Noname manuscript No.
(will be inserted by the editor)

On the use of Many-core Machines for the Acceleration of a Mesh Truncation Technique for FEM

Jose A. Belloch · Adrian Amor-Martin · Daniel Garcia-Donoro · Francisco J. Martínez-Zaldívar · Luis E. Garcia-Castillo

Received: date / Accepted: date

Abstract Finite Element Method (FEM) has been used for years for radiation problems in the field of electromagnetism. To tackle problems of this kind, mesh truncation techniques are required, which may lead to the use of high computational resources. In fact, electrically large radiation problems can only be tackled using massively parallel computational resources. Different types of multi-core machines are commonly employed in diverse fields of science for accelerating a number of applications. However, properly managing their computational resources becomes a very challenging task. On the one hand, we present a hybrid message passing interface (MPI)+OpenMP-based acceleration of a mesh truncation technique included in a FEM code for electromagnetism in a high performance computing (HPC) cluster equipped with 140 compute nodes. Results show that we obtain about 85% of the theoretical maximum speed-up of the machine. On the other hand, a graphics processing unit (GPU) has been used to accelerate one of the parts that presents high fine-grain parallelism.

Jose A. Belloch
Depto. de Tecnología Electrónica, Universidad Carlos III de Madrid, Spain
E-mail: jbelloc@ing.uc3m.es

Adrian Amor-Martin and Luis E. Garcia-Castillo
Dept. of Signal Theory and Communications, Universidad Carlos III de Madrid, Spain
E-mail: {aamor,legcasti}@ing.uc3m.es

Daniel Garcia-Donoro
School of Electronic Engineering
Xidian University, China
E-mail: daniel@xidian.edu.cn

Francisco J. Martínez-Zaldívar
Instituto de Telecomunicaciones y Aplicaciones Multimedia
Universitat Politècnica de València, Valencia, Spain
E-mail: fjmartin@dcom.upv.es

Keywords acceleration, parallelization, MPI, OpenMP, electromagnetism, finite elements.

1 Introduction

Finite Element Method (FEM) has been proven as a reliable, versatile and flexible tool for electromagnetism in the last decades. This method is based on the division of the physical domain into simpler geometrical shapes (tetrahedra are commonly used for a number of reasons) over which the solution is approximated by polynomials of a certain order. In this way, the original partial differential equations obtained from electromagnetism are translated into an algebraic system of equations. This code has been already verified and validated for several real problems, [1,2].

However, when FEM is applied to large scale problems in terms of electrical size, huge computational resources are required since the volumetric meshes mandatory in FEM lead to huge and highly sparse matrices. These matrices are commonly solved through direct solvers due to accuracy reasons and the necessity of sophisticated preconditioners to achieve convergence in iterative solvers, [3]. Moreover, the use of direct solvers allows an efficient introduction of a number of different right-hand sides (RHS), present in some problems, e.g., when computing radar cross section (RCS) of a target.

Moreover, when applying FEM to radiation problems, the free space has to be included in the volumetric mesh, which is critical since it can generate not only inaccurate solutions, but also increase significantly the computational time. Different mesh truncation techniques have already been provided in the literature, [4,5]. A technique based on a domain decomposition method (DDM), called finite element-iterative integral equation evaluation (FE-IIEE), was previously developed to obtain a certain accuracy in these radiation problems [6].

The fact that the current processors are composed of multiple cores has allowed to accelerate multiple applications in different fields. In fact, a high variety of programming technologies are available, including OpenMP [7] and MPI [8], that can be used for exploiting the cores of a processor and coordinating between different processors and machines. A platform that is suited for task-oriented parallelization is Xeon Phi [9]. Another kind of acceleration can be achieved by the use of Graphics Processing Units (GPUs). Following Flynn's taxonomy [10], from a conceptual point of view, a GPU can be viewed as a Single Instruction Multiple Data machine (SIMD), i.e., a computer in which a single flow of instructions is executed on different data sets. Thus, GPUs achieves high performances when the application presents fine-grain parallelism. The CUDA platform [11] provides a computing framework that enables the use of Graphics Processing Units (GPUs). Since the appearance of CUDA programming, many researchers in different areas have made use of it to achieve better performance in their respective fields. For example, in image processing [12], in acoustics [13], and even in solving electromagnetic problems

that involve the method of moments or finite-difference time-domain [14,15]. However, GPU is not a technique commonly used in FEM when using direct solvers to obtain the solution of the system of equations.

