A Study of High-Speed Electromagnetic Computation by Finite-Difference Time-Domain Method

Naoki TAKADA*, Tomoyoshi ITO**, Kuniyuki MOTOJIMA***, Taichi SATO*
, and Shogo KOZAKI***
*Department of Electronic Control Engineering, Oyama National College of Technology
771 Nakakuki, Oyama 323-0806 Tochigi, Japan
E-mail ntakada@oyama-ct.ac.jp, tsato@oyama-ct.ac.jp
** Department of Electronics and Mechanical Engineering,
Faculty of Engineering, Chiba University
1-33, Yayoi-cho, Inage-ku, Chiba 263-8522, Chiba, Japan
E-mail itot@cute.te.chiba-u.ac.jp
***Department of Electrical Engineering Faculty of Engineering, Gunma University
1-5-1 Tenjin-cho, Kiryu 376-8515 Gunma, Japan
E-mail motojima@el.gunma-u.ac.jp, kozaki@el.gunma-u.ac.jp

1. Introduction

Finite-Difference Time-Domain (FDTD) Method[1] is a numerical technique that can be used to solve electromagnetic boundary value problems in the time domain. This method is excellent in computational efficiency, practicality, and simplicity. However, large scale simulations, such as simulations of three dimensional or large scale two dimensional electromagnetic simulations, require large computer memory space and a long computation time. We desire FDTD method to overcome these difficulties.

In the conventional FDTD method, we must store all electromagnetic field values in a computer memory space. As for the computation time, an algorithm or programming is very important because it depends on a usage of a cache memory in a CPU. By the cache block algorithm, we can effectively use a cache memory for a high-speed computation of FDTD method. We investigated the relation between a cache block size and a calculation speed. This is one of our topics.

As for the large computer memory space, a distributed FDTD method has been developed[2][3]. In this method, we use a FDTD algorithm on several computers connected to each other by a computer network. Since a necessary computer memory is provided by several computers, we can compute a large scale problem. This method also accelerates the calculation speed by the parallel computation. However, this method must need the communication between computers. Generally, the algorithm of a distributed FDTD method is very complex and it is very difficult for us to use this method. So we have developed new distributed FDTD method[3]. In our method, we divide a computational domain into the number of computers on a network. Each of the divided domains contains overlapping points. Therefore, we can make the program of our method easily. Furthermore, we realized the high cost performance of this computation system by using a low-cost personal computer (PC) network. The report of our method is another topic. Finally, we confirm the high-speed computation with our method.

2. The computation time of the conventional FDTD method

In FDTD method, Maxwell's equations are discretized directly without any analytic processes. We must store all electromagnetic field values in a computer memory space. In each time step, the calculation of FDTD method requires the surrounding electromagnetic field values. The memory access time remarkably influences the computation time of FDTD method. In recent years, a computer develops rapidly. Fortunately, a CPU has the cache memory that can be loaded at high speed. However, a program isn't always optimized by a compiler to use a cache memory effectively. Unfortunately, the computation time of FDTD method depends on a program.

By the cache block algorithm, we can use a cache memory in a CPU effectively. The conventional

program (C language) that isn't optimized by the cache block algorithm is as follows.

```
/*******    the calculation of magnetic field Hx *********/
 for(k=k_min;k<=k_max;k++) {
     for(i=i_min;i<=i_max;i++) {
        for(j=j_min;j<=j_max;j++) {
           hx[i][j][k]=hx[i][j][k]
              -dtdy*(ez[i][j][k]-ez[i][j-1][k])
              +dtdz*(ey[i][j][k]-ey[i][j][k-1]);
        }
     }
  }
```

