Parallel Implementation of the Scalar Wave-Equation FDTD Method Using the Message Passing Interface

Omar Ramadan
Eastern Mediterranean University, Gazi Magusa, Mersin 10, Turkey
omar.ramadan@emu.edu.tr

Oyku Akaydin
Eastern Mediterranean University, Gazi Magusa, Mersin 10, Turkey
oyku.akaydin@emu.edu.tr

ABSTRACT

In this paper, the parallel implementation of the Wave-Equation Finite Difference Time Domain (WE-FDTD) method, using the Message Passing Interface (MPI) system, is presented. The WE-FDTD computational domain is divided into subdomains using one-dimensional topology. Numerical simulations have been carried out for a line current source radiating in two-dimensional domains of different sizes and performed on a network of different number of PCs interconnected with Ethernet. It has been observed that, for large computational domains, the parallel implementation of the WE-FDTD method provides a significant reduction in the computation time, when compared with the parallel implementation of the conventional FDTD algorithm.

Key words: Parallel Processing, Message Passing Interface, Finite-Difference Time-Domain, Local Area Network, and wave equation.

1. Introduction

Nowadays, numerical methods play a major role in almost all branches of science and technology as they accelerate and facilitate research and industrial development. The Finite-Difference Time-Domain method (FDTD) [1] is one of the most widely used numerical time-domain techniques in electromagnetism, as it covers many applications [2], such as antennas, optics, high-speed electronic circuits, and semiconductors, etc. Furthermore, the FDTD method provides a wideband frequency response via a simple Fourier transform from the obtained time domain solutions. The primary advantage of the FDTD method is that it is a straightforward solution of the six-coupled field components of Maxwell's curl equations. This method, known as Yee algorithm [1], computes the field components by discretizing the Maxwell's curl equations both in time and space, and then solving the discretized equation in a time marching sequence by alternatively calculating the electric and magnetic fields in the computational domain [1].

Recently, the FDTD method has also been extended for solving the scalar Helmholtz wave equation in source-free domains [3]. Unlike the conventional FDTD approach, this new method, which is called the Wave Equation FDTD (WE-FDTD), allows computing any single field component without the necessity of computing other field components. Therefore, significant savings in the computational time and the memory storage can be achieved. In addition, it has been shown that the WE-FDTD method is both mathematically and numerically equivalent to Yee's algorithm [3] in source free regions.

A major drawback of both the FDTD and the WE-FDTD schemes is that very large computational time and very large computer memory storage are required for analyzing large computational domains. This makes parallelizing these schemes necessary. In order to do this, the computational domain is divided into

subdomains, and each subdomain is processed by one processor. Recently, different techniques have been introduced for the parallel implementation of the conventional FDTD method [4, 5]. These techniques are based on the single-program-multiple-data (SPMD) architecture. In [4], a one-dimensional parallelism using the parallel virtual machine (PVM) has been introduced. This approach is based on the TCP/IP protocol over the Ethernet for passing inter-processor messages. In [5], a new parallel FDTD algorithm based on the message-passing interface (MPI) system has been introduced. This approach is becoming new international standard for parallel programming and it is tending to replace the other parallel protocols, such as the PVM [6, 7].

In this paper, the MPI is used in the parallel implementation of the WE-FDTD algorithm. The two-dimensional computational domain is divided into subdomains along one direction by using the one-dimensional topology introduced in [4]. Numerical simulations have been carried out using a line current at the center and perpendicular to the domain. A line current source is a source which radiates equally in all directions. The test has been performed on a network of PCs interconnected with Ethernet.

The paper is organized as follows. In section 2, the formulations of both the FDTD and the WE-FDTD algorithms are presented. In section 3, the proposed parallelization techniques are described. Section 4 includes the results of several numerical tests which evaluate the effectiveness of the proposed method. Finally, a summary and conclusions are included in section 5.