Two different accelerations are assessed in this work. On one side, we propose an MPI+OpenMP-based acceleration of a mesh truncation technique included on a finite element code for electromagnetism in a high performance computing (HPC) cluster equipped with 140 compute nodes. On the other side, we accelerate by using a GPU an specific part of the FE-IIEE that is composed of a high number of multiply and add (MAD) operations. For this last part, our work is focused on the issues of computing multiple RHS concurrently.

The rest of the paper is structured as follows: FEM formulation is briefly described in Section 2, flowchart of the parallelization for the truncation technique is included in Section 3; speed-up and numerical results are included in Section 4; and finally conclusions are drawn in Section 5.

2 Variational formulation

A weak formulation based on the double curl vector wave equation to characterize the electromagnetic field in a given problem domain is introduced in the FEM code. In the frequency domain, the double curl vector wave equation is

$$\nabla \times \frac{1}{\mu_r}(\nabla \times \mathbf{E}) - k_0^2 \varepsilon_r \mathbf{E} = \mathbf{O}, \tag{1}$$

where $\mathbf{E}$ is the electric field, k_0 is the wavenumber in vacuum, and $\mathbf{O}$ is the excitation related to impressed currents within the problem domain. Dirichlet Γ_{D}, Neumann Γ_{N} and Cauchy Γ_{C} boundary conditions are included in the code with the definitions

$$\hat{\mathbf{n}} \times \mathbf{E} = 0, \quad \text{on } \Gamma_{\mathrm{D}}, \tag{2}$$

$$\hat{\mathbf{n}} \times \frac{1}{\mu_{ri}}(\boldsymbol{\nabla} \times \mathbf{E}) = 0, \quad \text{on } \Gamma_{\mathrm{N}}, \tag{3}$$

$$\hat{\mathbf{n}} \times \frac{1}{\mu_{ri}}(\boldsymbol{\nabla} \times \mathbf{E}) + jk_0(\hat{\mathbf{n}} \times \hat{\mathbf{n}} \times \mathbf{E}) = \boldsymbol{\Psi}, \quad \text{on } \Gamma_{\mathrm{C}}, \tag{4}$$

where $\hat{\mathbf{n}}$ is the outward normal unit vector from the surface where the boundary condition is applied, and $\boldsymbol{\Psi}$ can be either the exterior boundary for open region problems or the excitation related to a waveport.

Then, Galerkin method is used on (1) to obtain the variational formulation of the problem as explained in the following.

Find $\mathbf{E} \in \mathbf{W}$ such that

$$c_1(\mathbf{F}, \mathbf{E}) - k_0^2\, c_2(\mathbf{F}, \mathbf{E}) + \gamma\, c_3(\mathbf{F}, \mathbf{E}) = l(\mathbf{F}), \quad \forall \mathbf{F} \in \mathbf{W}, \tag{5}$$

