GPU accelerated time domain DGA method for wave propagation problems on tetrahedral grids

Matteo Cicuttin[1], Lorenzo Codecasa[2], Bernard Kapidani[3], Ruben Specogna[3], Francesco Trevisan[3]
[1]Université Paris-Est, Cermics (ENPC), F-77455 Marne-la-Vallée, France
[2]Politecnico di Milano, Dip. di Elettronica, Informazione e Bioingegneria, I-20133, Milano, Italy
[3]Polytechnic Department of Engineering and Architecture (DPIA), Università di Udine, I-33100 Udine, Italy
kapidani.bernard@spes.uniud.it

The classical Finite Difference Time Domain method due to Yee is known to be massively parallelizable and efficently deployable on Graphical Processing Units (GPUs). Classical FDTD however requires a Cartesian discretization of the computational domain, which limits the abilities to model arbitrary geometries. The recently introduced explicit, consistent and conditionally stable DGATD method, which employs tetrahedral grids, enables full modeling flexibility while maintaining great computational efficiency. In this paper we present an evaluation of the DGATD method implemented on GPUs using NVidia CUDA libraries.

Index Terms—**Time Domain methods, wave propagation, Parallel computing, Tetrahedral meshes**

I. Introduction

The classical Finite Difference Time Domain (FDTD) method, devised by Yee [1] on two interlocked Cartesian orthogonal grids, has been extensively studied and used for its simplicity and computational efficiency. This method is used to numerically solve Maxwell's equations in a bounded region $\Omega \subset \mathbb{R}^3$. We recall that the equations for the continuous case are

$$\frac{\partial \boldsymbol{d}}{\partial t} = \nabla \times \boldsymbol{h}, \tag{1a}$$
$$\boldsymbol{e} = \varepsilon^{-1}\boldsymbol{d}, \tag{1b}$$
$$\frac{\partial \boldsymbol{b}}{\partial t} = -\nabla \times \boldsymbol{e}, \tag{1c}$$
$$\boldsymbol{h} = \mu^{-1}\boldsymbol{b}. \tag{1d}$$

The generic FDTD algorithm has been reinterpreted in the FIT framework [2], which is also based on a discretization employing two interlocked grids, a primal grid $\mathcal{G}$ and a dual grid $\tilde{\mathcal{G}}$ obtained by barycentric subdivision of $\mathcal{G}$. This yields the discrete system of equations

$$\frac{\tilde{\mathbf{\Psi}}^n - \tilde{\mathbf{\Psi}}^{n-1}}{\Delta t} = \tilde{\mathbf{C}}\tilde{\mathbf{F}}^{n-\frac{1}{2}}, \tag{2a}$$
$$\mathbf{U}^n = \mathbf{M}_{\varepsilon^{-1}}\tilde{\mathbf{\Psi}}^n, \tag{2b}$$
$$\frac{\mathbf{\Phi}^{n+\frac{1}{2}} - \mathbf{\Phi}^{n-\frac{1}{2}}}{\Delta t} = -\mathbf{C}\mathbf{U}^n, \tag{2c}$$
$$\tilde{\mathbf{F}}^{n+\frac{1}{2}} = \mathbf{M}_{\mu^{-1}}\mathbf{\Phi}^{n+\frac{1}{2}}, \tag{2d}$$

