Efficient PEEC-Based Simulations using Reluctance Method for Power Electronic Applications

Danesh Daroui and Jonas Ekman

Department of Computer Science, Electrical and Space Engineering
Luleå University of Technology, Luleå, 971 87, Sweden
danesh.daroui@ltu.se, jonas.ekman@ltu.se

Abstract ─ This paper presents a partial element equivalent circuit (PEEC)-based solver that has been accelerated to exploit the massively parallel structure of graphics processing unit (GPU) technology, in order to employ a reluctance-based method in an efficient way. A grouping algorithm is also presented which makes reluctance calculation efficient, suitable for GPUs, and feasible even for very large problems. It has been shown that by using the reluctance method, the coefficient matrix in the system equation can be safely sparsified whilst the required accuracy is maintained. Because the calculation of the reluctance matrix includes matrix inversion, which is a task with high computational complexity, GPUs as cooperative units are utilized to reduce computational costs by taking advantage of parallelism. Two test models have been simulated and analyzed to benchmark the solver, and the results have been compared with the previously developed solver. Furthermore, analyzing the results reveals that the reluctance method makes it possible to use a considerably sparser system and thereby solve large problems by decreasing the memory demands and the solution time. It is also proven that the solution is reliable and accurate, whereas the problem has become noticeably smaller.

Index Terms - PEEC, electromagnetic simulation, reluctance, GPU.

I. INTRODUCTION

High frequency electronic circuits have a very important role in modern electronic devices. When the operational frequency of electronic devices is increased, magnetic and electrical couplings can cause unexpected results, such as current imbalance, thermal hotspots and chip overload, in a high frequency circuit. Moreover, other phenomena such as signal distortion, crosstalk and ground bouncing [1] need to be taken into account. Examination of these phenomena, leads to electromagnetic compatibility (EMC) standards which demand the compliance with limitations for electromagnetic interferences. Therefore, studying susceptibility and emission is an important aspect of high frequency circuit design.

Various methods have been developed to address solutions for electromagnetic analysis. Among these methods, partial element equivalent circuit (PEEC) [2-5] is known to be suitable for combined electromagnetic and circuit analysis and has been widely used in power electronics, PCB design, antenna design and other industrial applications. As the complexity of the electrical systems and devices and the working frequency is increasing continuously [6], new acceleration techniques should be developed to solve real world problems in a reasonable time using limited computational resources. In previous works, other acceleration techniques have been used to increase the performance of the PEEC-based solver for general EMC applications, where no approximation is involved [7-8]. This paper presents a reluctance-based technique that can be employed to solve complex PEEC models in less time and with less memory consumption while some approximations are included in the final solution. A grouping algorithm is proposed to partition the geometry in 3D space and assemble the partial inductances into block diagonal form, to efficiently calculate the reluctance matrix.

Submitted On: Apr. 17, 2012
Accepted On: Oct. 1, 2012

Additionally, to the best knowledge of the authors of this article, no PEEC-based solver which is based on GPUs has been reported. However, since GPU technology is relatively new, hence the available hardware and software tools that have been employed in this research are strictly limited to the problems that can fit into GPU internal memories which are usually less than the main memory which is available on modern machines. The presented approach in this work is notably suitable for power electronic systems analysis, since capacitive couplings can be ignored and time retardation is not assumed in the simulations. It has been proven that using reluctances instead of partial inductances, makes it possible to eliminate a large amount of data from the equations, as long as the solution is kept to be accurate and valid [9]. On the other hand, calculation of the reluctance matrix is an expensive process and requires matrix inversion which has a high order in computational complexity which is $O(n^3)$ for a $n \times n$ matrix. In particular, for large structures that will result in large inductance matrices, reluctance calculation can be costly or even impossible to carry out. Hence, parallel processing approach in order to accelerate the reluctance calculation can be considered as a reasonable solution. Typically, modern processing units are enhanced with only few cores, whereas GPUs installed on modern graphic cards can offer processors with hundreds of cores and few GFLOPS of computational performance which makes these units appropriate for extensive and costly computations. Due to the high computational power and availability of GPUs, the technology has successfully used in many research areas, including electromagnetic simulations [10-12] and more specifically, FDTD [13][14] and Method of Moments [15]. It is also shown that the performance gained when a GPU is used is dependent on the problem size, which means that larger problems result in better speedups on GPU-enhanced graphic cards [15].

