OpenMP Parallelization of NURBS-HOMM for Electromagnetic Scattering Problems on Multi-Core Computer

Zi-Liang Liu and Chao-Fu Wang
Temasek Laboratories, National University of Singapore
#09-02, 5A Engineering Drive 1, Singapore 117411
E-mail: tslliuz@nus.edu.sg, cfwang@nus.edu.sg

Introduction

The moment method (MM) is a commonly used to simulate many electromagnetic (EM) problems by solving the integral equations [1]. In traditional MM, the surfaces of 3D objects are divided into many flat triangle or quadrilateral elements, the size of which is about $\lambda/10$ in each dimension. Then the low order basis functions, such as RWG and rooftop basis functions, are defined on these elements. When the object to be simulated is electrically large, a great number of elements and unknowns are required, and these lead to the cost of huge amount of memory and CPU time.

In recent years, the modeling technique of non-uniform rational B-splines (NURBS) is popular in computer aided geometric design (CAGD). It can accurately represent complex bodies in a manner of curve conforming with fewer curved patches, and no factitious geometric discontinuities are introduced due its ease to meet the tangential continuous conditions by adjusting the control points and weights. To reduce the computational cost for solving large problem, the authors have combined NURBS model with higher-order moment method (HOMM) and applied to the efficient analysis of EM scattering from large targets with the use of fewer unknowns to dramatically reduce the memory requirement [2]. However, NURBS-HOMM always takes longer CPU time to compute the impedance matrix, because each element is obtained by implementing a four-dimensional integral, and this kind of integrals are performed $N \times N$ times (N is the number of unknowns).

More recently, the quad-core processor has been widely used on PCs and severs. A new industry standard has been accordingly created with the aim to serve as a good basis for the development of parallel programs on multi-core machines: open multi-processing (OpenMP) [3]. It is an application programming interface (API) that supports multi-platform shared memory multiprocessing programming in C/C++ and Fortran on many architectures, including Unix and Microsoft Windows platforms. It consists of a set of compiler directives, library routines, and environment variables that influence run-time behavior. Compared with the message passing interface (MPI), OpenMP does not need a fair amount of restructuring code for parallelization. The existing code can be easily parallelized by placing OpenMP directives around time consuming loops which do not contain data dependences, leaving the source code unchanged. So OpenMP gives programmers a simple and flexible interface for developing parallel applications for platforms ranging from the desktop to the supercomputer.

In this paper, OpenMP is employed to speedup the EM scattering simulation with NURBS-HOMM, and the comparison of OpenMP and MPI is not the aim of this paper. The examples are executed on a PC of quad-core processor and the servers of multi-core processors, and the obtained results show that CPU time is much reduced and good parallel efficiency is achieved.

978-1-4244-4968-2/10/$25.00 ©2010 IEEE

Higher-Order Moment Method on NURBS Model

Let us consider a PEC surface S with unit normal $\hat{n}$ and denote $\vec{E}^i$ the impressed electric field on surface S. $\vec{J}_S$ is the surface current density induced by $\vec{E}^i$, and they satisfy the electric field integral equation (EFIE) on surface S

$$\hat{n}\times\vec{E}^i = \hat{n}\times[\,j\omega\mu\int_S \vec{J}_S(\vec{r}')G(\vec{r}',\vec{r})dS' - \frac{1}{j\omega\varepsilon}\nabla\int_S \nabla'_S\cdot\vec{J}_S(\vec{r}')G(\vec{r}',\vec{r})dS'], \tag{1}$$

To solve EFIE, surface S is divided into Q NURBS patches, on which the higher-order modified Legendre basis functions are defined. On each patch, the surface current density $\vec{J}_S$ is described in terms of the contravariant components as

$$\vec{J}_S(\vec{r}) = J_S^u(\vec{r})\vec{a}_u + J_S^v(\vec{r})\vec{a}_v\,, \tag{2}$$

where $\vec{a}_u = \partial\vec{r}/\partial u$ and $\vec{a}_v = \partial\vec{r}/\partial v$ are the covariant unitary vectors.