2. Formulation

In a linear, homogeneous, isotropic medium, the Maxwell equations can be written as

$$\nabla \times \mathbf{H} = \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} \tag{1}$$

$$\nabla \times \mathbf{E} = -\mu_0 \frac{\partial \mathbf{H}}{\partial t} \tag{2}$$

where E and H are, respectively, the electric and the magnetic field vectors, ε_0 is the electric permittivity, and μ_0 is magnetic permeability of the medium. In the rectangular coordinate system, the above coupled curl equations can be decomposed into a system of six scalar differential equations in terms of the E_x, E_y, E_z, H_x, H_y, and H_z field components. For the sake of simplicity, consider the Maxwell's equations for the two dimensional transverse electromagnetic (TM) problem where only the field components E_z, H_x, and H_y exist. In this case, (1)-(2) give the following:

$$\frac{\partial E_z}{\partial t} = \frac{1}{\varepsilon_0}\left(\frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y}\right) \tag{3}$$

$$\frac{\partial H_x}{\partial t} = -\frac{1}{\mu_0}\frac{\partial E_z}{\partial y} \tag{4}$$

and

$$\frac{\partial H_y}{\partial t} = \frac{1}{\mu_0}\frac{\partial E_z}{\partial x} \tag{5}$$

By using the Yee's algorithm [1], the above equations can be discretized in space and time as

$$E_{zi,j}^{n+1} = E_{zi,j}^{n} + \frac{\Delta_t}{\varepsilon_0 \Delta}\left(H_{yi,j}^{n+1/2} - H_{yi-1,j}^{n+1/1} - H_{xi,j}^{n+1/2} + H_{xi,j-1}^{n+1/2}\right) \tag{6}$$

$$H_{xi,j}^{n+1/2} = H_{xi,j}^{n-1/2} - \frac{\Delta_t}{\mu_0 \Delta}\left(E_{zi,j+1}^{n+1} - E_{zi,j}^{n+1}\right) \tag{7}$$

$$H_{yi,j}^{n+1/2} = H_{yi,j}^{n-1/2} + \frac{\Delta_t}{\mu_0 \Delta}\left(E_{zi+1,j}^{n+1} - E_{zi,j}^{n+1}\right) \tag{8}$$

where Δ_t is the time step size and $\Delta = \Delta x = \Delta y$ is the space cell size in the x, and y directions, respectively.

To reduce the computational requirements of the conventional FDTD algorithm described above, (3)-(5) can be combined in a source free two-dimensional domain [3] as

$$\frac{1}{c^2}\frac{\partial^2 E_z}{\partial t^2}=\frac{\partial^2 E_z}{\partial x^2}+\frac{\partial^2 E_z}{\partial y^2} \qquad (9)$$

where c is the speed of light defined as $c=1/\sqrt{\varepsilon_0\mu_0}$. Equation (9) forms the basics of the WE-FDTD algorithm, which can be discretized as

$$E^{n+1}_{zi,j}=2E^{n}_{zi,j}-E^{n-1}_{zi,j}+\frac{c^2\Delta_t^2}{\Delta^2}\begin{pmatrix}E^{n}_{zi+1,j}+E^{n}_{zi-1,j}+E^{n}_{zi,j+1}\\+E^{n}_{zi,j-1}-4E^{n}_{zi,j}\end{pmatrix} \qquad (10)$$

To truncate open region problems, Absorbing Boundary Conditions (ABCs) are needed. In this paper, Mur's first order ABC is used [8]. As an example, the E_z field along the x=0 boundary can be computed as

$$E^{n+1}_{z0,j}=E^{n}_{z1,j}+\frac{c\Delta_t-\Delta}{c\Delta_t+\Delta}\left(E^{n+1}_{z1,j}-E^{n}_{z0,j}\right) \qquad (11)$$

3. Parallelizing the FDTD and the WE-FDTD Algorithms

In order to parallelize the above FDTD and the WE-FDTD algorithms, the computational domain is divided into subdomains. Using the one-dimensional topology introduced in [4], the computational domain is divided into subdomains along the x-direction, where each subdomain is assigned to one processor, as shown in Fig. 1. At the computational domain boundaries, i.e., X=0, X=Xn, Y=0 and Y=Yn boundaries, the fields are computed by using ABCs [8]. At the internal cells of the subdomains, the fields are computed directly from (6)-(8), in the case of the FDTD method, or from (10), in the case of the WE-FDTD method. However, to calculate the fields at the subdomain boundaries, data from the neighboring subdomains are needed. In this paper, the MPI system is used to exchange data between processors. A complete detail of the usage of the MPI is provided in [6, 7].

