significantly weakens the limitation of the number of MPI processes used. In the ternary parallelization approach, several higher levels of the MLFMA tree are termed as plane wave partitioning (PWP) levels, on which each box partitions all its plane waves evenly on MPI processes along only the $\theta$-direction. Lower levels are termed as box partitioning (BP) levels, boxes of the same level are partitioned among MPI processes. The rest levels are hierarchical structure partitioning (HSP) levels, which switch the partitioning patterns of PWP and HSP by partitioning boxes into groups, and then partitioning planewaves in each box on a group of MPI processes along the $\theta$-direction. Readers can refer to [17] for technical details of the ternary parallelization approach of MLFMA.

### 3 SW26010 many-core architecture and the Athread programming model

In this article, the authors provide an overview of the SW26010 many-core architecture and the Athread programming model before delving into the details of the SW-PMLFMA. The SW26010 processor is heterogeneous and distinct from existing pure CPU, CPU-MIC, and CPU-GPU architectures. It comprises four CGs, each of which includes one management processing element (MPE) and one CPE cluster of 64 CPEs. Thus, a single SW26010 processor has four MPEs and 256 CPEs. The four CGs are interconnected by an on-chip network and typically used as four independent nodes. Additionally, each CPE has two execution pipelines, and its computational power is approximately half that of the MPE. Furthermore, the SW26010 processor has two types of memory spaces: 8GB main memory for each MPE and a local 64 KB
SPM for each CPE. The main memory can be accessed directly by all CPEs through global load/store with a bandwidth of 1.45GB/s. Data can also be transferred into the SPM of a CPE with a bandwidth of approximately 22.6GB/s via DMA channels. Additionally, the DMA operation is asynchronous, enabling users to implement hidden communication operations easily. Utilizing the SPM fully and reducing data communications are critical in SW26010 many-core acceleration computation.

![Fig. 1 SW26010 many-core architecture](image)

For SW26010, MPI and Athread parallel programming models are used for different aims. MPI is a widely-used, standardized, and portable message-passing interface that facilitates the management of parallel computations across diverse computational grids. On the other hand, Athread is a directive-based, user-driven, and performance-portable parallel programming model that leverages the parallelism of many-core processors. It enables the use of normal fork and join parallelism with up to 64 CPEs in the same CG, with one thread per CPE. In addition, it offers a set of DMA operation interfaces. The processing on the CPEs can be executed asynchronously to the MPE, and a hybrid MPI and Athread parallel programming model using the CPU and GPU asynchronous computing pattern is depicted in Figure 2. The serial phase of the program on each MPI process is first executed on the MPE, followed by the parallel phase where CPEs in the same CG undertake a computation-intensive task. Upon completion of the parallel work by every CG, the MPE host process resumes runtime and executes instructions in the serial phase.

4 Many-core acceleration implementation of the ternary PMLFMA

The MLFMA ternary parallelization approach involves a two-part computation process, namely the setup and the iteratively solution parts. During the setup stage, the near-field matrix, as well as the aggregation/disaggregation matrix at the lowest
level, interpolation/interpolation, and translation matrix at each level, are explicitly distributed and stored. In the iteratively solution procedure, each MVM computation is further divided into near and far field interactions. The near field interaction involves a sparse matrix-vector multiplication (SpMV), while the far field interaction comprises three stages, namely, the aggregation, the translation, and the disaggregation. To accelerate the computation on SW26010 processors, we propose a many-core accelerated massively parallel MLFMA, which we abbreviate as SW-PMLFMA. This approach is implemented using a hybrid MPI and Athread programming model. The execution order of the different stages of the SW-PMLFMA is illustrated in Figure 3. In the subsequent sections, we provide a detailed description of the primary stages of the proposed approach.

