Proceedings of ISAP 2014, Kaohsiung, Taiwan, Dec. 2-5, 2014

A Parallelized Multilevel Green's Function Interpolation Method for Open and Closed Objects

Peng Zhao[1], Da Qing Liu[1] and Chi Hou Chan[1]

[1] State Key Laboratory of Millimeter Waves, City University of Hong Kong, Hong Kong SAR, China

***Abstract* - A parallelized multilevel Green's function interpolation method (MLGFIM) based on extended combined field integral equation is developed for the analysis of electromagnetic problems involving composite open and closed conductors. The parallelized MLGFIM is developed using message passing interface technique on distributed-memory computer. Some issues for the achievement of an efficient parallelized MLGFIM are discussed. Numerical results demonstrate the accuracy and efficiency of this algorithm.**

***Index Terms* — Green's function interpolation, parallel computing, combined field integral equation.**

I. Introduction

As a kernel independent algorithm, MLGFIM has been developed to accelerate the matrix-vector multiplication [1]. This method inherits the multilevel tree structure of multilevel fast multipole algorithm and adopts the interpolation technique. The MLGFIM has been developed for the analysis of conducting objects based on electric field integral equation (EFIE). Unfortunately, EFIE encounters convergence and interior resonance problems for the solution of objects involving closed surfaces. Conventionally, combined field integral equation (CFIE) is applied to remedy this problem. To analyze composite open and closed surfaces, the extended CFIE [2] could be used. On the other hand, parallelized MLGFIM can enable the solution of very large problems. It has been realized with OpenMP technique on shared-memory computer [3]. However, OpenMP technique is not suitable for distributed-memory computer.

In this paper, a parallelized MLGFIM is developed based on the extended CFIE for the analysis of composite open and closed objects. Message passing interface (MPI) technique is applied to implement the parallelization on distributed-memory computer. Numerical results are given to demonstrate the accuracy and efficiency of the algorithm.

II. CFIE-Based MLGFIM

For 3-D electromagnetic problems, EFIE and magnetic field integral equation (MFIE) are written as

$$\begin{cases} \hat{t}\cdot\int_S dr'\left[\mathbf{J}(\mathbf{r}')+\dfrac{1}{k^2}\nabla'\cdot\mathbf{J}(\mathbf{r}')\nabla\right]G\left(\mathbf{r},\mathbf{r}'\right)=\dfrac{4\pi i}{k\eta}\hat{t}\cdot\mathbf{E}^{inc}(\mathbf{r}) & (1)\\ 2\pi\mathbf{J}(\mathbf{r})-\hat{n}\times\nabla\times\int_S dr'G\left(\mathbf{r},\mathbf{r}'\right)\mathbf{J}(\mathbf{r}')=4\pi\hat{n}\times\mathbf{H}^{inc}(\mathbf{r}) & (2)\end{cases}$$

where $\mathbf{J}(\mathbf{r}')$, k, η, $\mathbf{E}^{inc}(\mathbf{r})$ and $\mathbf{H}^{inc}(\mathbf{r})$ are the current density, wavenumber, impedance of medium, incident electric and magnetic fields, respectively. In (1) and (2), $G(\mathbf{r},\mathbf{r}') = \exp(ikR) / R$ and $R = |\mathbf{r} - \mathbf{r}'|$. The extended CFIE is a combination of EFIE and MFIE where the combination parameter $\alpha(\mathbf{r})$ changes with the field point position [2], viz.,

$$\text{CFIE}=\alpha(\mathbf{r})\text{EFIE}+(1-\alpha(\mathbf{r}))\,i/k\,\text{MFIE} \tag{3}$$

After applying the Galerkin method on (3), we obtain

$$\overline{\overline{A}}\,\overline{x}=\overline{V} \tag{4}$$

where entries of the matrix $\overline{\overline{A}}$ are expressed as

$$\begin{aligned} A_{i,j}&=\alpha(\mathbf{r})\int_{s_i}ds\int_{s_j}ds'[\mathbf{f}_i(\mathbf{r})\cdot\mathbf{f}_j(\mathbf{r}')-\nabla\cdot\mathbf{f}_i(\mathbf{r})\nabla'\cdot\mathbf{f}_j(\mathbf{r}')/k^2]G(\mathbf{r},\mathbf{r}')\\ &+(1-\alpha(\mathbf{r}))i/k\,\{2\pi\int_{s_i}ds\mathbf{f}_i(\mathbf{r})\cdot\mathbf{f}_j(\mathbf{r}')-\int_{s_i}ds\int_{s_j}ds'\mathbf{f}_i(\mathbf{r})\cdot\hat{n}_i\times[\nabla G(\mathbf{r},\mathbf{r}')\times\mathbf{f}_j(\mathbf{r}')]\}\\ &=A^E_{i,j}+A^M_{i,j}\end{aligned} \tag{5}$$