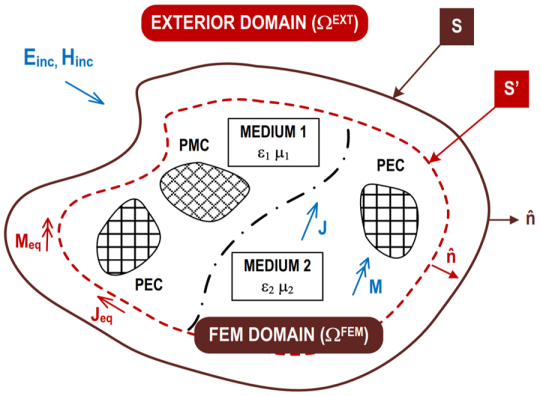


Fig. 1 Decomposition in two domains for FE-IIEE

where the bilinear (c_1, c_2 and c_3) and linear forms (l) are defined with

$$
\begin{aligned}
c_1(\mathbf{F},\mathbf{E}) &= \int_{\Omega} (\boldsymbol{\nabla}\times\mathbf{F})\cdot(\frac{1}{\mu_r}\boldsymbol{\nabla}\times\mathbf{E})\,d\Omega,\\
c_2(\mathbf{F},\mathbf{E}) &= \int_{\Omega} \mathbf{F}\cdot\varepsilon_r\mathbf{E}\,d\Omega,\\
c_3(\mathbf{F},\mathbf{E}) &= \int_{\Gamma_{\mathrm{C}}} (\hat{\mathbf{n}}\times\mathbf{F})\cdot(\hat{\mathbf{n}}\times\mathbf{E})\,d\Gamma_{\mathrm{C}},
\end{aligned}
\tag{6}
$$

$$
l(\mathbf{F}) = \int_{\Omega} (\mathbf{F}\cdot\mathbf{O})\,d\Omega - \int_{\Gamma_{\mathrm{C}}} (\mathbf{F}\cdot\boldsymbol{\Psi})\,\Gamma_{\mathrm{C}},
\tag{7}
$$

and the space of functions is

$$
\mathbf{W} := \{\mathbf{A}\in\mathbf{H}(\mathrm{curl},\Omega),\ \hat{\mathbf{n}}\times\mathbf{A}=0 \text{ on } \Gamma_D\}.
\tag{8}
$$

To discretize the FEM domain, second order isoparametric curl-conforming tetrahedral and triangular prismatic finite elements, [16,17] are used.

For open region problems, as introduced in Sec. 1, a non-standard mesh truncation technique (FE-IIEE, [6]) is included. FE-IIEE is based on a two-domain decomposition multiplicative Schwarz paradigm, dividing the original infinite domain into two overlapping domains limited by a finite FEM domain bounded by an exterior surface S and by an infinite domain exterior to the auxiliary boundary S' located within S, as it is shown in Fig. 1.

In practice, the distance between S and S' is a small fraction of the wavelength, typically in the range of $0.05\,\lambda$ to $0.2\,\lambda$, hence allowing the truncation boundary to be placed very close to the sources. Following the Schwarz paradigm, FE-IIEE iterates between these two domains until the scattering field error between two consecutive iterations is lower than a given threshold or until the maximum number of iterations is reached. Note that these iterations only involve changes on the right-hand side of the problem. This means

that the FEM matrix needs to be factorized only in the first iteration, and only forward and backward substitutions are involved in the remaining iterations when a direct solver is used.

Specifically, the integral equation representation of the exterior field to S' provides an improved version of the residual function $\boldsymbol{\Psi}$ related to the Cauchy boundary conditions at each iteration step. Thus, an asymptotically exact absorbing boundary condition for FEM is implemented retaining the original sparse nature of the finite element matrices. In practice, the use of the FE-IIEE method means that function $\boldsymbol{\Psi}$ now is the weighted sum of $\boldsymbol{\Psi}_{\text{inc}}$ and $\boldsymbol{\Psi}_{\text{scat}}$, [6], where $\boldsymbol{\Psi}_{\text{inc}}$ is set to zero for radiation problems and $\boldsymbol{\Psi}_{\text{scat}}$ is computed in each iteration of the FE-IIEE method. The computation of $\boldsymbol{\Psi}_{\text{scat}}$ requires a number of operations which reminds of method of moments, with different computational requirements than FEM. Indeed, this specific part of the code is the most suitable to be run on GPU since it is composed of the combination of multiple parameters. Firstly, Green's function is computed, whose operations are implemented by a dozen of multiplications between complex numbers, which involves 5 operations each (4 real multiplications and a sum). Afterwards, combinations of J-current and M-current are used to obtain the potential $\mathbf{V}$ and $\boldsymbol{\nabla} \times \mathbf{V}$ (see blue box in Fig.2 and note that for this formulation, based on electric field, $\mathbf{V} = \mathbf{E}$), which involves the combination of thousands of complex elements. All of the described operations are carried out independently for each of the RHS. Thus, porting this part of the code that fits within the SIMD paradigm to a GPU can provide a boost in the performance.

In Fig. 2 more details about how FE-IIEE is embedded in the FEM code are included. Typical flowchart for FEM solver is included in the green box while blue boxes show how boundary conditions are imposed on domain S. The boundary conditions are updated until the convergence criteria are achieved.