Fortunately, reluctance method in combination with sparsification techniques will bring a system of equations which consists of a sparse coefficient matrix. By having sparser systems, memory consumption will decrease, and sparse solvers can be used that can solve very large systems in less time than conventional direct solvers which are tailored for dense matrices. The PEEC-based solver presented in this paper is based on reluctance technique when only inductive couplings exist in the model and capacitive couplings have been neglected. This assumption is primarily valid for applications in power electronics where high current electrical devices are studied where inductance extraction plays an important role because of high $\frac{di}{dt}$ which can cause high induced voltages in a circuit.

II. THE PEEC THEORY

This section gives a brief summary of the classical PEEC formulation. For further information, see [2-5].

A. Extraction of the equivalent circuit

The classical PEEC method is derived from the equation for the total electric field at a point [16] written as

$$\vec{E}^i(\vec{r},t) = \frac{J(\vec{r},t)}{\sigma} + \frac{\partial \vec{A}(\vec{r},t)}{\partial t} + \nabla \phi(\vec{r},t), \qquad (1)$$

where $\vec{E}^i$ is an incident electric field, J is a current density, $\vec{A}$ is the magnetic vector potential, _ is the scalar electric potential, and ϕ is the electrical conductivity all at observation point $\vec{r}$. By using the definitions of the scalar and vector potentials, the current- and charge-densities are discretized by defining pulse basis functions for the conductors and dielectric materials. Pulse functions are also used for the weighting functions, resulting in a Galerkin type solution. By defining a suitable inner product, a weighted volume integral over the cells, the field equation (1) can be interpreted as Kirchhoff's voltage law over a PEEC cell consisting of partial self inductances between the nodes and partial mutual inductances representing the magnetic field coupling in the equivalent circuit. The partial inductances shown as L_{P11} and L_{P22} in Fig. 1 are defined as

$$L_{P\alpha\beta} = \frac{\mu}{4\Pi} \frac{1}{a_\alpha a_\beta} \int_{a_\alpha} \int_{a_\beta} \int_{l_\alpha} \int_{l_\beta} \frac{dl_\alpha \bullet dl_\beta}{\left|\vec{r}_\alpha - \vec{r}_\beta\right|} da_\alpha da_\beta \qquad (2)$$

for volume cell α and β. Figure 1 also shows the node capacitances which are related to the

coefficients of potential p_{ii} while ratios consisting of p_{ij}/p_{ii} are leading to the current sources in the PEEC circuit. The coefficients of potentials are computed as

$$P_{ij} = \frac{1}{S_i S_j} \frac{1}{4\Pi\varepsilon_0} \int_{S_i} \int_{S_j} \frac{1}{|\vec{r}_i - \vec{r}_j|} dS_j dS_j, \quad (3)$$

and a resistive term between the nodes is defined as

$$R_\gamma = \frac{l_\gamma}{a_\gamma \sigma_\gamma} \quad (4)$$

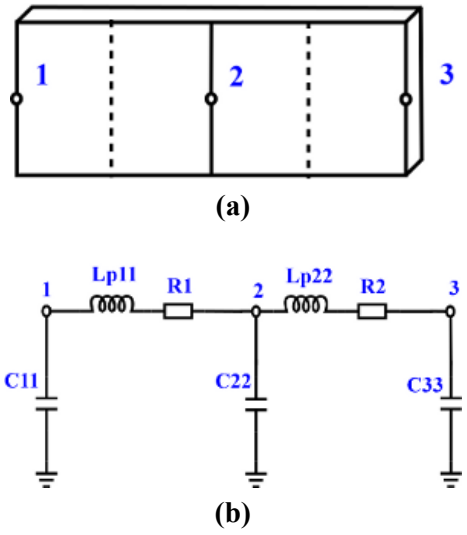


Fig. 1. Metal strip with 3 nodes and 2 cells (a) and corresponding PEEC circuit (b) (mutual couplings are not shown).