Here, hx[i][j][k] is the matrix of the magnetic field Hx values, Δt is the time inclement, Δy and Δz are the size of the spatial divisions, $dtdy = \Delta t / \mu \Delta y$ and $dtdz = \Delta t / \mu \Delta z$.
The program (C language) optimized by the cache block algorithm is as follows.

```
 /*******    the calculation of magnetic field Hx by using cache block algorithm *********/
for(kk=k_min;kk<=k_max;kk=kk+loop_size) {
     for(ii=i_min;ii<=i_max;ii=ii+loop_size) {
        for(jj=j_min;jj<=j_max;jj=jj+loop_size) {
              for(k=kk;(k<=(kk+loop_size-1))&&(k<=k_max);k++) {
                    for(i=ii;(i<=(ii+loop_size-1))&&(i<=i_max);i++) {
                       for(j=jj;(j<=(jj+loop_size-1))&&(j<=j_max);j++) {
                         hx[i][j][k]=hx[i][j][k]
                                       -dtdy*(ez[i][j][k]-ez[i][j-1][k])
                                       +dtdz*(ey[i][j][k]-ey[i][j][k-1]);
                       }
                    }
               }
        }
     }
  }
```

In the cache block program, the loop_size influences the computation time of FDTD method. Next, we discuss the computation time of the conventional FDTD method by using a single PC. The PC is an IBM-PC compatible (CPU: Intel Pentium II 450Mhz, Memory: 256Mbyte, OS: Linux). We wrote the program in the C language (C compiler :gcc-2.7.2.3). Fig. 1 shows the computation time by the cache block program and the compiler options (-O3 -m486 -funroll-loops) against the number of loop sizes. In Fig.1, the size of a computational domain is $150\Delta x \times 150\Delta y \times 150\Delta z$ and the number of time steps is 100. When the loop size is 7, the computation time is the shortest by using a single PC. Fig.2 shows the computation time by using single PC against a computational domain. In Fig.2, T_n is the time when the program isn't optimized ,namely, we use the no compiler option and no cache block algorithm. T_L is the time when we only use the compiler options (-O3 -m486 -funroll-loops). T_B is the time when we use the compiler options (-O3 -m486 -funroll-loops) and the cache block program (loop_size=7). In Fig.2, the speed-up by the optimized program is about six times higher, compared with no optimized program. Therefore, it is reconfirmed that computation time depends on a program greatly.

3. Our distributed FDTD method[3]

We have developed the new lattice with the overlapping points. For computation in parallel, we improved the lattice of the conventional FDTD method as shown in Fig.3. The overlapping points are the same points of the electromagnetic field values in each lattice allotted to neighbor computers. In our method, we must communicate the electromagnetic field values on overlapping points between neighbor computers. By using our lattice, we can easily program distributed FDTD method because the electromagnetic field

values except for the absorbing boundary in each subdividing lattice can be calculated at once and we can program it visually. Furthermore, our method minimizes the communication cost. The accuracy of the absorbing boundary is important for the of FDTD method. We use the PML absorbing boundary[4] which is high accuracy in our method. Fig.4 shows the lattice of PML absorbing boundary. In Fig.4, we must communicate the electromagnetic field values between neighbor computers. In PML absorbing boundary, the electromagnetic field values for communication are the same in free space (Fig.3). Therefore, the algorithm of our method is very simple, as follows:

We repeat steps (1) to (8) in each time step.

(1) In free space, calculate the magnetic field values in parallel.
(2) Apply the magnetic boundary conditions.
(3) In PML, calculate the magnetic field values in parallel.
(4) In PML and free space, communicate the magnetic field values on the overlapping points between neighbor computers.
(5) In free space, calculate the electric field values in parallel.
(6) Apply the electric boundary conditions.
(7) In PML, calculate the electric field values in parallel.
(8) In PML and free space, communicate the electric field values on the overlapping points between neighbor computers.