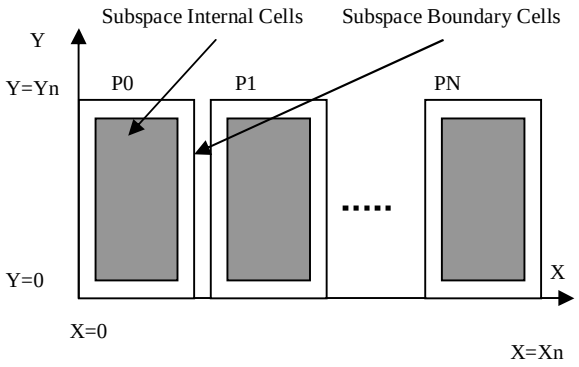

Fig. 1. Computational domain partitioning

Figure 2 shows the data to be exchanged between neighboring subdomains in order to parallelize the conventional FDTD method. As can be seen from Fig. 2, to calculate the E_z using (6) at the cells located at the plane on the left boundary of the subdomain, the values of the H_y field from the left subdomain are needed. Similarly, this subdomain must send the values of H_y at cells located at the plane on the right boundary of the subdomain to right subdomain. In the same manner, to calculate H_y using (8) at the cells located at the plane on the right boundary of the subdomain, the values of the E_z in the right subdomain are needed. Also, this subdomain should send the values of the E_z at the cells located at the plane on the left boundary of the subdomain to the left subdomain.

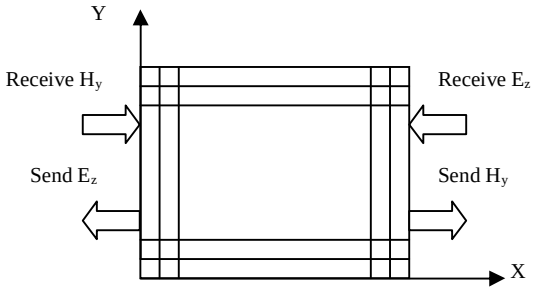

Fig. 2. Communications at the boundary of a subdomain for the one-dimensional topology of the conventional FDTD method

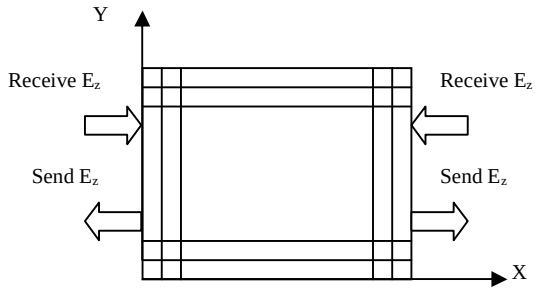

Fig. 3. Communications at the boundaries of a subdomain for the one-dimensional topology of the WE-FDTD method

Based on Fig. 2, the parallel implementation of the FDTD algorithm described in (6)-(8) can be summarized as:

1. MPI initialization.
2. Reading the simulation parameters.
3. Divide the computational domain into subdomains.
4. At each time step:
   4.1 Calculate the H_y field component.
   4.2 Communicate the H_y field component at the subdomain boundaries.
   4.3 Calculate the E_z field component.
   4.4 Communicate the E_z field component at the subdomain boundaries.
5. Apply the ABCs at the domain boundaries.
6. MPI finalization.

Similarly, Fig. 3 shows the data to be exchanged between neighbouring subdomains to implement the WE-FDTD algorithm described in (10). In order to calculate the E_z field component at the cells located at the left and the right planes of the subdomain, the values of E_z from the left and the right subdomains are needed. Similarly, this subdomain must send the values of the E_z at the left and the right planes to the left and the right subdomains. Therefore, the steps for the parallel implementation of the WE-FDTD algorithm described in (10) can be summarized as:

1. MPI initialization.
2. Reading the simulation parameters.
3. Divide the computational domain into subdomains.
4. At each time step:
   4.1 Calculate the E_z field component.
   4.2 Communicate the E_z field component at the subdomain boundaries.
   4.3 Apply the ABCs at the domain boundaries.
5. MPI finalization.