The two components $J_S^u(\vec{r})$ and $J_S^v(\vec{r})$ are expanded as

$$J_S^u(\vec{r}) = J_S^u(u,v) = \frac{1}{J_a(u,v)}\sum_{m=0}^{M}\sum_{n=0}^{N} b_{mn}^u f_{mn}^u(u,v), \tag{3a}$$

$$J_S^v(\vec{r}) = J_S^v(u,v) = \frac{1}{J_a(u,v)}\sum_{m=0}^{M}\sum_{n=0}^{N} b_{mn}^v f_{mn}^v(u,v), \tag{3b}$$

where b_{mn}^u and b_{mn}^v are the unknown coefficients, and $J_a(u,v) = |\,\vec{a}_u\times\vec{a}_v\,|$ is the surface Jacobian. $f_{mn}^u(u,v)$ and $f_{mn}^v(u,v)$ are the higher-order modified Legendre basis functions [2].

Applying the Galerkin testing procedure to equation (1) yields the matrix equation of NURBS-HOMM as follows

$$\begin{bmatrix}[V_{st}^u]_p\\ [V_{st}^v]_p\end{bmatrix} = \begin{bmatrix}[Z_{st,mn}^{uu}]_{pq} & [Z_{st,mn}^{uv}]_{pq}\\ [Z_{st,mn}^{vu}]_{pq} & [Z_{st,mn}^{vv}]_{pq}\end{bmatrix}\begin{bmatrix}[b_{mn}^u]_q\\ [b_{mn}^v]_q\end{bmatrix} \tag{6}$$

where

$$\begin{aligned} Z_{st,mn,pq}^{uu} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}[\; & j\omega\mu\, f_{st,p}^u(u,v)f_{mn,q}^u(u',v')\vec{a}_{u,p}\cdot\vec{a}_{u,q} \\ & + \frac{1}{j\omega\varepsilon}\frac{\partial f_{st,p}^u(u,v)}{\partial u}\frac{\partial f_{mn,q}^u(u',v')}{\partial u'}\;]G(\vec{r}_p,\vec{r}'_q)du'dv'dudv, \end{aligned} \tag{7}$$

$$\begin{aligned} Z_{st,mn,pq}^{uv} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}[\; & j\omega\mu\, f_{st,p}^u(u,v)f_{mn,q}^v(u',v')\vec{a}_{u,p}\cdot\vec{a}_{v,q} \\ & + \frac{1}{j\omega\varepsilon}\frac{\partial f_{st,p}^u(u,v)}{\partial u}\frac{\partial f_{mn,q}^v(u',v')}{\partial v'}\;]G(\vec{r}_p,\vec{r}'_q)du'dv'dudv, \end{aligned} \tag{8}$$

$$\begin{aligned} Z_{st,mn,pq}^{vu} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}[\; & j\omega\mu\, f_{st,p}^v(u,v)f_{mn,q}^u(u',v')\vec{a}_{v,p}\cdot\vec{a}_{u,q} \\ & + \frac{1}{j\omega\varepsilon}\frac{\partial f_{st,p}^v(u,v)}{\partial v}\frac{\partial f_{mn,q}^u(u',v')}{\partial u'}\;]G(\vec{r}_p,\vec{r}'_q)du'dv'dudv, \end{aligned} \tag{9}$$

$$Z^{vv}_{st,mn,pq} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\Big[\ j\omega\mu\, f^{v}_{st,p}(u,v) f^{v}_{mn,q}(u',v')\vec{a}_{v,p}\cdot\vec{a}_{v,q} + \frac{1}{j\omega\varepsilon}\frac{\partial f^{v}_{st,p}(u,v)}{\partial v}\frac{\partial f^{v}_{mn,q}(u',v')}{\partial v'}\ \Big] G(\vec{r}_p,\vec{r}'_q)\,du'dv'dudv. \tag{10}$$

Parallel Computation with OpenMP

When the impedance matrix $[Z]$ is computed, $[2\times(M+1)\times(N+1)\times Q]^2$ times of four dimensional integrals are required, which consume much CPU time. To reduce CPU time, OpenMP is used to compute matrixes $[Z]$ in parallel.