4.1 Setup Stage

At the very beginning of the program, preparation tasks include the auxiliary tree based parallel mesh refinement, the constructing and partitioning of the MLFMA octree, are executed by using only MPE. After the preparation tasks are completed, matrices filling is done with the aid of CPEs. Since the total number of CPEs in each CG is only 64, the idea of ‘one CPE per observer’ is used. The observer stands for a near box pair for near field matrix filling, a box for the aggregation/disaggregation matrix filling, a box pair for the translation matrix filling, etc. Take near field matrix
filling as an example, there are far more boxes than CPEs at all the lowest MLFMA level. Hence when partitioning workload among CPEs during near field matrix filling, computation associated with a box is assigned to a CPE. During the execution, each CPE fetches the required mesh data for RWG basis functions in a near box pair, then calculates corresponding nonzero entries in near-field matrix following the procedure of MoM. As studied in [27], the mesh data need to be preprocessed based on the structure of array to make the data in the order of calculation, thereby significantly reduces the total times of accessing global memory and in turn improves overall computation efficiency. The double buffering scheme proposed in [27] is also adopted to make data transition between main memory and the SPM overlapped CPE’s computation. Similar optimizations are also required for these arrays used for filling far field matrices, the aggregation/disaggregation matrix, the interpolation/antepolation matrix, the translation matrix, etc.

Since the SPM of each CPE in SW26010 is very small, frequent data transferring is inevitable. The storage formats of different matrices should be carefully designed to improve data access efficiency in the iterative solution procedure through DMA. When local Lagrange interpolation is adopted to match different sampling rate of planewaves between successive levels, each target point in the fine grid has contributions from at most 16 neighboring points in the coarse grid, as shown in Figure 4. The interpolation operation is in fact sparse matrix vector products (SpMV) when local Lagrange interpolation is adopted, where the number of nonzero entries for each row of an interpolation matrix is no greater than 16. The antepolation matrix is the transpose of the interpolation matrix, therefore the number of nonzero entries for each column of an antepolation matrix is no greater than 16. In coincidence with the special characteristic of local Lagrange interpolation, the compressed sparse row (CSR) and the compressed sparse column (CSC) sparse matrix storage format is used for storing interpolation and antepolation matrices respectively. To reduce the amount of storage that is required, the symmetry of the plane waves is used on the BP levels,
with which significant saving in memory for storing lowest level aggregation/disaggregation matrix, the translation matrix, and the interpolation/interpolation matrix can be achieved.

Fig. 4 Local interpolation of a higher-level plane wave by lower-level waves

4.2 Iterative Solution Stage

After finishing the setup stage, the final matrix equation system is solved iteratively, with the aid of MLFMA to speed up MVM in each iteration. These coefficients are multiplied by the near field matrix to evaluate the near interaction. The near interaction evaluation can be done with the aid of CPEs using the idea of ‘one CPE per box’. The acceleration of near interaction is simple and straightforward. Since the SPM is too small, matrix entries associated with a near box pair and the corresponding coefficients are transferred to SPM via DMA. After finishing computation, the resulting data is transferred back to the main memory, and then the next computation begins. The double buffering scheme is also used to make computation and data transformation overlapped. The far-field interaction calculation is far more complicated. At the very beginning, the coefficients provided by the iterative solver are multiplied by the aggregation matrix to obtain the radiation patterns. Since the number of edges and planewaves in a box is not large, the idea of ‘one CPE per box’ is used for aggregation on the lowest level. Each lowest level box is assigned to a CPE, and contributions from each edge source in the box to different direction planewaves is calculated in sequence. The contribution to a same plane wave from all edges in a box should be summarized to obtain the result. Then, the radiated plane waves of the child boxes at the lower level are interpolated and central shifted to obtain the radiation plane waves of boxes on the higher level. This procedure continues until it reaches the second level of the MLFMA tree. Different parallelization strategies are adopted for CPEs at different MLFMA tree levels. On these levels where the total number of boxes partitioned to the process is far greater than 64, the simple partitioning strategy, in which the idea of ‘one CPE per box’ is used. However, on the rest high levels, for example the second
highest level, the number of nonempty boxes is comparable or even smaller than the number of CPEs. If we still use the simple partitioning strategy, serious load unbalance occurs. To overcome this problem, the hierarchical partitioning strategy, in which the idea of ‘one CPE group per box’ and ‘one CPE per planewave group’ is used. We first divide 64 CPEs into groups with the same number of CPEs, and then partitioning plane waves in a box is partitioned along the $\theta$-direction on CPEs in the same CPE group. Being aware of that the number of plane waves in each box decreases twofold in the $\theta$-direction from the current level to the next lower level, the number of CPE group should increase at the same speed.