in which $\tilde{\mathbf{\Psi}}^n$, $\tilde{\mathbf{\Psi}}^{n-1}$ are the vectors of the fluxes of the electric displacement across the faces of $\tilde{\mathcal{G}}$ at time instant $n\Delta t$ and $(n-1)\Delta t$ respectively, $\mathbf{U}^n$ is the vector of the line integrals of the electric field along the edges of $\mathcal{G}$ at time instant $n\Delta t$; $\mathbf{\Phi}^{n+\frac{1}{2}}$, $\mathbf{\Phi}^{n-\frac{1}{2}}$ are the vectors of the fluxes of the magnetic induction across the faces of at time instants $(n+\frac{1}{2})\Delta t$ and $(n-\frac{1}{2})\Delta t$ and $\tilde{\mathbf{F}}^{n+\frac{1}{2}}$ is the vector of the line integrals of the magnetic field along the edges of $\tilde{\mathcal{G}}$ at time instant $(n+\frac{1}{2})\Delta t$. Matrices $\mathbf{C}$ and $\tilde{\mathbf{C}}$ are the face-edge incidence matrices of $\mathcal{G}$ and $\tilde{\mathcal{G}}$, respectively. Equations (2a) and (2c) are discrete equivalents of Ampère–Maxwell's law and Faraday's law respectively, and are exact equations. Equations (2b) and (2d) are discrete counterparts of the electric and magnetic constitutive relations and are instead approximate equations: the usual approach for approximating the material parameters ε and μ on a discrete domain is to assume their value to be uniform on each volume of the primal grid $\mathcal{G}$. The choice on how to construct these global discrete approximate relations affects the stability and consistency features of the algorithm. In [4], a method for constructing matrices $\mathbf{M}_{\mu^{-1}}$ and $\mathbf{M}_{\varepsilon^{-1}}$ based on the Discrete Geometric Approach (DGA) has been introduced, yielding an explicit, consistent and conditionally stable algorithm over tetrahedral meshes. We will refer to this new approach as DGA in time-domain (DGATD). In this approach $\mathbf{M}_{\mu^{-1}}$ is constructed locally on each tetrahedron in the mesh and $\mathbf{M}_{\varepsilon^{-1}}$ is constructed by locally inverting $\mathbf{M}_{\varepsilon}$ on each dual volume of the mesh. The accuracy and performance of the resulting numerical scheme have been studied on CPUs in [5], where a comparison with alternative implicit methods is also given. Differently from implicit schemes, where the most part of the computational burden is on the solution of a linear system for each time step of the simulation, in the explicit DGATD method the leapfrog scheme of the generic time step computation is

$$\mathbf{U}^n = \mathbf{U}^{n-1} + \Delta t \mathbf{M}_{\varepsilon^{-1}}\tilde{\mathbf{C}}\tilde{\mathbf{F}}^{n-\frac{1}{2}}, \tag{3a}$$
$$\tilde{\mathbf{F}}^{n+\frac{1}{2}} = \tilde{\mathbf{F}}^{n-\frac{1}{2}} - \Delta t \mathbf{M}_{\mu^{-1}}\mathbf{C}\mathbf{U}^n, \tag{3b}$$

in which the performance is bound by how efficiently we can perform matrix-vector products. As with the original Cartesian orthogonal scheme [6], we can show that the algorithm is amenable to massive parallelization: since each unknown in column vector $\mathbf{U}^n$, computed in (3a), depends only on the quantities computed in (3b) at time step $(n-\frac{1}{2})\Delta t$ in the two dual volumes that intersect the primal edge to which the unkown corresponds (see Fig. 1), we can compute the contributions of each dual volume to $\mathbf{U}^n$ concurrently. Similarly, each

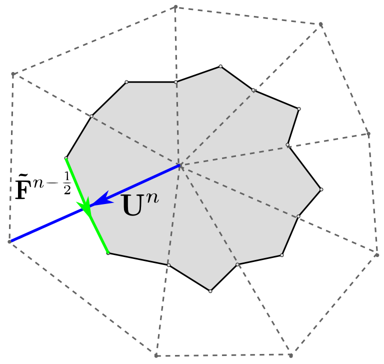

Fig. 1. Local quantities defined on the single dual cell and relation to neighboring cells. For the sake of clarity we show a 2D version.

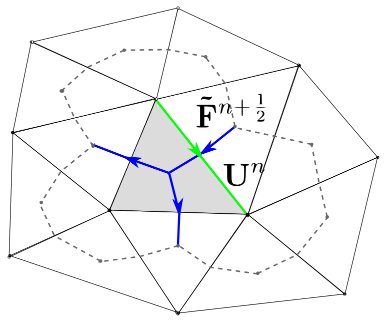

Fig. 2. Local quantities defined on the single primal cell and relation to neighboring cells. For the sake of clarity we show a 2D version.

unknown in column vector $\mathbf{F}^{n+\frac{1}{2}}$, computed in (3b), depends only on the quantities computed in (3a) at time step $n\Delta t$ in the two primal volumes that intersect the dual edge to which the unkown corresponds (see Fig. 2), and again we can compute the contributions of each primal volume to $\mathbf{F}^{n+\frac{1}{2}}$ concurrently.