In (2), (3) and (4), a represents the cross section of the rectangular volume cell normal to the current direction γ, l is the length in the current direction and S is the charge surface cells. For a detailed derivation of the method, including the non-orthogonal formulation, see [17].

B. Solution of the equivalent circuit

The discretization process of the EFIE in (1) and the successive Galerkin's weighting leads to an equivalent circuit formulation. When Kirchhoff's voltage and current laws are enforced to the N_i independent loops and N_ϕ independent nodes of the PEEC equivalent circuit, the following equations are obtained

$$\begin{aligned} (j\omega P^{-1} + Y_L) I_L - A^T \phi &= I_S \\ -AI_L + (R + j\omega L_P)\phi &= V_S \end{aligned} \quad (5)$$

where

- $\phi \in \Re^{N_\phi}$ is the vector of node potentials to infinity; $\Re^{N_\phi}$ is the node space of the equivalent network;
- $I_L \in \Re^{N_i}$ is the vector of currents including both conduction and displacement currents; $\Re^{N_i}$ is the current space of the equivalent network;
- L_P is the matrix of partial inductances describing the magnetic field coupling;
- P is the matrix of coefficients of potential describing the electric field coupling;
- Y_L is an admittance matrix describing the lumped components connected to the PEEC model;
- R is the matrix of cell resistances;
- A is the connectivity matrix which describes the current direction between each pair of nodes, assigned to each cell;
- V_S is the vector of distributed voltage sources due to external electromagnetic fields or lumped voltage sources;
- I_S is the vector of lumped current sources.

The equation system in (5) is equivalent to the circuit equations formulated in SPICE-type solvers for obtaining the solution in node voltages and branch currents. However, for PEECs the equation system in (5) contains denser matrices (L_P and P) than a pure electric network system solution due to the large number of mutually coupled inductors and mutual capacitances. Therefore, the solution of PEECs requires linear algebra packages suitable for dense matrices. The equation system in (5) is often entitled a Modified Nodal Analysis (MNA) formulation [18] and can be modified to suit the solution of PEECs [19]. Neglecting capacitive couplings from the PEEC model will lead to (6) which is the reduced form of (5) and discussed in this paper.

$$\begin{aligned} Y_L I_L - A^T \phi &= I_S \\ -AI_L - (R + j\omega L_P)\phi &= V_S \end{aligned} \quad (6)$$

The next section explains that how the dense system derived from MNA formulation is converted to a sparse system, without getting numerically unstable and invalid solution [20-21].

III. SPARSE MATRIX FORMULATIONS USING THE RELUCTANCE METHOD

In this section, the reluctance method and its application to sparsifying systems of linear equations in PEEC are discussed.

A. The reluctance method

From the partial inductance matrix L_P, the reluctance matrix K is defined as the inverse of L_P.

$$K = L_P^{-1} \tag{7}$$

By multiplying L_P with a vector of N branch currents, a vector containing the drop of the magnetic vector potential along each segment will be obtained, as shown in (8).

