Sparse Finite Difference Time Domain Method

Christopher R. Doerr, *Fellow, IEEE*

***Abstract*— We propose a new electromagnetic simulation method called the sparse finite difference time domain (FDTD) method. It is based on standard FDTD but is faster by approximately an order of magnitude for large waveguide circuits, because it calculates only where significant electromagnetic energy is present. We derive and demonstrate 2-D sparse FDTD.**

***Index Terms*— Integrated optics, waveguide, silicon photonics, simulation.**

I. Introduction

FIGURE 1 shows a typical photonic integrated circuit (PIC) waveguide layout in silicon photonics, in this case a Mach-Zehnder delay interferometer (MZDI). Despite tremendous advances in both integrated optics and computing, today there is no commercial software that can simulate this structure directly from a mask polygon file, e.g. a Graphic Data System (GDS) file, in a reasonable amount of time. The main available packages are based on the beam propagation method (BPM) [1], the eigenmode expansion (EME) method [2], or the finite difference time domain (FDTD) method [3]. BPM cannot simulate this entire PIC because BPM requires paraxial transmission. EME and FDTD can theoretically simulate this PIC but would require an extraordinary amount of time and memory because of the large PIC size compared to an optical wavelength. EME may be faster than FDTD, but EME calculates for only one optical frequency at a time and thus must be run many times to calculate the response over a wide frequency range.

Today, to reasonably simulate this PIC, one must break it into pieces and simulate only parts of it, such as the couplers, idealize the rest of it, and calculate the response using scattering matrices. This method requires many assumptions, will not catch all layout errors, may incorrectly calculate path lengths, and can miss subtle effects, such as multimoding and unwanted coupling.

Our goal is to develop an electromagnetic simulation method that can directly simulate the response vs. wavelength of a large, high-index-contrast-waveguide PIC in a few orders of magnitude less time than existing commercial methods. We develop a 2D simulation method that is about an order of magnitude faster than conventional 2D simulation with nearly the same accuracy.

Manuscript received May 25, 2013; revised July 23, 2013; accepted October 2, 2013. Date of publication October 17, 2013; date of current version October 29, 2013.

The author is with Acacia Communications, Hazlet, NJ 07730 USA (e-mail: chris.doerr@acacia-inc.com).

Color versions of one or more of the figures in this letter are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LPT.2013.2285181

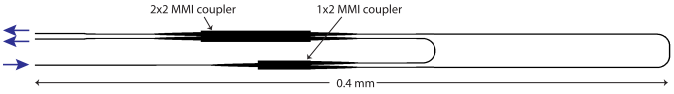

Fig. 1. Mach-Zehnder delay interferometer (MZDI) layout in Si photonics.

II. Sparse FDTD Derivation

Here we derive the equations for sparse 2D FDTD. The PIC is in the x-y plane, and we derive for TE polarization. We assume no variation in the z dimension and only isotropic, non-magnetic, non-dispersive materials. Maxwell's equations for this case are [4]

$$-\mu_0 \frac{\partial H_z}{\partial t} = \frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} \tag{1}$$

$$\varepsilon_0 \varepsilon_r \frac{\partial E_x}{\partial t} = \frac{\partial H_z}{\partial y} \tag{2}$$

$$\varepsilon_0 \varepsilon_r \frac{\partial E_y}{\partial t} = -\frac{\partial H_z}{\partial x} \tag{3}$$

where E and H are the electric and magnetic fields, respectively, and μ and ε are the susceptibility and permittivity, respectively.