4. Numerical Study

To demonstrate the performance of the proposed parallel algorithms, numerical simulations were carried out for the Transverse Magnetic (TM) case [8]. In this test, a line current source is used to radiate electromagnetic waves at the center of a two-dimensional domain, as shown in Fig. 4 [8]. The space cell size in the x and y directions was chosen as $\Delta = \Delta x = \Delta y = 0.015m$ [8]. The time step was 25ps, and the simulation time was taken to be 500 time steps.

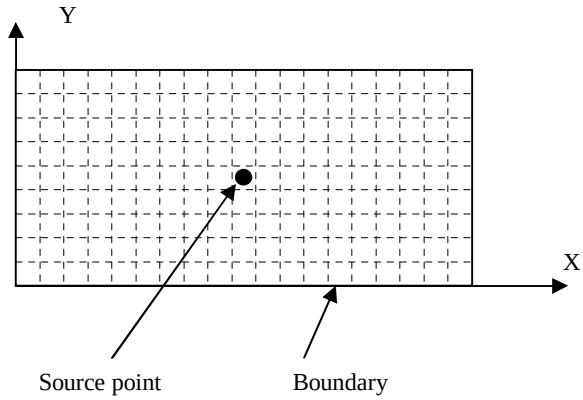

Fig. 4. Computational domain for the numerical study

The excitation used was a Gaussian pulse defined as

$$E_{z_{0,0}}^{n} = \begin{cases} \alpha(15\omega_1 \sin(\omega_1\xi) - 6\omega_2 \sin(\omega_2\xi) \\ +\omega_3 \sin(\omega_3\xi)); & \xi \le \tau \\ 0 & ;\xi \le \tau \end{cases} \quad (12)$$

where $\alpha = \frac{1}{320}$, $\tau = 10^{-9}$ s, $\xi = \text{nt}$, and $\omega_m = \frac{2\pi m}{\tau}, m = 1,2,3$. The above test was carried out for different number of processors and for different computational

domain size. The simulation was carried out on a network of Celeron™ 333 MHz PCs running with 64 MB of memory each. The PCs were interconnected with a 10 Mbit/s Ethernet. Tables 1 and 2 show the simulation time (in seconds) for the parallel FDTD and the parallel WE-FDTD methods obtained by using one, three, and five PCs for three different domain sizes.

Table 1. Simulation time (in seconds) for the parallel FDTD algorithm

Grid size (cells)	Number of PCs		
	1 PC	3 PCs	5 PCs
150x50	8.09	4.40	3.55
300x100	32.51	14.25	9.77
600x200	129.08	49.06	31.70

Table 2. Simulation time (in seconds) for the parallel WE-FDTD algorithm

Grid size (cells)	Number of PCs		
	1 PC	3 PCs	5 PCs
150x50	3.34	2.57	3.58
300x100	13.86	7.61	5.89
600x200	55.72	22.69	16.28

To measure the performance of the parallel algorithms, we computed the speedup, which is defined as

$$S(P) = T(1)/T(P) \qquad (13)$$

where T(1) is the time needed to solve the problem using one PC and T(P) is the time needed to solve the same problem using P PCs.