![Diagram](image)

**Fig. 5** Illustration of hierarchical partitioning scheme for aggregation on high levels

The interpolation operation between plane waves on two successive levels is in fact a sparse matrix vector products (SpMV), in which the input and output vectors are plane waves of a child box and its parent, respectively. As discussed in the last section, when local Lagrange interpolation is adopted, nonzero entries of each row of the interpolation matrix which is stored in CSR format, is no greater than 16. Hence in each interpolation operation of a child box, we partitioning rows of the interpolation matrix into several subblocks. In each time, all nonzero entries of a subblock is transferred into SPM via DMA. On lower MLFMA levels, the box size is not large, plane waves in each child box can be cached in the SPM completely. However, on higher levels, the memory requirement for caching the whole plane waves of a box exceeds SPM size. If the input vector is stored in main memory, expensive memory access latency due to irregular accesses to the input vector brings significant computation deficiency. To overcome this bottleneck, a specially designed cache mechanism of hybrid dynamic and static buffers using the SPM is proposed. In the proposed buffer scheme, a large fixed size static buffer and a small dynamic buffer are used. In each computation, the static buffer first caches a fixed length of continuous input vector entries in it, with the start row number equals to the column number of the first nonzero submatrix entry. During calculation, if row number of a required input vector entry is judged to be not cached in the static buffer, it will be searched in the dynamic buffer. If it still not founded in the dynamic buffer, a small length of row continuous input vector entries, with the not founded one as the first, will be fetched from the main memory into the
dynamic buffer. Such procedure continues until the SpMV is finished. The result vector is stored continuously and finally transferred back to main memory and stored in the corresponding position.

The disaggregation is conducted opposite the traversal direction for aggregation. The parallelization strategy used for disaggregation is similar to the aggregation stage. The anterpolation operation in disaggregation is also a SpMV, with the input and output vectors are plane waves of a parent box and its child, respectively. For the same two boxes, the anterpolation matrix is the transpose of the interpolation. When stored in CSC format, the nonzero entries of each column are stored continuously, with maximum number of nonzero entries no greater than 16. Different from the interpolation computation, the input vector entries are continuously cached into SPM, while the output vector entries are stored using the hybrid static and dynamic buffer scheme, where the dynamic buffer is used for writing data back to the main memory. The use of hybrid dynamic and static buffers scheme improves data access efficiency significantly, which in turn improves the computation efficiency. During disaggregation, translation operations of plane waves belonging to the second near boxes on the same level are done at the same time. The parallelization strategy used for disaggregation is also the same as that for the aggregation and disaggregation stages. At the lowest level, the incoming plane waves are multiplied by the disaggregation matrix and received by the testing functions. The final calculation results are then summed with the near interaction results to complete the matrix-vector multiplication.

5 Numerical results and analysis

This section presents a series of numerical experiments to assess the accuracy, efficiency, and performance of SW-PMLFMA on the Sunway TaihuLight supercomputer. The computational frequency of each MPE and CPE of an SW26010 CPU is 1.45GHz, with PMLFMA being executed solely on MPEs, while SW-PMLFMA is executed with 64 CPEs to enhance the computation. The proposed approach is first validated by comparing the Radar Cross Sections (RCSs) of a conducting sphere with Mie series, where
a 30 m diameter sphere is illuminated by a 0.6 GHz plane wave and discretized into 786,597 triangular patches with 2,312,652 unknowns. In total, 28 CGs are employed for the computation. The results of PMLFMA, SW-PMLFMA, and Mie series are in good agreement, as illustrated in Figure 7.