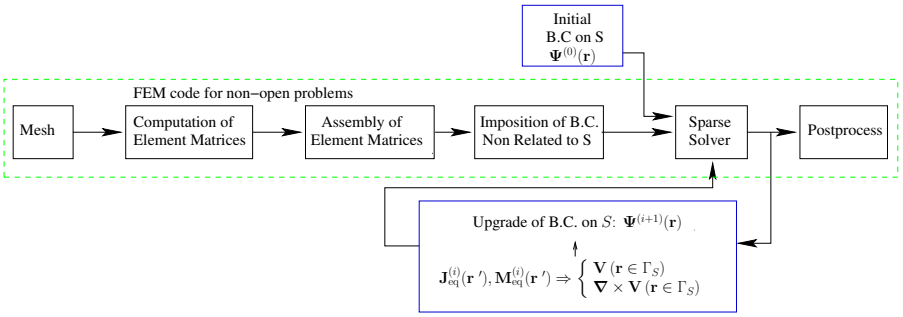


Fig. 2 Conceptual flowchart for FE-IIEE.

3 Parallelization

The code was initially designed for small HPC cluster environments, and it has recently experienced a number of modifications in order to be able to run on large-scale computer systems and hence, to be able to deal with larger problems in terms of number of unknowns. Thus, the code has been written from scratch to make an efficient use of HPC platforms. HOFEM implements a hybrid parallel methodology based on the use of MPI processes and OpenMP threads within each process. This hybrid methodology is already used in direct solvers as MUMPS, [18], which leveraged from the code to obtain the solution with high accuracy from the system of equations generated with FEM so it is efficient to use the same scheme in the code.

A workflow particularized for the FE-IIEE method is included in the Fig. 3. Here, a process is denoted as a box (numbered as `p0`, `p1`, and so on) and a thread is plotted as a smaller box below the process being filled with color if the thread is used. In this example, four threads per process are shown in the workflow. At first, all the input data is read and then, for each frequency, a typical assembly of FEM system of equations is performed. In this assembly, different elements are assigned to each process while loops are accelerated with OpenMP threads. To solve the system of equations, MUMPS is used —as a direct solver— and if the problem is open region, FE-IIEE is enabled so that the scattering field is computed, the right-hand side is updated only in `p0`, and then the system of equations is solved. This is repeated until the convergence error is achieved or the maximum number of iterations is reached. In this solving step, the matrix is factorized in the first iteration and then forward and backward substitutions are applied for the remaining iterations. In this loop, the performance of OpenMP is critical to attain a good speedup as shown in Section 4.

It has to be noted how the number of RHS affects to the scalability of the code. The number of RHS is related, e.g., to the spatial resolution when computing monostatic radar cross section (RCS): if plane for $\theta = 90^\circ$ is characterized with an angle sampling of two degrees, 181 points — which means 181 excitations, so 181 RHS if no more excitations are present in the electromagnetic problem — will be needed; in fact, if a 3D half-sphere wants to be characterized with the same angle sampling, 8281 RHS are needed —91 points in θ planes (from $\phi = 0$ to $\phi = 180^\circ$), and 91 possible values of θ(from $\theta = 0$ to $\theta = 180^\circ$)—. The most expensive part in terms of computational time is the factorization of the matrix, which is performed once for each frequency. Then, to add more RHS when using a direct solver simply means applying more forward and backward substitutions. However, the scattering field used in FE-IIEE method has to be computed for each different RHS, which makes it suitable for accelerating with GPU due to the distinctive features of its computation.