Next, we discuss the effect of the high-speed computation by our method. For the purpose, we compared the computation time of our method with that of the conventional FDTD method by a single computer. We used System 1, which consisted of an IBM-PC compatible (CPU: Intel Pentium II 450MHz, Memory:256MByte) and an Ethernet network (100 Base-Tx). In this system, we used PC-UNIX (Linux) as the operating system and gcc-2.7.2.3 as the compiler of C language. In this computation, we used the cache block algorithm and the compiler options (-O3 -m486 -funroll-loops). The number of time steps is 100. Fig.5 shows the computation time against the size of the computational domain $L \times L \times L$. In Fig.5, T_1 is the computation time of the conventional FDTD method. T_n is the computation time of our method by using several PCs (n is the number of PCs). Fig.6 shows the estimated speed-up of our method against the conventional FDTD method. In Fig.6, $S_{pc=n}$ is the speed-up of our method by using several PCs (n is the number of PCs). In the computation of our method by using two PCs, the speed-up is about twice higher. However, the speed-up is about three times in the computation of our method by using four PCs. We consider that this difficulty is the reason why the communication program isn't optimized.

4. Conclusion

In this paper, we discussed the high-speed electromagnetic computation by FDTD method. By the cache block algorithm, the computation time of the conventional FDTD method reduces greatly. It shows that the memory access time remarkably influences the computation time of FDTD method. Furthermore, the high-speed computation is established by our method with the cache block algorithm.

Reference

[1] K. S. Yee, "Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media," IEEE Trans. Antennas and Propag., vol.AP-14, no.3, pp. 302-307,May 1966.

[2] D. P. Rodohan, et al. "A distributed implementation of the finite difference time domain (FDTD) method," Int. J. Numerical Modelling Electronic networks, Devices and Fields, vol.8, pp283-291, 1995.

[3] N.TAKADA, "New Distributed Implementation of the FDTD Method," Trans. IEICE, vol.J80-C-I No.2,pp.47-54, Feb. 1997 (in Japanese).
(N. Takada, K.Ando, K.Motojima, T.Ito, S.Kozaki, "New Distributed Implementation of the FDTD Method," Electronics and Communications in Japan, Part 2,vol.80, No.5,pp.8-16, 1997 (in English).)

[4] J. P. Berenger, "Three-Dimensional Perfectly Matched Layer for the Absorption of Electromagnetic Waves," Journal of Computational Physics, 127, pp. 363-379, 1996.

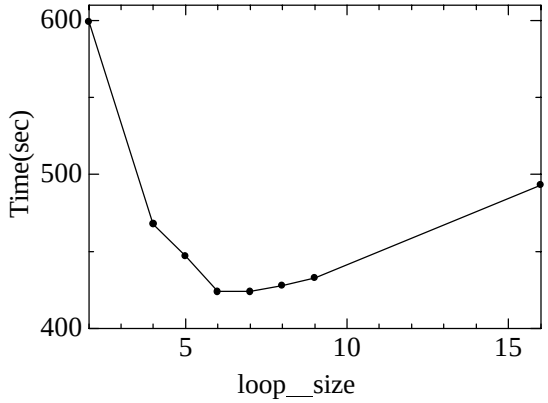

Fig. 1 The computation time
(the cache block algorithm)

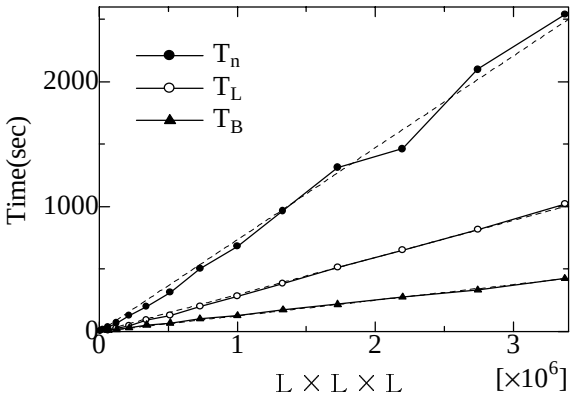

Fig.2 The computation time
(the conventional FDTD Method)