Using the Yee lattice [3], i.e. center differences, we discretize in space and time ($x = p\Delta x, y = q\Delta y, t = s\Delta t$) to get

$$\begin{aligned} &H_z[p, q, s+1] \\ &\quad = H_z[p, q, s] - \frac{\Delta t}{\mu_0 \Delta x}(E_y[p+1, q, s] - E_y[p, q, s]) \\ &\qquad + \frac{\Delta t}{\mu_0 \Delta y}(E_x[p, q+1, s] - E_x[p, q, s]) \end{aligned} \tag{4}$$

$$\begin{aligned} &E_x[p, q, s+1] \\ &\quad = E_x[p, q, s] + \frac{\Delta t}{\varepsilon_0 \varepsilon_r((p+0.5)\Delta x, q\Delta y)\Delta y} \\ &\qquad \times (H_z[p, q, s+1] - H_z[p, q-1, s+1]) \end{aligned} \tag{5}$$

$$\begin{aligned} &E_y[p, q, s+1] \\ &\quad = E_y[p, q, s] - \frac{\Delta t}{\varepsilon_0 \varepsilon_r(p\Delta x, (q+0.5)\Delta y)\Delta x} \\ &\qquad \times (H_z[p, q, s+1] - H_z[p-1, q, s+1]) \end{aligned} \tag{6}$$

These are the equations used in conventional 2D FDTD. A rectangular 2D area that contains the structure of interest is defined, which has its spatial distribution of permittivity represented by ε_r. One starts with H_z, E_x, and E_y fields in the area at time $s = 1$. This initial distribution can be set as a pulse at the center wavelength of interest in a starting waveguide in the PIC. The pulse will propagate through the PIC, and the spectral content of the pulse at the end of the PIC gives the transmission over all wavelengths of interest in a single simulation run.

Then one calculates the new H_z field in the area using Eq. 4. Then one calculates the new E_x and E_y fields in the area using Eqs. 5 and 6. s is incremented by 1, and the process is repeated until the pulse (which may become multiple pulses) has finished propagating through the PIC.

This conventional FDTD algorithm is highly successful, but it is too slow and memory-consuming to simulate a very large PIC. This is because the E and H points in the entire area are calculated every step.

However, in typical PIC simulations most of the area contains negligible energy at a given time instant resulting in large wastes in memory and time. There is a speed-up technique used in radio wave simulations called moving window FDTD [5] that moves a small rectangular window along with a pulse propagating in the medium and the computation takes place only in the small window. However, such a method cannot easily handle cases where the pulse divides into multiple pulses, such as from couplers and/or reflections, or changes shape or direction; and moving window FDTD still has regions where the energy is negligible yet calculations are performed.

Here we propose a different concept called sparse FDTD. In sparse FDTD only the points that contain significant field energy are calculated. Similar to the concept of sparse matrices, a list of the field points that have electromagnetic energy greater than the maximum energy in the field divided by f, where f is a user-chosen constant $\gg 1$, is maintained, and only points in this list are used in the calculation. All other points are assumed to contain zero energy. Each list point contains the values of the three field components, E_x, E_y, and H_z and its p, q location.

In conventional FDTD, one cycles through all the p, q points for each step. For sparse FDTD we instead want to cycle through the list of active p, q points, and these points could be in any order. Let us start with Eq. 4 . Unfortunately we cannot efficiently use this equation as is because it has different E_x and different E_y points in the same equation, which makes it difficult to know when to create new active points. So instead we break the equation up into three parts in which all E_x and E_y in the same instruction are from the same p, q point and instead the H_z are from different points. The first part contains only H_z at the same p, q point and is easy to execute by cycling through the active-point list:

$$H_z[p,q] \Leftarrow H_z[p,q] + \frac{\Delta t}{\mu_0 \Delta x} E_y[p,q] - \frac{\Delta t}{\mu_0 \Delta y} E_x[p,q] \quad (7)$$

We dropped the s and instead wrote it as a computer instruction which calculates the new H_z using the previous H_z. The other two parts are

$$H_z[p-1,q] \Leftarrow H_z[p-1,q] - \frac{\Delta t}{\mu_0 \Delta x} E_y[p,q] \quad (8)$$

$$H_z[p,q-1] \Leftarrow H_z[p,q-1] + \frac{\Delta t}{\mu_0 \Delta y} E_x[p,q] \quad (9)$$