$$\begin{bmatrix} L_{11} & L_{12} & \dots & L_{1N} \\ L_{21} & L_{22} & \dots & L_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ L_{N1} & L_{N2} & \dots & L_{NN} \end{bmatrix} \begin{bmatrix} i_1 \\ i_2 \\ \vdots \\ i_N \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{N} (\int A_{1i} d\vec{l}_1) \\ \sum_{i=1}^{N} (\int A_{2i} d\vec{l}_2) \\ \vdots \\ \sum_{i=1}^{N} (\int A_{Ni} d\vec{l}_N) \end{bmatrix} \tag{8}$$

$$\begin{bmatrix} K_{11} & K_{12} & \dots & K_{1N} \\ K_{21} & K_{22} & \dots & K_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ K_{N1} & K_{N2} & \dots & K_{NN} \end{bmatrix} \begin{bmatrix} \sum_{i=1}^{N} (\int A_{1i} d\vec{l}_1) \\ \sum_{i=1}^{N} (\int A_{2i} d\vec{l}_2) \\ \vdots \\ \sum_{i=1}^{N} (\int A_{Ni} d\vec{l}_N) \end{bmatrix} = \begin{bmatrix} i_1 \\ i_2 \\ \vdots \\ i_N \end{bmatrix} \tag{9}$$

To extract values of the reluctance matrix, the linear equation system (9) should be solved. Therefore, unlike the elements of the partial inductance matrix, there is no formulation to calculate the reluctance values directly. Therefore, the L_P matrix is needed to calculate the K matrix. The physical meaning of K_{ij} is defined as the induced current in the i^{th} conductor (aggressor) when the total flux for the j^{th} conductor (victim) is equal to one and the total flux along each other conductor is set to zero. This definition suggests that the induced current in victims should be in the opposite direction of the current in the aggressor to keep the flux around all victims at zero. These induced currents will generate magnetic fields around victims, which cancel the part of the field induced on the aggressor and shield the field on the aggressor from going farther. This phenomenon explains the locality (shielding effect) of the reluctance matrix [9]. The locality property of the method has been demonstrated in Fig. 2, where the third conductor results in a shorter arrow for the induced current, accounting for the overall effect. Figure 2 also shows the induced current in the third conductor, contributed to by two other bars, in dashed lines, and the overall effect of the induced currents using a solid line arrow. Because of this rippling effect of the magnetic field and the induced current, the value of each K_{ij} in the reluctance matrix represents the overall effect rather than a single active line. Therefore, the off-diagonal values in K would decrease more rapidly than L_P, which exhibits locality and the shielding effect of the reluctance matrix [22].

Like the capacitance matrix, the reluctance matrix is supposed to be symmetric positive definite and have the locality property [20]. Hence, only a small number of elements in the K matrix need to be kept to maintain an appropriate level of accuracy of the solution. The stability of the method is directly related to how the structure is discretized. It has been proven that stability is ensured for a sufficiently discretized structure [21].

B. Extraction of the reluctance matrix

As stated in the previous section, there is no formulation to calculate reluctance matrix directly and thus, the inductance matrix must be calculated first and then inversion carried out. Because matrix inversion is a cumbersome numerical task with time complexity of $O(n^3)$ for a $n \times n$ matrix, inverting a large matrix can highly degrade the overall performance of the solution. Several algorithms have been developed to overcome this problem by defining a window over a structure and then traversing the geometry using the window and the calculating the inductive couplings within the covered area and then finally

extract values for the reluctance sub-matrix [23][24]. In window-based approaches, the stability of the method is highly dependent on the window size. Further, choosing an unnecessarily large window in order to reach a high level of the accuracy can make the reluctance extraction computationally expensive and a small window will return inaccurate results. Therefore, finding an optimum size for the window to be traversed is crucial and usually challenging and hard to find.

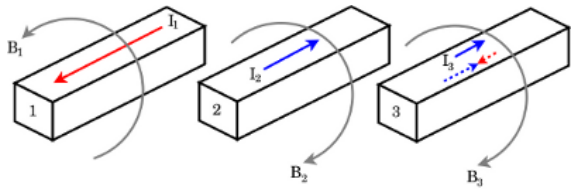


Fig. 2. The effect of reluctance on three parallel bars.