In (5), $\mathbf{f}_i(\mathbf{r})$ are the basis functions. We cannot accelerate the matrix-vector multiplication by MLGFIM unless (5) has the same form as the expression in [1]. By decomposing the entries into their scalar parts and with some manipulations, entries (5) associated with EFIE and MFIE in the far field calculation are rewritten as

$$\begin{cases} A^E_{i,j}=\sum\limits_{*=x,y,z}\int_{s_i}ds\int_{s_j}ds'\alpha(\mathbf{r})\mathbf{f}^*_i(\mathbf{r})G_E(\mathbf{r},\mathbf{r}')\mathbf{f}^*_j(\mathbf{r}') \\ \qquad -\dfrac{1}{k^2}\int_{s_i}ds\int_{s_j}ds'\alpha(\mathbf{r})\nabla\cdot\mathbf{f}_i(\mathbf{r})G_E(\mathbf{r},\mathbf{r}')\nabla'\cdot\mathbf{f}_j(\mathbf{r}') & (6)\\ A^M_{i,j}=\sum\limits_{*=x,y,z}\int_{s_i}ds\int_{s_j}ds'\left(1-\alpha(\mathbf{r})\right)\left[\mathbf{f}_i(\mathbf{r})\times\hat{n}_i\right]^*G_M(\mathbf{r},\mathbf{r}')\left[\mathbf{r}'_j\times\mathbf{f}_j(\mathbf{r}')\right]^* \\ \qquad +\sum\limits_{*=x,y,z}\int_{s_i}ds\int_{s_j}ds'\left(1-\alpha(\mathbf{r})\right)\left[\mathbf{r}\times(\mathbf{f}_i(\mathbf{r})\times\hat{n}_i)\right]^*G_M(\mathbf{r},\mathbf{r}')\,\mathbf{f}^*_j(\mathbf{r}') & (7)\end{cases}$$

where $G_E(\mathbf{r},\mathbf{r}') = G(\mathbf{r},\mathbf{r}')$ and $G_M(\mathbf{r},\mathbf{r}') = (ikR-1)G(\mathbf{r},\mathbf{r}') / R^2$. After that, (6) and (7) has the same form as

$$\hbar_{i,j}=\int_{s_i}ds\int_{s_j}ds'\tau_i(\mathbf{r})\Theta(\mathbf{r},\mathbf{r}')\varsigma_j(\mathbf{r}') \tag{8}$$

With the interpolation approach, (8) can be approximated as

$$\hbar_{i,j}=[\int_{s_i}ds\tau_i(\mathbf{r})\overline{W}^T_m(\mathbf{r})]\overline{\overline{\Theta}}_{mn}[\int_{s_j}ds'\varsigma_j(\mathbf{r}')\overline{W}_n(\mathbf{r}')]=\overline{v}^T_{m,i}\cdot\overline{\overline{\Theta}}_{mn}\cdot\overline{u}'_{n,j} \tag{9}$$

where $\overline{W}_m$ and $\overline{W}_n$ are the interpolation function vectors in the field box m and source box n, respectively.

III. Implementation of Parallelization

The development of parallelized MLGFIM on a distributed-memory computer is a nontrivial work. In the following, the suitability of this algorithm is investigated.

A. Partitioning

The whole task should be distributed into processors with minimal duplication. Since the MLGFIM is based on multilevel structure, it contains many boxes at the lower levels and it is appropriate to assign each of the boxes into a single processor (distributed levels). At the high levels, the number of boxes is very small, and hence it is appropriate to assign them into all processors (shared levels).

B. Load balancing and reduction of data communication

For achieving high efficiency, we must ensure the workload is balanced and minimize the data communication between processors as well. For the distributed levels, the load could be balanced by assigning approximately the same number of boxes into the processors. However, this may cause great data traffic. For example, the load shown in Fig. 1 is balanced. To calculation the interaction from the boxes *i* to *j* using the lower-to-upper level interpolation method [1], the data of box *i* in processor p_1 should be first sent to processor p_2. After the performance of translation at level 5, the result will be sent from processors p_3 to p_2. To reduce the data communication, each root box of the distributed levels and all its descendants should be assigned to the same processor. And we should make the load balanced as well.

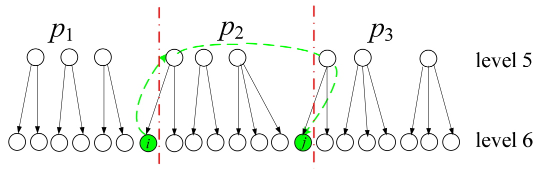

Fig. 1. Task distribution without the consideration of data communication.