These three instructions are equivalent to Eq. 4. For instructions 8 and 9 we again cycle through the list of active points, but now for each point p, q we have to find the locations in the list of the points $p-1, q$ and $p, q-1$ to update those H_z and to know whether or not to create new points. One option is to do a linear search of the list. However, this would take too much time. Another option is to constantly sort the list in terms of p, q and do a geometric search, but again this would take too much time. The option we chose is to keep in memory a $N_x \times N_y$ array of the list indices, where N_x and N_y are the dimensions of the computational window. In other words, the point p, q in the array contains the location in the list of the active point from that spatial location. If the value at point p, q is zero then the point does not exist in the list. Thus, for example, if we need to find where in the list the point $p-1, q$ is, we simply look up the value in the 2D array with index $p-1, q$ and that tells us the index in the list. The amount of memory consumed for this is not great because the values in the array are only 32-bit integers.

In executing instruction 8, if it is found that the point $p-1, q$ is not in the active-point list, then we add it to the list. To add it to the list, we look at the last element in the list of holes in the active-point list (discussed below) and put it in that hole. If the list of holes is empty, then we lengthen the list of active points and add it to the end. Instruction 9 with point $p, q-1$ is handled similarly.

Eq. 5 is handled similarly with

$$E_x[p,q] \Leftarrow E_x[p,q] + \frac{\Delta t}{\varepsilon_0 \varepsilon_r((p+0.5)\Delta x, q\Delta y)\Delta y} H_z[p,q] \quad (10)$$

$$E[p,q+1] \Leftarrow E_x[p,q+1] + - \frac{\Delta t}{\varepsilon_0 \varepsilon_r((p+0.5)\Delta x, (q+1)\Delta y)\Delta y} H_z[p,q] \quad (11)$$

and likewise for Eq. 6.

$$E_y[p,q] \Leftarrow E_y[p,q] - \frac{\Delta t}{\varepsilon_0 \varepsilon_r(p\Delta x, (q+0.5)\Delta y)\Delta y} H_z[p,q] \quad (12)$$

$$E_y[p+1,q] \Leftarrow E_y[p+1,q] + + \frac{\Delta t}{\varepsilon_0 \varepsilon_r((p+1)\Delta x, (q+0.5)\Delta y)\Delta x} H_z[p,q] \quad (13)$$

As the simulation proceeds, many new points are added to the active-point list, because the optical field is propagating. This would make the list grow longer and longer. Fortunately as the pulse propagates it naturally clears itself away in its wake. We need to remove these points so that the active point list length stays approximately the same. To do this, after every cycle we calculate the energy, $\varepsilon_0 \varepsilon_r (E_x^2 + E_y^2) + \mu_0 H_z^2$, of each of the active points. If a point has an energy below the maximum energy in the list divided by f, it is removed from the list. To mitigate shock waves from abrupt changes in energy, points with an energy 10 times above this are attenuated. When a point is removed the list, it leaves a hole in the active-point list. We keep a separate list of the holes which contains the indices of the holes in the active-point list.

Finally, one must deal with the window boundaries. In conventional FDTD, the most successful boundaries are perfectly matched layers [6], which produce no reflections. While these can probably be implemented in sparse FDTD, for now we employ the simpler absorbing boundary conditions (ABCs) [7], which completely eliminate reflections only

perfectly normal to the boundary. Here we show a new ABC that is simple to implement.

The concept behind ABCs is the following: the fields on the computation boundaries are set such that it is an outward traveling wave with the speed of propagation in the material at that point with the amplitude that matches the field adjacent to the boundary. A plane wave traveling in direction $\hat{n}$ in material with ε_r obeys

$$\mathbf{E} = \hat{n} \times \mathbf{H} \sqrt{\frac{\mu_0}{\varepsilon_0 \varepsilon_r}} \tag{14}$$

Thus the ABCs at the four boundaries are

$p = 1$ boundary

$$E_y[1, q] = H_z[1, q] \sqrt{\frac{\mu_0}{\varepsilon_0 \varepsilon_r}} \tag{15}$$