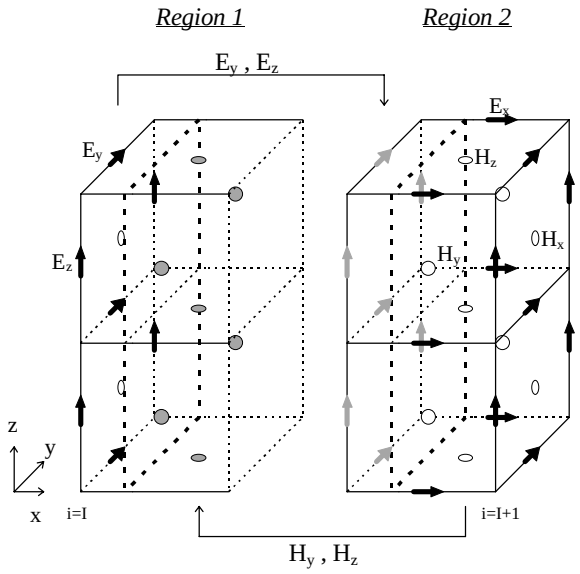

Fig.3 The lattice of our method (Free space)

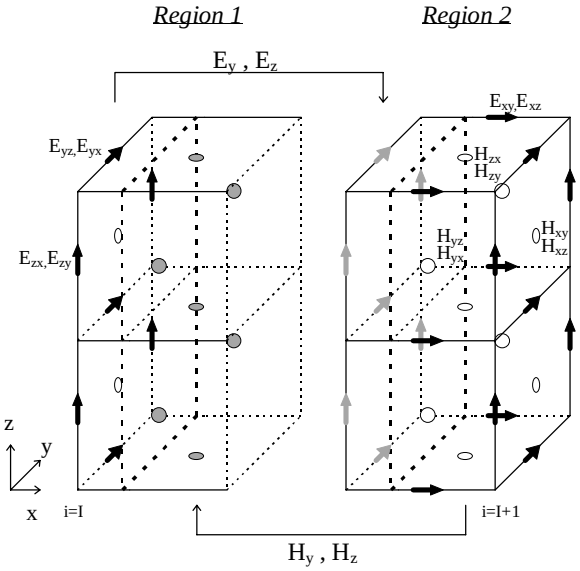

Fig.4 The lattice of our method (PML)

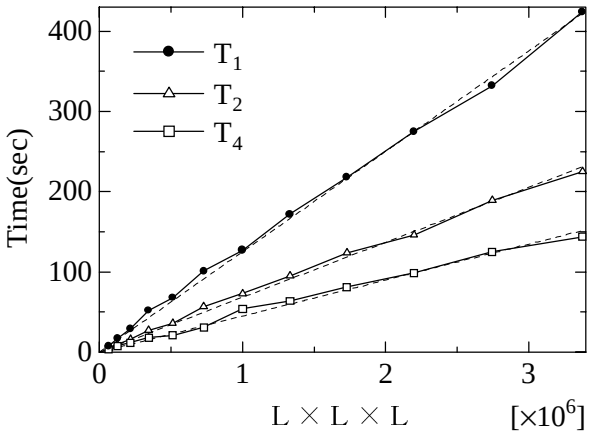

Fig.5 The computation time
(Our method)

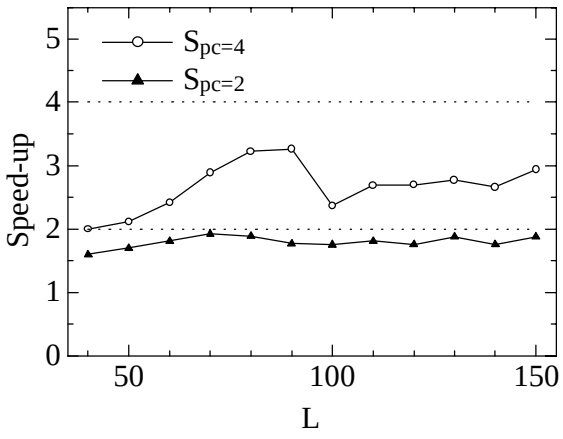

Fig.6 Speed-up
(Our method)