A novel approach to calculate reluctance matrix is proposed in this paper which is based on grouping the parts of the structure along axes, meaning that three main groups are created, which contain parts of the model in the x- , y- and z-axis. Because elements in each group are geometrically perpendicular to the other groups, there will be no inductive couplings between groups and only couplings within each group are considered. This assumption leads to the existence of a block diagonal partial inductance matrix where each block can simply be inverted to determine the reluctance matrix. The total process of inverting all blocks is substantially less costly than inverting the whole matrix. By holding the assumption that the studied structure is uniformly placed in the space along all three axes, each block will have roughly the dimension size of $\frac{1}{3}$ of the dimensions of partial inductance matrix, as shown in (10). By substituting (10) into (11), the time complexity of the inversion of the original matrix is compared to the time complexity of the inversion of each block. According to (11), the maximum speedup that can be achieved is approximately 9, which is a worthwhile improvement in the performance. Moreover, it can be concluded that the reluctance calculation can be a bottleneck in the total solution. Thus, a massively parallel approach using GPU technology to accelerate the performance of this part of the solution is a reasonable solution.

$$n_x \approx \frac{n}{3}, n_y \approx \frac{n}{3}, n_z \approx \frac{n}{3} \tag{10}$$

$$\frac{O((n_x + n_y + n_z)^3)}{O(n_x{}^3) + O(n_y{}^3) + O(n_z{}^3)} \leq 9 \tag{11}$$

Figure 3 depicts the reluctance matrix, calculated from a block diagonal partial inductance matrix, where each block which represents parts of the model in each axis, is inverted separately.

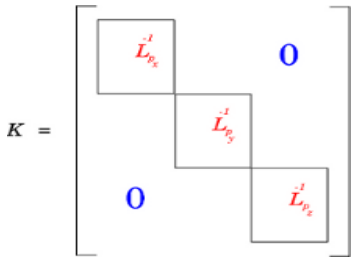


Fig. 3. Calculating reluctance from an inductance matrix in block diagonal form.

C. Reluctance formulation in PEEC

In this paper, only quasi-static (R,Lp)PEEC is studied, where capacitive couplings are neglected in the problem. This assumption is mostly valid in power electronics and for the models that carry high currents, e.g. bus bars in power frequency converters, where inductive loops are interesting to be identified and studied in order to simulate the induced voltage when $\frac{di}{dt}$ is high in such a circuit. Extracting inductance values in high frequency power electronics circuits will help to improve the layout design to reduce the stray inductance and consequently reduce voltage overshoots and reduce switching losses. Based on the MNA formulation in PEEC, the system equation for an (R,Lp)PEEC model is defined as

$$\begin{bmatrix} Y_L & -A^T \\ -A & R + j\omega L_P \end{bmatrix} \begin{bmatrix} \phi \\ I_L \end{bmatrix} = \begin{bmatrix} I_S \\ V_S \end{bmatrix} \tag{12}$$

All elements in (12) are described in Section II-B. In system equation (12), the right hand side is for excitation sources and the unknowns in the equation are *node potential* and *cell currents*.

By applying the reluctance matrix, as defined in (7) to both sides of (12), the new formulation which is shown in (13) will be achieved. This new formulation is introduced as (R,K)PEEC.

$$\begin{bmatrix} Y_L & -A^T \\ -KA & KR + j\omega I \end{bmatrix} \begin{bmatrix} \phi \\ I_L \end{bmatrix} = \begin{bmatrix} I_S \\ KV_S \end{bmatrix} \tag{13}$$

IV. PEEC-BASED SOLVER

This section describes a solver developed based on the PEEC method. Details about solver acceleration using GPUs and sparse solver are also presented in this section.