II. Preliminary Numerical results

The DGATD method was implemented inside our own C++ electromagnetic framework (EMT) using the NVidia cuSparse library, which allows to perform sparse matrix-vector product (SpMv) on GPU without having to write our own CUDA kernels. We used a TESLA C2075 accelerator for all tests. The accelerator features the double-precision Fermi microarchitecture and compute capability 2.0; the underlying processor supports 448 threads and the available memory is 6 GB. For all meshes, we set $\Delta t = 1$ps to ensure stability on the finest mesh used. To add comparisons to a Cartesian orthogonal FDTD scheme, we used the free FDTD package MEEP [7]. It is worth noting that, as with Finite Elements, the DGATD method requires a smaller time step with respect to the Cartesian orthogonal FDTD scheme.

The accuracy and performance of the various competing methods is assessed on a problem for which a very accurate solution is available on Cartesian grids. We simulated a rectangular waveguide of size 5×2.5 cm and length 10 cm in the z direction. At $z = 0$, a TE$_{10}$ electric field is applied while at the other end ($z = 10$ cm) a Perfect Electric Conductor (PEC) termination is applied.

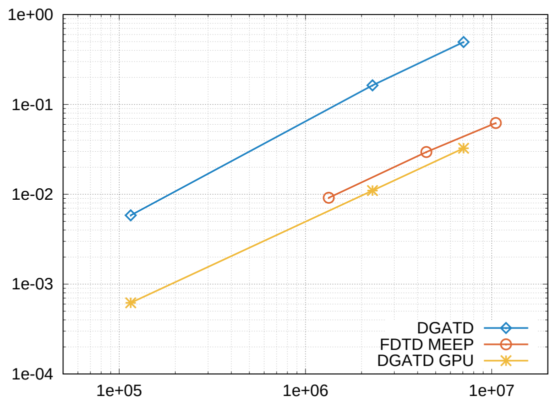

Fig. 3. Average computational time (seconds) of a single time step vs. the number of DOFs of the problem. Axes are in logarithmic scale.

In Fig. 3 we show the single time step average computational time (since the focus is on showing the improvement due to parallelization) versus the number of unknowns of the problem for DGATD on CPU, FDTD on CPU and DGATD on GPU. In average, this initial GPU version of the DGATD method is 10 times faster than the CPU version. Since the accelerator could provide far better performance in single precision, we are investigating this possibility and we will present the results about accuracy and computational efficiency in the full paper.

The non GPU-accelerated implementation of DGATD is the same we already used in [5], compiled with option -O3 on a machine with a Xeon E5-2687Wv4 processor. Algebraic operations are performed with the latest release (3.3.1) of the open-source *Eigen* linear algebra library.

III. Conclusions

Despite the GPU implementation of DGATD being preliminary and not yet fully optimized, we observe important speedups with respect to the CPU implementation. This suggests that the explicit nature of DGATD combined with the processing power of modern GPU accelerators can open the way for new and effective tools in computational electromagnetism.

References

[1] K.S. Yee, Numerical solution of initial boundary value problems involving Maxwells equations in isotropic media, *IEEE Transactions on Antennas and Propagation*, vol. AP-14, no. 3, pp. 302307, May 1966.
[2] T. Weiland, Time domain electromagnetic field computation with finite difference methods, *International Journal of Numerical Modeling*, vol. 9, pp. 295319, 1996.
[3] A. Taflove and S. Hagness, Computational Electromagnetics, The Finite Difference Time Domain Method, Second Edition. Boston, MA: Artech House, 2000.
[4] L. Codecasa, M. Politi, Explicit, Consistent and Conditionally Stable Extension of FD-TD by FIT, *IEEE Transactions on Magnetics*, Vol. 44, pp. 1258-1261, 2008.
[5] B. Kapidani, L. Codecasa, M. Cicuttin, R. Specogna, F. Trevisan, A comparative performance analysis of time-domain formulations for wave propagation problems, under review at *IEEE Transactions on Magnetics*.
[6] P. Sypek, A. Dziekonski, and M. Mrozowski, How to Render FDTD Computations More Effective Using a Graphics Accelerator, *IEEE Transactions on Magnetics*, vol. 45, no. 3, pp. 1324-1327, 2009.
[7] A.F. Oskooi, D. Roundy, M. Ibanescu, P. Bermel, J.D. Joannopoulos, S.G. Johnson, MEEP: A flexible free-software package for electromagnetic simulations by the FDTD method, *Computer Physics Communications*, Vol.181, pp. 687702, 2010.