This method can reduce the data communication in the far field calculation. However there may still have much data traffic in the near field calculation, because each box and its neighbors may not locate at the same processor. If so, the neighbors should be duplicated to this processor.

C. Solution

The parallelized MLGFIM can be developed by modifying the steps of the original MLGFIM given in [1]. First, the matrix-vector multiplication $\overline{S}_{n_L} = \overline{\overline{W}}_{n_L} \overline{x}_{n_L}$ is calculated at the finest level $l = L$ with its own sub-tasks. Second, the vectors $\overline{S}_{n_l} = \sum_{n_{l+1} \subset n_l} \overline{\overline{C}}_{n_l;n_{l+1}} \overline{S}_{n_{l+1}}$ for each box assigned to this processor are computed from level $l = L - 1$ to 2. Then the translation from the source group to the interaction list is computed. Data communication is required in this step, and we use MPI_Allreduce command to collect the vectors $\overline{S}_{n_l}$ from each processor, and distribute it to all processors. After that, the vectors $\overline{B}_{m_l} = \sum_{n_l \in \text{interaction list of } m_l} \overline{\overline{G}}_{m_l;n_l} \overline{S}_{n_l} + \overline{\overline{C}}^{T}_{m_l;m_{l-1}} \overline{B}_{m_{l-1}}$ in the corresponding boxes for the processor from level $l = 3$ to L are computed. Finally, the matrix-vector multiplication $\overline{b}_{m_L} = \overline{b}_{near} + \overline{\overline{W}}^{T}_{m_L} \overline{B}_{m_L}$ is performed and the vector $\overline{b}_{m_L}$ is also collected and distributed to all processors.

IV. Numerical Example

In this section, MLGFIM is applied to analyze the field distribution in a reverberation chamber with a horizontal stirrer, as shown in Fig. 2(a). The chamber and stirrer are modeled as closed and open surfaces, respectively. The structure is excited by a half wavelength dipole operating at 240MHz. The total number of unknowns is 101,939. We compare the magnitudes of electric field at the line of *a-b* calculated by the original MLGFIM and parallelized MLGFIM with eight processors, as shown in Fig. 2(b). Good agreement is observed from this figure. The CPU time for performing one matrix-vector multiplication, speed-up factor and efficiency factor are given in Table I. According to this table, we observe that the computational time is reduced significantly with the increase of the number of processors. The efficiencies of the parallelized MLGFIM are beyond 80% for all cases. The CPU time is reduced by 84.8% after calculating this problem with eight processors.

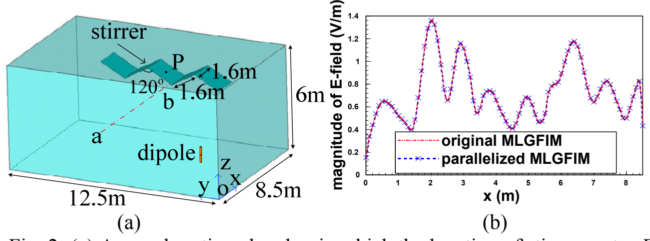

Fig. 2. (a) A reverberation chamber in which the location of stirrer center *P*, dipole center, points *a* and *b* are at (6.6, 6.25, 4.25), (2, 2, 1.6), (0, 6.2, 3) and (8.5, 6.2, 3), respectively. (b) The magnitude of E-field at line *a-b*.

TABLE I
Comparison of CPU Time and Memory Requirement

Number of processors	CPU time per matrix-vector multiplication	Speed-up	Efficiency (%)	Save time (%)
1	85.30 s	1.0	100	0
2	43.01 s	1.98	99.1	49.6
4	23.12 s	3.69	92.2	72.9
8	12.95 s	6.59	82.3	84.8

V. Conclusion

A parallelized MLGFIM has been developed to analyze metallic objects composed of open and closed surfaces. The reduction of data communication, the load balancing and the solution steps are discussed. Numerical example shows that the efficiency of the parallelized MLGFIM is beyond 80%.

References

[1] H. G. Wang and C. H. Chan, “The implementation of multilevel Green’s function interpolation method for full-wave electromagnetic problems,” *IEEE Trans. Antennas Propag.*, vol. 55, no. 5, pp. 1348-1358, May 2007

[2] Ö. Ergül and L. Gürel, “Iterative solutions of hybrid integral equations for coexisting open and closed surfaces," *IEEE Trans. Antennas Propag.*, vol. 57, no. 6, pp. 1751-1758, Jun. 2009.

[3] Y. Shi and C. H. Chan, “An OpenMP parallelized multilevel Green's function interpolation method accelerated by fast Fourier transform technique,” *IEEE Trans. Antennas Propag.*, vol. 60, no. 7, pp. 3305-3313, Jun. 2012.