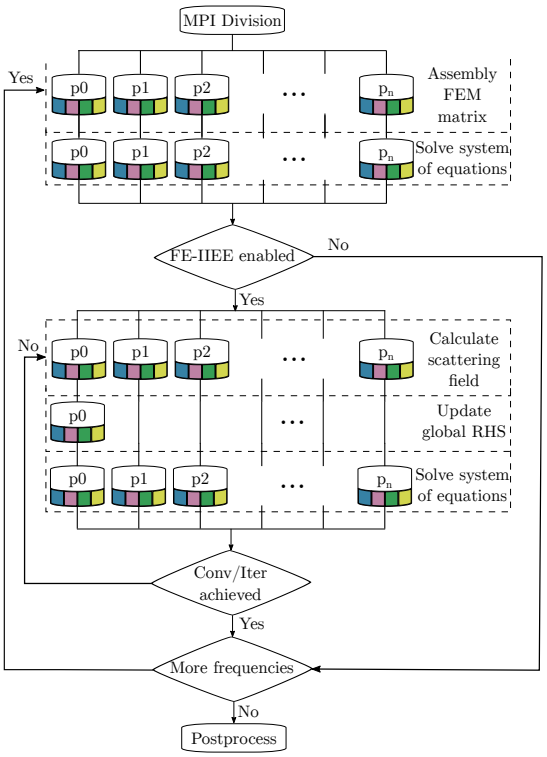


Fig. 3 Parallel workflow for FE-IIEE.

3.1 GPU-based implementation for accelerating RHS

The parallelization of the GPU consists on launching as many GPU-threads (executions in parallel that are launched at the GPU) as the number of RHS in the electromagnetic problem, so that each GPU-thread deals independently with one RHS. In fig. 4, the parallelization of the computations of $\boldsymbol{\Psi}_{\text{scat}}$ composed of multiple RHS is shown. Performance tests were carried out using an Intel(R) Xeon(R) CPU E5-2697 v2 @ 2.70GHz and a Nvidia GPU K20c, [19].

The integration of the GPU code with the original HOFEM software has been cumbersome since HOFEM was developed in Fortran and a wrapper that connecting both codes has been developed. As a consequence, the number of memory transactions between GPU and CPU has been increased and thus, the boost in the performance has not been as significant as it was expected initially. Despite of these facts, Fig. 5 shows that, using a GPU for computing a scattering vector $\boldsymbol{\Psi}_{\text{scat}}$, a better performance than original HOFEM software with CPU is obtained when the scattering field is composed by more than 2000 RHS.

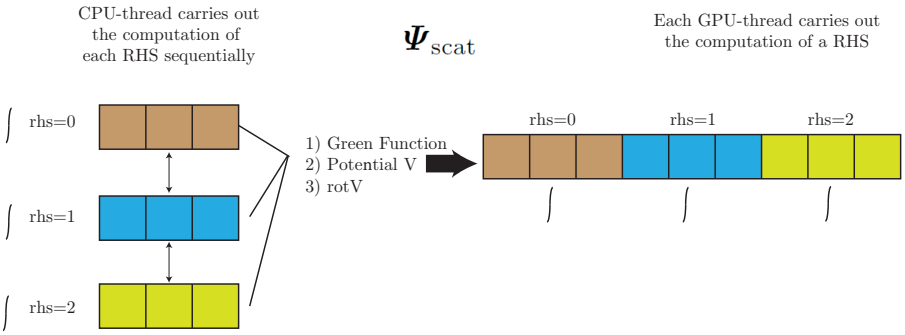


Fig. 4 GPU-based parallelization of the scattering vector that is composed of multiple RHS

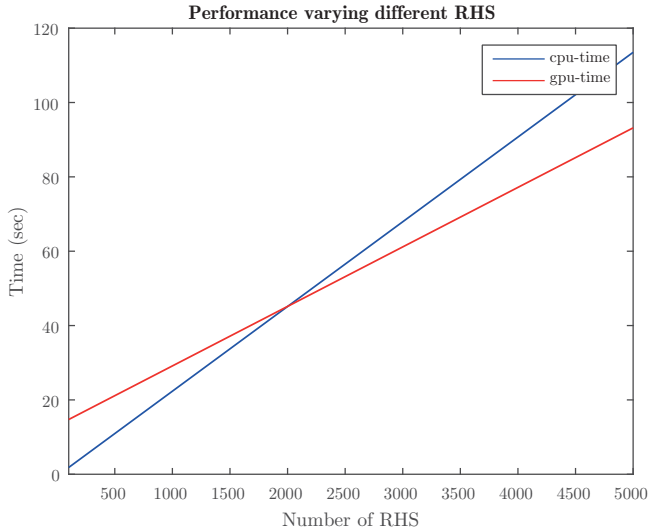
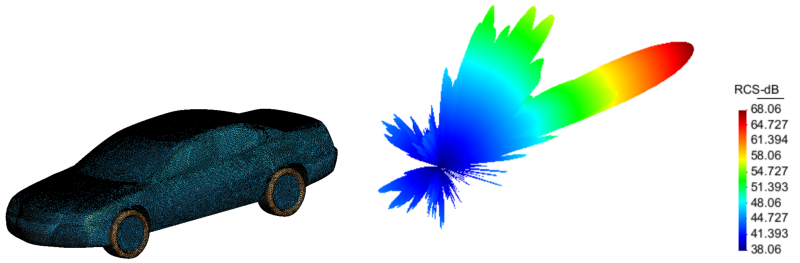