A. Implementation of the solver

A PEEC-based solver has been written in the C++ language, which utilizes the Intel Math Kernel Library (MKL) [25], optimized for multi-core systems, and the CUDA-based LAPACK (CULA) [26], which is included for the parts of the solver that use the GPU. To solve the sparse system after applying the reluctance matrix, the MUMPS package [27] is used. After calculating each block of the reluctance matrix, it can safely be sparsified. The sparsification can be described as applying a truncation process to small off-diagonal elements in the reluctance matrix by means of the *coupling factor*

$$k_{ij} = \sqrt{\frac{K_{ij}^{\ 2}}{|K_{ii}| \bullet |K_{jj}|}} \tag{14}$$

between the elements i and j in the matrix. The calculated factor will be compared to a constant value and values that are smaller than the constant will be removed [28]. The approximation in (13) comes from the relation in (15), which is due to the sparsified reluctance matrix. As the K matrix is more sparsified, (15) is more approximate. Similarly, if K matrix is not sparsified at all, the equation $KL_P = I$ will hold which leads to the exact solution.

$$KL_P \approx I \tag{15}$$

It should be noted that, sparser reluctance matrices will create sparser coefficient matrices, which will speed up the solving process. Moreover, due to the locality property of the reluctance matrix, K can be sparsified up to a high level, at which the desirable level of accuracy is still maintained [22]. In a PEEC-based solution, the coefficient matrix in the MNA formulation consumes the largest part of the total allocated memory. Using (R,K)PEEC, this matrix is converted to a sparse matrix where the majority of its elements are set to zero. Thus, the solver assembles the matrix, in a row-by-row manner, and at each step, the assembled row is stored in sparse format to save the memory. Sparse direct solvers involve much more complicated algorithms than solvers suited for dense systems. This class of solvers is used to solve the matrix equation $Ax = b$, where the coefficient matrix A is considered to be large and sparse. The main challenge in these types of solvers is the efficient fill-in of the L and U factors of a sparse system. Typically, matrices in PEEC are dense, unsymmetric, indefinite and ill-conditioned. By sparsifying such matrices, the system becomes even closer to being singular and thereby would need numerical techniques, i.e. pivot perturbation or iterative refinement, to compensate for numerical instabilities. Later, by acquiring a valid sparse system, MUMPS will be utilized as an appropriate solver for this purpose [27]. For unsymmetric matrices, MUMPS first tries to symmetrize the matrix, based on the pattern $A + A^T$, and then reorder the matrix to minimize the cost of the factorization. This process, which is known as symbolic factorization, is necessary to determine the non-zero structure of the factors, before performing any numerical factorization. By having a matrix equation where the coefficient matrix is symmetric, the symbolic factorization can be performed as

$$\psi A \psi^T = LDU \tag{16}$$

where

- A is the sparse coefficient matrix;
- ψ is the permutation matrix that reorders A;

- L and U are triangular matrices, parts of the factorized A;
- D is a diagonal matrix, a part of the factorized A.

Using the MUMPS package, various methods for reordering are available, e.g. Approximate Minimum Degree (AMD) [29] and METIS [30]. The reordering can also be performed by providing the solver with a permutation matrix ψ. Using a sparse direct solver, the solution is carried out in three main steps:

1. *Analysis*: The symbolic factorization, which involves reordering, is performed on the symmetric pattern of the coefficient matrix. Permutation applies on the row and columns of the original matrix through a permutation matrix which consists of a set of orthogonal reordering vectors.
2. *Factorization*: By having symbolic data, the numerical factorization is performed using a numerical pivoting method. In the case of detecting zero pivots, perturbation will be performed by the sparse solver. The perturbations can affect the accuracy of the results. The accuracy will be retained by iterative refinement steps after the solution is done.
3. *Solution*: The factorized system is solved in this phase, using backward-forward substitution. As mentioned, some iterative refinement steps are also performed to correct the effect of possible perturbations.

Due to the structure of the sparse coefficient matrix in (R,K)PEEC formulation, many of the diagonal elements are set to zero, which can easily cause the final solution to become numerically unstable. Thus, certain algorithms known as *weighted matching* and *scaling* are used by the solver, to increase the accuracy of the pivoting [31]. MUMPS offers several remedies, and it is very important to choose a proper algorithm for this purpose. During several experiments, it was observed that when no scaling was applied, a few unwanted spikes could appear in the solution. On the other hand, when the scaling algorithms were applied, the spikes didn't appear and accurate results were acquired. Figure 4 depicts the effect of using scaling to solve a linear equation. It is observed that spikes do not appear in the final solution when a scaling feature with more accurate pivoting is enabled.