The speedup rate is defined by the following formula:

$$S_p = T_1 / T_p\,, \tag{11}$$

where p is the number of processors, T_1 is the execution time of the sequential code, and T_p is the execution time of the parallel code with p processors.

The parallel efficiency is a performance metrics defined as

$$E_p = S_p / p = T_1 /(pT_p)\,. \tag{12}$$

Numerical Results

The first example is a PEC sphere of radius 2 wavelength modeled by 128 NURBS patches. The orders of basis functions are M=3, N=2, 2256 unknowns are needed. This example is executed on 32-bit Windows XP PC with 2.83 GHz Intel quad-core processor and 3.0GB RAM. Fig. 1 shows the bistatic scattering pattern of the sphere, and Fig. 2 gives the CPU time of using 1 to 4 cores of the computer. The speedup rate and parallel efficiency are illustrated in Fig. 3.

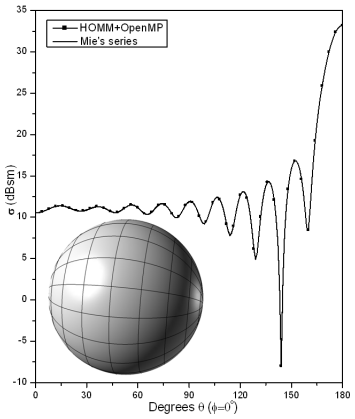

Fig. 1 Bistatic scattering pattern of PEC sphere

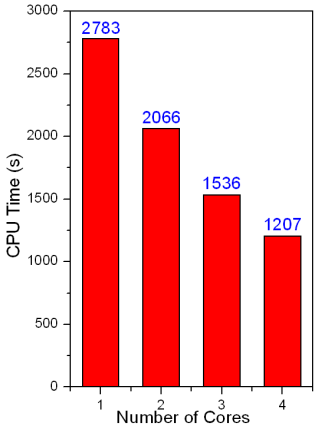

Fig. 2 CPU time for 1 to 4 cores

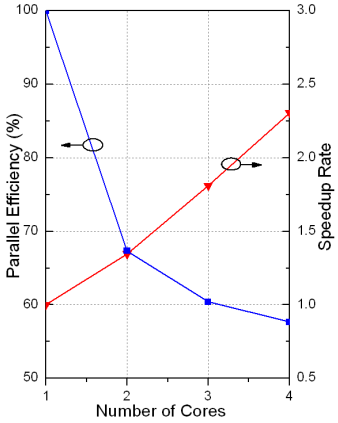

Fig. 3 Speedup rate and parallel efficiency

The second example is PEC aircraft-like geometry modeled with 176 NURBS patches. The length and wingspan of the geometry is about 12.86λ and 15.71λ, respectively. The surface area of the aircraft is about $184.53\lambda^2$. The orders of basis functions are M=3, N=2, 3000 unknowns are needed. A multi-core server is used to obtain the results, which equipped by Nehalem quad-core processors at 2.66GHz and

46-bit Linux OS. Fig. 4 and Fig. 5 display the bistatic scattering pattern of aircraft. CPU time is given in Fig. 6 when 1 to 8 cores are used. The speedup rate and parallel efficiency are plotted in Fig. 7.