Fig. 7 VV-polarized bistatic RCS of a sphere of diameter 30m, where 0 and 180 correspond to the back-scattering and forward-scattering directions, respectively

The computational statistics for the sphere with diameter 30m are presented in Table 1, in which ‘MPE’ and ‘CPE’ represent the PMLFMA and the SW-PMLFMA, respectively. \( V_s \) and \( V_f \) denote aggregation and disaggregation matrix assembly on the lowest MLFMA level, and \( Z_{near} \) denotes near-field matrix assembly. As we can see from this table, the speedup of near-field system matrix assembly part is over 14 times, which is significant because it is highly computational dense. However, the speedup achieved in the aggregation/disaggregation/translation process is not as pronounced, owing to the fact that the speedup of the far interaction is restricted not only by the data communications between the MPE and CPE but also by the MPI communications between MPEs. Nevertheless, the overall speedup achieved, which is almost 10 times, is a highly commendable result.

In order to demonstrate the efficiency of the parallel approach, scattering by a sphere is solved with varying numbers of CGs ranging from 4 to 40, using 1 MPI process and 64 CPEs per CG. The parallel efficiency is plotted against the number of CGs in Figure 8. The parallel efficiency is defined as 100% when 4 CGs are employed. It is worth noting that due to the limited number of levels in the MLFMA and the varying level sizes, the parallel efficiency may occasionally exceed 100%. As illustrated in the aforementioned figure, the parallel efficiency drops to 92.4% when 40 CGs are
Table 1  Speedup of bistatic rcs calculation of a conductor sphere of diameter 30m at 0.6GHz

<table>
<thead>
<tr>
<th>Time(s)</th>
<th>MPE 4CPE 64CPE</th>
<th>Speed up</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_s$ and $V_f$</td>
<td>2.02 11.93 1.03</td>
<td>1 0.17 1.96</td>
</tr>
<tr>
<td>$Z_{near}$</td>
<td>637.96 1267.55 44.51</td>
<td>1 0.50 14.33</td>
</tr>
<tr>
<td>Aggregation</td>
<td>34.38 151.86 5.20</td>
<td>1 0.23 6.61</td>
</tr>
<tr>
<td>Translation</td>
<td>77.07 696.15 25.13</td>
<td>1 0.11 3.07</td>
</tr>
<tr>
<td>Disaggregation</td>
<td>30.32 213.69 5.32</td>
<td>1 0.14 5.70</td>
</tr>
<tr>
<td>Total</td>
<td>781.75 2341.18 81.19</td>
<td>1 0.33 9.63</td>
</tr>
</tbody>
</table>

employed. The decrement in computing efficacy can be attributed to the escalation in the number of CGs, resulting in a reduction of the computing time and a concomitant rise in the time taken for MPI communication.

Fig. 8  Parallel efficiency for the solution of a sphere of diameter 30m involving 2,312,652 unknowns

To exemplify the efficiency and capability of the proposed method, a plane model as displayed in Figure 9 is employed. The aircraft has a length of 38 meters and the incident plane wave frequency is set at 1.34 GHz. The calculation utilized a final mesh comprising of 1,873,675 triangular patches, resulting in a total of 5,473,044 unknowns. In the process, 80 MPEs and 5,120 CPEs were utilized, and simulation details are presented in Table 2. The near-field system matrix assembly part demonstrated a speedup of approximately 10 times, whereas the overall speedup achieved was nearly 9 times, indicating an exceptional performance of the proposed method.

To demonstrate the computational proficiency of the SW-PMLFMA approach, the frequency of the incident plane wave has been raised to 15GHz. The aircraft model
surface has been discretized into 213,243,788 triangular patches, resulting in a total of 633,132,588 unknowns. The computation employs a combination of 1,600 MPEs and 102,400 CPEs, requiring 2.50TB of peak memory usage. After 63 iterations, a relative residual error of 0.005 is attained within a computation time of 103 min. The VV-polarized and VH-polarized bilateral RCSs of the aircraft have been visualized in Figure 10, while additional information regarding the simulations are tabulated in Table 3.