$q = 1$ boundary

$$E_x[p, 1] = H_z[p, 1] \sqrt{\frac{\mu_0}{\varepsilon_0 \varepsilon_r}} \tag{16}$$

$p = N_x$ boundary

$$H_z[N_x, q] = E_y[N_x, q] \sqrt{\frac{\varepsilon_0 \varepsilon_r}{\mu_0}} \tag{17}$$

$q = N_y$ boundary

$$H_z[p, N_y] = -E_x[p, N_y] \sqrt{\frac{\varepsilon_0 \varepsilon_r}{\mu_0}} \tag{18}$$

Eqs. 17 and 18 are executed by going through the list and seeing if any points are on a boundary after executing instructions 7 - 9 and likewise with Eqs. 15 and 16 after 10 - 13.

III. Simulation Operation

At the start of a 2D sparse FDTD simulation, a TE-polarized pulse is launched into an input waveguide of the PIC, centered at the optical frequency of interest. Using a pulse ensures that the area occupied by field is small and also allows one to measure the response of the structure at all frequencies at once. Because the waveguide is dispersive, one cannot make the pulse too short or else the pulse will quickly occupy a large area, slowing down the simulation, and the pulse cannot be too long or else again it occupies a large area. 15 fs is generally an optimum pulse width for high-index-contrast PICs.

Light that scatters into the cladding can quickly increase the list size, greatly slowing the simulation. To mitigate this, we fill regions in between waveguides (leaving a margin around waveguides) with a material that is slightly absorbing. To make the material absorbing we multiply the field point by 0.99 each iteration when the permittivity is a certain value.

IV. Simulation Example

We implemented the code in a C subroutine that is called from a graphical user interface in MATLAB. Here we simulate the MZDI of Fig. 1. It is in Si wire waveguides with a core thickness of 220 nm. We use the effective index for the core, since it is a 2D simulation. Our code takes a GDS file input. The ε_r 2D array is filled in using the polygons in the GDS file. The refractive index is averaged over 8×8 regions on a grid that is 1/8 that of the simulation grid.

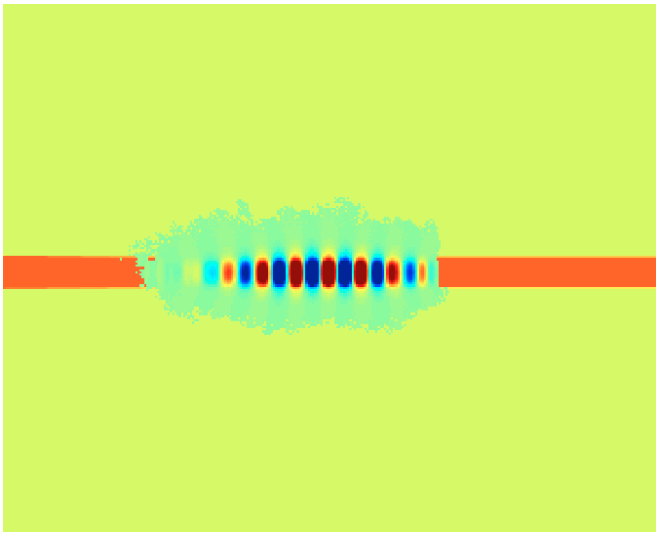

Fig. 2. Simulated pulse in a Si wire waveguide using sparse FDTD. The cloud around the mode shows the points in the list. The pulse is moving from left to right.