Fig. 5 Performance comparison between the use of the CPU and the use of GPU inside the HOFEM software in order to compute $\boldsymbol{\Psi}_{\text{scat}}$ with multiple RHS.

4 Results

Given aside the acceleration using the GPU implementation, the speedup of the whole code in an HPC environment —HPC cluster of Xidian University (XDHPC), equipped with 140 compute nodes (each one with two twelve-core Intel Xeon 2690 V2 2.2GHz CPUs with 64 GB RAM and 1.8 TB hard disk) connected by 56 Gbps InfiniBand network— is included. Here, the second order basis functions for tetrahedra introduced in [16] are used. A snapshot of the car together with the 3D representation of the RCS of the car at 500 MHz is illustrated in Fig. 6.

Fig. 7 reports the parallel performance of the FEM code considering only the execution of the FE-IIEE truncation technique. As it was described in



(a) Geometry of the problem. (b) RCS (planewave illuminates the front)

Fig. 6 RCS analysis of a Chevrolet impala at 500 MHz.

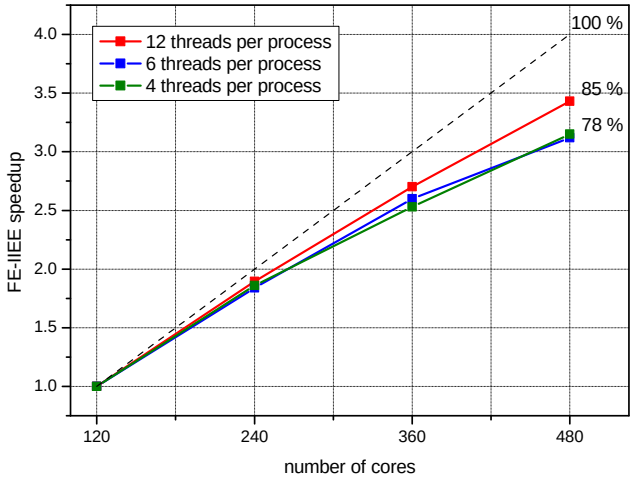


Fig. 7 Speedup graph corresponding to the mesh truncation phase

the previous section, once the code computes the solution of the problem, the FE-IIEE technique is executed (when enabled), avoiding any undesired reflection from the truncation boundary that can disturb the solution. The parallel performance obtained by HOFEM during these phases is near 85% almost in all the configurations when using 480 cores. For a smaller number of cores the performance is even higher. It is worth mentioning that, during the execution of the FE-IIEE method, a back substitution process is required in order to compute the solution of the system, slightly degrading parallel performance.

5 Conclusions

We have presented a GPU-based implementation in order to accelerate one specific part of the FE-IIEE that is composed of multiple RHS. The proposed

GPU implementation outperforms the CPU implementation when the number of RHS is upper to 2000.

On the other hand, we have presented a FEM formulation developed from scratch to be used in HPC environments. Details of the acceleration of a non-standard mesh truncation technique designed for open problems are given. Results show that up to 85% of the theoretical maximum speedup of the machine is obtained with the computation of an RCS of a target modeled as a car.

Acknowledgements

This work has been financially supported by TEC2016-80386-P, TIN2017-82972-R, CAM S2013/ICE-3004 projects and "Ayudas para contratos predoctorales de Formación del Profesorado Universitario FPU".