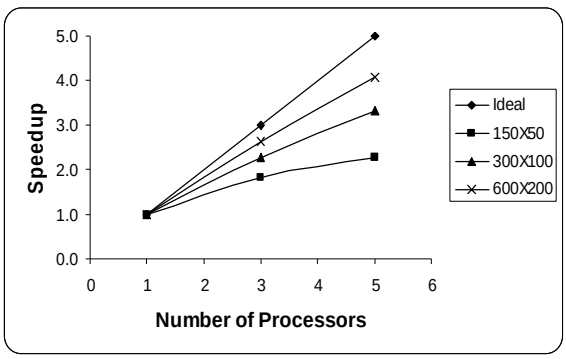

Fig. 5. Speed-up using the parallel FDTD algorithm

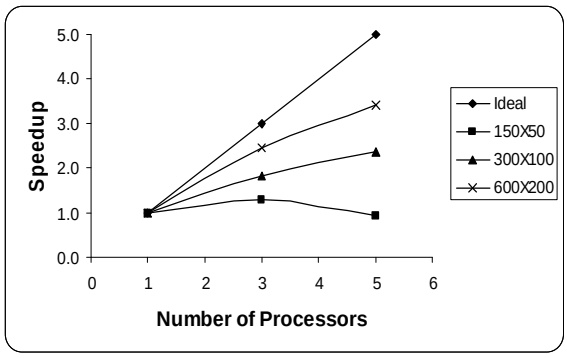

Fig. 6. Speed-up using the parallel WE-FDTD algorithm

Figures 5 and 6 show the speedup obtained with three and five PCs. For comparison purpose, the ideal speedup is also shown. From Figs. 5 and 6, it can be observed that as the computational domain size increases, the efficiency of the parallel FDTD and the WE-FDTD algorithms increases. On the other hand, when partitioning the computational domains over many processors, especially for the small domains, the efficiency of the parallelization will reach a limitation. This is because the computational time needed to update the fields will be reduced to point where it has the same order as the communication time needed to perform the data exchange between the processors. This explains the abnormal efficiency obtained with the parallel WE-FDTD algorithm for solving the 50x150 domain using five processors, as shown in Fig. 6. This abnormal phenomenon is not observed for the parallel FDTD algorithm. This is due to the fact that the computational time of the WE-FDTD method is much less than that of the FDTD algorithm as can be seen clearly from Tables 1 and 2.

5. Conclusion

In this paper, two parallel algorithms, based on the conventional FDTD and the new WE-FDTD algorithms have been implemented for solving the Maxwell's curl equations using the MPI system. The performance of these parallel algorithms has been studied by using a line current source radiating in a two-dimensional

domain. It has been observed that the parallel implementations of these two methods provide a significant reduction in the simulation time as compared with the sequential solution. On the other hand, when partitioning the computational domains over many processors, especially for the small domains, the efficiency of the parallelization will reach a limitation. This is because of the computation time required to update the fields will be reduced to a point where it has the same order as the communication time needed to exchange the fields between the processors. Finally, it has been observed that for large computational domains, the parallel WE-FDTD algorithm provides much more saving in the computational time compared with the parallel FDTD algorithm. The scheme can be generalized to two dimensional mesh of processors in the same manner. Finally, the new parallel algorithm can be improved by using better boundary conditions to truncate the computational domains, such as the Perfectly Matched Layer (PML) [9, 10].

References:

[1] K. S. Yee, "Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media,'' *IEEE Transaction on Antennas and Propagation*, Vol. 14, 1966, pp. 302-307.

[2] A. Taflove, *Computational Electrodynamics: The Finite-Difference Time-Domain Method*, Artech House, Boston, London, 1995.

[3] P. H. Aoyagi, J.-F. Lee and R. Mittra, "A hybrid Yee algorithm/scalar-wave quation approach,'' *IEEE Transaction Microwave Theory and Techniques*, Vol. 41, 1993, pp. 1593-1600.

[4] V. Varadarajan, and R. Mittra, "Finite-difference time domain (FDTD), analysis using distributed computing," *IEEE Microwave and Guided Wave Letters*, Vol. 4, No. 5, 1994, pp. 144-145.

[5] C. Guiffaut and K. Mahdjoubi, "A parallel FDTD algorithm using the MPI library," *IEEE Antennas and Propagation Magazine*, Vol. 43, No. 2, 2001, pp. 94-103.

[6] W. Gropp, E. Lusk, and A. Skjellum, *Using MPI: Potable parallel Programming with the Message-Passing Interface*, MIT Press, Cambridge, Mass., 1994.

[7] P. S. Pacheco, *Parallel Programming with MPI*, Morgan Kaufmann Publishers, San Francisco, Calif., 1997.

[8] P. A. Tirkas, C. A. Balanis and R. A. Renaut, "Higher order absorbing boundary conditions for the finite-difference time-domain method," *IEEE Transaction on Antennas and Propagation*, Vol 40, 1992, pp. 1215-1222.

[9] J. P. Berenger, "A perfectly matched layer for the absorption of electromagnetic waves," *Journal of Computational Physics*, Vol. 114, 1994, pp. 185-200.

[10] O. Ramadan, and A.Y. Oztoprak, "An efficient implementation of the PML for truncating FDTD domains," *Microwave and Optical Technology Letters*, Vol. 36, No. 1, 2003, pp. 55-60.