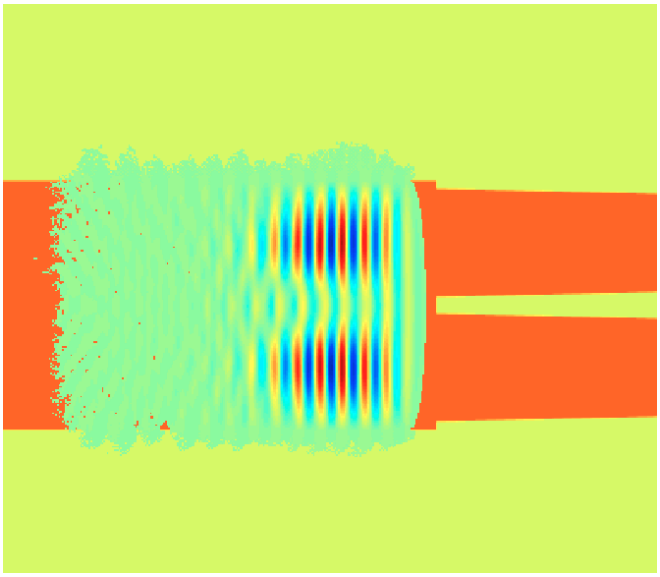

Fig. 3. Simulated pulse in the 1×2 MMI coupler of the MZDI using sparse FDTD. Again, the cloud shows the points in the list.

Figure 2 shows a screen shot from the sparse FDTD simulation of the pulse in the input waveguide. The pulse center wavelength is 1548 nm. The plotting routine zooms in and centers on the point with the most energy, and thus the plot window follows along the waveguide during execution. One can see a cloud around the mode, showing the extent of the active points in the list. The grid size is 40 nm and f is 100,000.

Fig. 3 shows a screen shot of the pulse in the 1×2 MMI coupler, and Fig. 4 shows a zoomed-out screen shot showing the two pulses (having split from the input pulse in the 1×2 MMI coupler) in the two arms of the MZDI.

The simulation took 6.0 minutes to complete on a MacBook Pro computer. The transmissivity as recorded by the pulse is

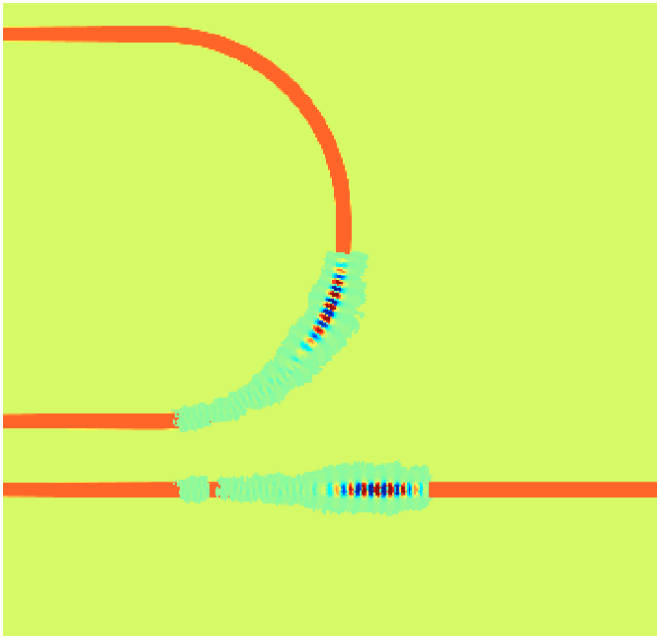

Fig. 4. Zoomed-out screen shot of the simulated pulse in the MZDI using sparse FDTD. One can see that it has split into two pulses, one in each arm of the MZDI.