References

1. D. Garcia-Donoro, L. E. García-Castillo, and S. W. Ting, "Verification process of finite-element method code for electromagnetics: Using the method of manufactured solutions," *IEEE Antennas and Propagation Magazine*, vol. 7, no. 2, pp. 28–38, Apr. 2016.
2. D. Garcia-Donoro, S. Ting, A. Amor-Martin, and L. E. Garcia-Castillo, "Analysis of planar microwave devices using higher order curl-conforming triangular prismatic finite elements," *Microwave and Optical Technology Letters*, vol. 58, no. 8, pp. 1794–1801, Aug. 2016.
3. O. G. Ernst and M. J. Gander, "Why it is difficult to solve helmholtz problems with classical iterative methods," in *Numerical analysis of multiscale problems*. Springer, 2012, pp. 325–363.
4. J.-P. Bérenger, "Perfectly matched layer (PML) for computational electromagnetics," *Synthesis Lectures on Computational Electromagnetics*, vol. 2, no. 1, pp. 1–117, 2007.
5. W. J. P. and K. V. N., "Absorbing boundary conditions for the finite element solution of the vector wave equation," *Microwave and Optical Technology Letters*, vol. 2, no. 10, pp. 370–372. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/mop.4650021010
6. R. Fernandez-Recio, L. E. Garcia-Castillo, I. Gomez-Revuelto, and M. Salazar-Palma, "Convergence study of a non-standard Schwarz domain decomposition method for finite element mesh truncation in electromagnetics," *Progress In Electromagnetics Research (PIER)*, vol. 120, pp. 439–457, 2011.
7. "OpenMP: The OpenMP API specification for parallel programming," http://openmp.org/.
8. "Message Passing Interface Forum," http://www.mpi-forum.org/.
9. A. Sodani, R. Gramunt, J. Corbal, H. Kim, K. Vinod, S. Chinthamani, S. Hutsell, R. Agarwal, and Y. Liu, "Knights landing: Second-generation Intel Xeon Phi product," *IEEE Micro*, vol. 36, no. 2, pp. 34–46, Mar 2016.
10. M. Flynn, "Some computer organizations and their effectiveness," *IEEE Transactions on Computers*, vol. 21, pp. 948–960, 1972.
11. "Nvidia CUDA Developer Zone," https://developer.nvidia.com/cuda-zone, (accessed 2014 April 10).
12. W. Liu, B. Schmidt, G. Voss, and W. Muller-Wittig, "Streaming algorithms for biological sequence alignment on GPUs," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 9, pp. 1270–1281, 2007.

13. J. A. Belloch, A. Gonzalez, F. J. Martínez-Zaldívar, and A. M. Vidal, "Real-time massive convolution for audio applications on GPU," *Journal of Supercomputing*, vol. 58, no. 3, pp. 449–457, December 2011.
14. S. Peng and Z. Nie, "Acceleration of the method of moments calculations by using graphics processing units," *IEEE Transactions on Antennas and Propagation*, vol. 56, no. 7, pp. 2130–2133, 2008.
15. D. De Donno, A. Esposito, L. Tarricone, and L. Catarinucci, "Introduction to GPU computing and CUDA programming: A case study on FDTD [EM programmer's notebook]," *IEEE Antennas and Propagation Magazine*, vol. 52, no. 3, pp. 116–122, 2010.
16. M. Salazar-Palma, T. K. Sarkar, L. E. García-Castillo, T. Roy, and A. R. Djordjevic, *Iterative and Self-Adaptive Finite-Elements in Electromagnetic Modeling.* Norwood, MA: Artech House Publishers, Inc., 1998.
17. A. Amor-Martin, D. Garcia-Donoro, and L. E. Garcia-Castillo, "Second-order Nedelec curl-conforming prismatic element for computational electromagnetics," *IEEE Transactions on Antennas and Propagation*, vol. 64, no. 10, pp. 1–12, Oct. 2016.
18. "MUMPS Solver," http://mumps.enseeiht.fr/.
19. K20, "NVIDIA Kepler Architecture," http://www.nvidia.com/content/PDF/kepler/NVIDIA-Kepler-GK110-Architecture-Whitepaper.pdf, 2014, (accessed 2018 July 31).