Table 3 Simulation details for an aircraft model

<table>
<thead>
<tr>
<th>Target</th>
<th>Aircraft model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximums dimension</td>
<td>1901.31A</td>
</tr>
<tr>
<td>Unknowns</td>
<td>633,132,588</td>
</tr>
<tr>
<td>MLFMA levels</td>
<td>8</td>
</tr>
<tr>
<td>Number of processes</td>
<td>1600</td>
</tr>
<tr>
<td>Number of MPEs</td>
<td>1600</td>
</tr>
<tr>
<td>Number of CPEs</td>
<td>102,400</td>
</tr>
<tr>
<td>Matrix filling time</td>
<td>25.96</td>
</tr>
<tr>
<td>Residual error</td>
<td>0.005</td>
</tr>
<tr>
<td>Iteration number</td>
<td>63</td>
</tr>
<tr>
<td>Iteration time</td>
<td>1h,5min</td>
</tr>
<tr>
<td>Total time</td>
<td>1h,43min</td>
</tr>
<tr>
<td>Peak memory on CPU</td>
<td>2.50TB</td>
</tr>
</tbody>
</table>

Table 2 Speedup of bistatic RCS calculation of an aircraft at 1.34GHZ

<table>
<thead>
<tr>
<th></th>
<th>Time(s)</th>
<th>Speed up</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MPE</td>
<td>1CPE</td>
</tr>
<tr>
<td>Vₚ and V₇</td>
<td>2.76</td>
<td>15.39</td>
</tr>
<tr>
<td>Zₙear</td>
<td>1065.02</td>
<td>2632.2</td>
</tr>
<tr>
<td>Aggregation</td>
<td>17.7</td>
<td>52.2</td>
</tr>
<tr>
<td>Translation</td>
<td>26.9</td>
<td>222.38</td>
</tr>
<tr>
<td>Disaggregation</td>
<td>17.00</td>
<td>95.7</td>
</tr>
<tr>
<td>Total</td>
<td>1129.38</td>
<td>3017.87</td>
</tr>
</tbody>
</table>
6 Conclusions

This study introduces a hybrid approach to parallelizing the MLFMA on SW26010 processors to solve extremely large 3D scattering problems. The approach utilizes a combination of MPI and Athread parallelization techniques, employing a flexible ternary partitioning scheme for MPI processes and a simple and hierarchical strategy for parallelization on the Compute Processing Elements (CPEs) at various MLFMA tree levels to achieve a balanced workload. The interpolation and anterpolation matrices are stored in the CSR and CSC sparse matrix formats, respectively. To enhance data access efficiency, a hybrid dynamic and static buffer scheme is proposed. Numerical results demonstrate the effectiveness of the algorithm, with a total speedup of the SW-PMLFMA achieved between 8.8 and 9.6 times compared to the PMLFMA on the MPEs. Furthermore, the study includes the results of a RCS analysis of an electrically large aircraft model, comprising over 600 million unknowns.

Acknowledgments. This work was supported by the NSFC (Grant Nos. 62231003 and Nos. 61971034).

Declarations

- Ethical Approval: not applicable.
- Competing interests: All authors disclosed no relevant relationships.
- Authors’ contributions: Conceptualization, Xin-Duo Liu, Wei-Jia He and Ming-Lin Yang; methodology, Xin-Duo Liu and Ming-Lin Yang; software, Xin-Duo Liu and Wei-Jia He; validation, Wei-Jia He, Ming-Lin Yang and Xin-Qing Sheng; formal
analysis, Wei-Jia He and Xin-Qing Sheng; investigation, Wei-Jia He and Xin-Qing Sheng; resources, Xin-Duo Liu and Xin-Qing Sheng; data curation, Xin-Duo Liu and Ming-Lin Yang; writing—original draft preparation, Xin-Duo Liu and Wei-Jia He; writing—review and editing, Xin-Duo Liu and Wei-Jia He. All authors have read and agreed to the published version of the manuscript.

- Funding: This work was supported by the NSFC (Grant Nos. 62231003 and Nos. 61971034).
- Availability of data and materials: Authors have the required data and supporting materials.

References