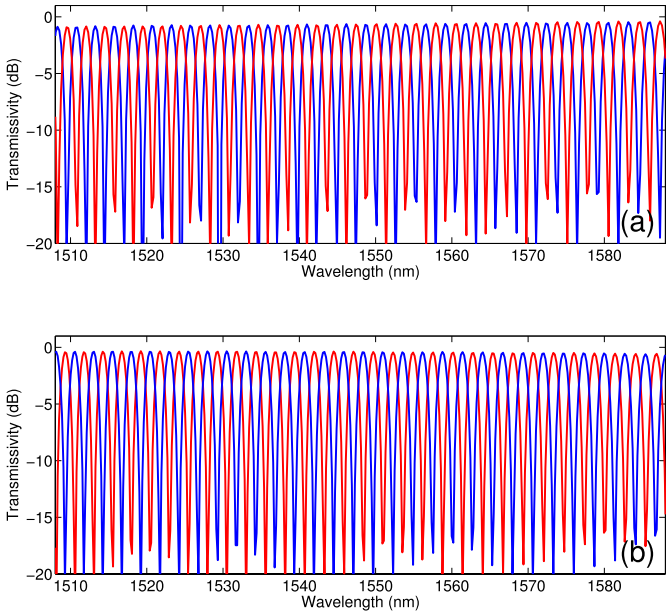

Fig. 5. Calculated responses of the MZDI from sparse FDTD (a) and a commercial FDTD simulator (Lumerical) (b). The red and blue curves are from the upper and lower output ports respectively.

shown in Fig. 5(a). The transmission is calculated by recording E_y and H_z at a slice on the input waveguide and at two slices on the two output waveguides, multiplying the Fourier transform of E_y by the complex conjugate of the Fourier transform of H_z, taking the real part, and then dividing the output by the input to get the transmission.

For comparison, the same structure was simulated in 2D in a commercial FDTD simulator, Lumerical, using the same grid size and computer. The simulation took 49.6 minutes. The results are shown in Fig. 5(b). The results are nearly identical, except Lumerical shows a 0.50-dB lower loss at short wavelengths and a slight wavelength shift in the peaks. The slightly higher loss for sparse FDTD may be due to sidewall scattering in the bends due to the finite grid size, and Lumerical has a more sophisticated method of handling dielectric constant averaging.

The measured free-spectral range for both cases is ∼30% in error, because the 2D simulation ignores the group velocity dispersion caused by the vertical confinement. This can be corrected by either changing to a 3D simulation or modifying the simulator to handle dispersive materials.

The sparse FDTD simulation is 8.3 times faster than the Lumerical simulation. Comparing the number of instructions between conventional and sparse FDTD, the speed-up should have been significantly greater. We believe that better optimization of the sparse FDTD code and parallelization for multiple cores, will result in more disparity. With larger PICs, the disparity is significantly higher.

Sparse FDTD was tested on many PIC structures. It works best on structures where the fields are well confined. When simulating structures where the field spreads out significantly, such as the star couplers in arrayed-waveguide gratings, the active-point list can become very long, slowing the simulation considerably.

In conclusion, a new way of calculating electromagnetic field propagation using FDTD in 2D was proposed and demonstrated. It calculates the response for all optical frequencies of interest simultaneously directly from the GDS file. It is about one order of magnitude faster and uses significantly less memory than conventional FDTD when simulating large PICs with low amounts of scattering.

REFERENCES

[1] Y. Chung and N. Dagli, "An assessment of finite difference beam propagation," *IEEE J. Quantum Electron.*, vol. 26, no. 8, pp. 1335–1339, Aug. 1990.

[2] P. Bienstman and R. Baets, "Optical modelling of photonic crystals and VCSELs using eigenmode expansion and perfectly matched layers," *Opt. Quantum Electron.*, vol. 33, nos. 4–5, pp. 327–341, 2001.

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

[4] K. Kawano and T. Kitoh, *Introduction to Optical Waveguide Analysis*. Hoboken, NJ, USA: Wiley, 2001.

[5] J. Schuster, K. Wu, R. Ohs, and R. Luebbers, "Application of moving window FDTD to predicting path loss over forest covered irregular terrain," in *Proc. IEEE Antennas Propag. Soc. Int. Symp.*, vol. 2. Jun. 2004, pp. 1607–1610.

[6] J.-P. Berenger, "A perfectly matched layer for the absorption of electromagnetic waves," *J. Comput. Phys.*, vol. 114, no. 2, pp. 185–200, 1994.

[7] G. Mur, "Absorbing boundary conditions for the finite-difference approximation of the time-domain lectromagnetic-field equations," *IEEE Trans. Electromagn. Compat.*, vol. EMC-23, no. 4, pp. 377–382, Nov. 1981.