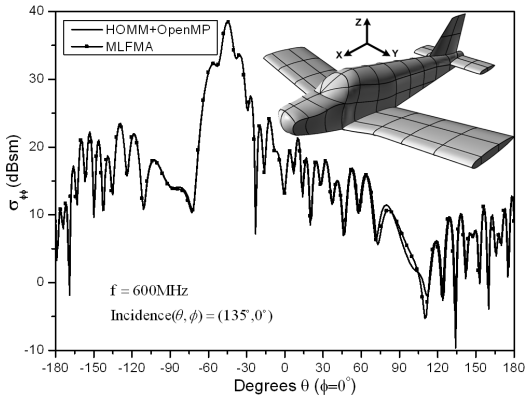

Fig. 4 Bistatic scattering pattern of aircraft ($\phi\phi$ polarization)

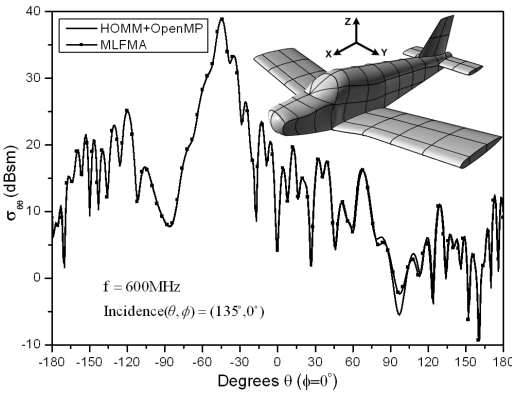

Fig. 5 Bistatic scattering pattern of aircraft ($\theta\theta$ polarization)

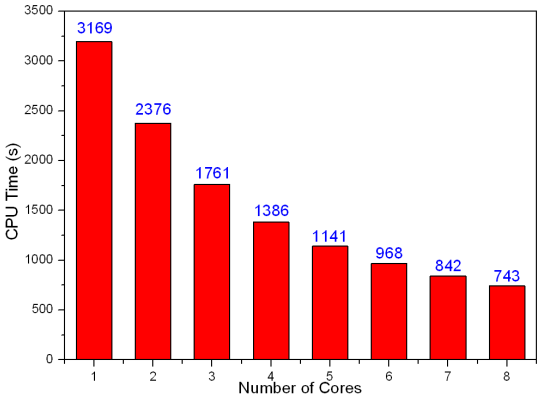

Fig. 6 CPU time for 1 to 8 cores

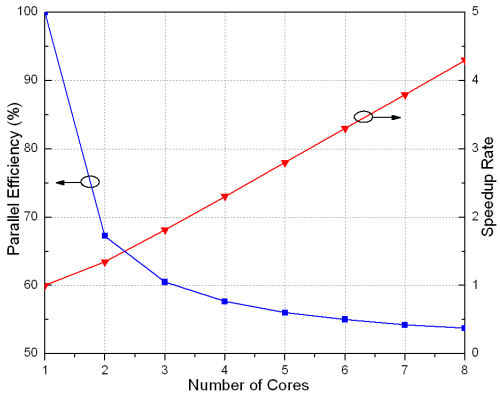

Fig. 7 Speedup rate and parallel efficiency

OpenMP is a convenient parallel technique to allow the code to be executed in parallel on a multi-core machine without mass restructuring code for parallelization. To reduce the computational time of NURBS-HOMM, OpenMP is employed to compute the impedance matrix in parallel. Numerical results show that CPU time is much reduced, and good speedup rate and parallel efficiency is achieved.

References

[1] S. M. Rao, D. R. Wilton, and A. W. Glisson, "Electromagnetic scattering by surfaces of arbitrary shape," IEEE Trans. Antennas Propagat., vol. 30, pp. 409-418, May 1982.

[2] Z.-L. Liu, and J. Yang, "Analysis of electromagnetic scattering with higher-order moment method and NURBS model", Progress In Electromagnetics Research (PIER), vol. 96, pp.83-100, 2009.

[3] M. Hermanns, "Parallel programming in Fortran 95 using OpenMP", University of Polytechnic of Madrid, Spain, Apr. 2002.