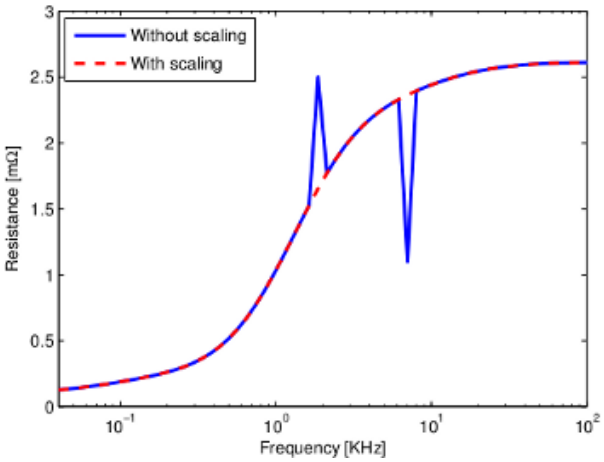


Fig. 4. The effect of scaling on the accuracy of pivoting during the factorization of an ill-conditioned sparse coefficient matrix.

B. GPU Acceleration

According to (10) and (11), grouping the structure along each axis can improve the performance of the reluctance calculation up to a factor of 9. Therefore, massively parallel solutions can be used to speed up this process. In the implemented PEEC-based solver, each block of the reluctance matrix is calculated in the following steps:

1. Calculate each block of the partial inductance matrix;
2. Transfer the calculated block from the host to the GPU device;
3. Invert the transferred block on the GPU device;
4. Transfer the inverted block back to the host;
5. Sparsify the block, and store it in sparse format;
6. Release the memory for that block.

Because the solver uses complex double precision data to solve problems in the frequency domain, NVIDIA Fermibased GPUs are exploited in this research. NVIDIA Fermibased M2050

Tesla series graphic cards offer 448 CUDA cores with 3 GB internal memory and have the computational power of 515 GFLOPS which makes them appropriate for expensive mathematical calculations [32].

The inversion using the GPU is performed by calling a CULA-appropriate routine [26]. Then, the routine will transfer the whole block of data to the graphic card's internal memory, to minimize the communication overhead, and will invert it using the massively parallel structure of the graphic card. When the inversion has been completed, the solution is transferred back to the main memory of the host system. Since the internal memory of available GPUs is commonly less than the available memory on the modern machines, thus the problem size is limited to the size of the internal memory on the graphic cards. In the future, with more powerful GPUs with larger memory and advances in the software tools which could partition the data into blocks which could be solved separately on GPUs, it would be possible to perform the whole solution solely on GPUs. CULA routines use some reserved memory for internal usage, where this workspace is allocated on both the GPU memory and the main memory [32]. The process of inversion is actually performed by both the host CPU and all processing cores on the GPU. However, the computations are mostly handled by the cores available on the GPU.

Additionally, the sparse solver has been compiled to use BLAS [33] operations on the GPU. Hence, all BLAS operations of the solver are performed by the GPU to reach the highest level of parallelism.

V. RESULTS

In this section, the performance of the solver will be studied using a PEEC model of a interconnection bus bar as a part of a power frequency converter. The analyzed model consists of a planar DC-link bus bar as typically used in multi-level medium voltage frequency converters [34]. The purpose of the DC-link in a multi-level frequency converter is to store the energy between the front-end rectifier and back-end inverter units, and the DC-link must therefore be designed for low stray inductance to limit the overvoltage peaks when switching high currents. Furthermore, the requirements and complexity of the bus bar strongly depend on the circuit topology used for power conversion [35]. The simulation model is depicted in Fig. 5.

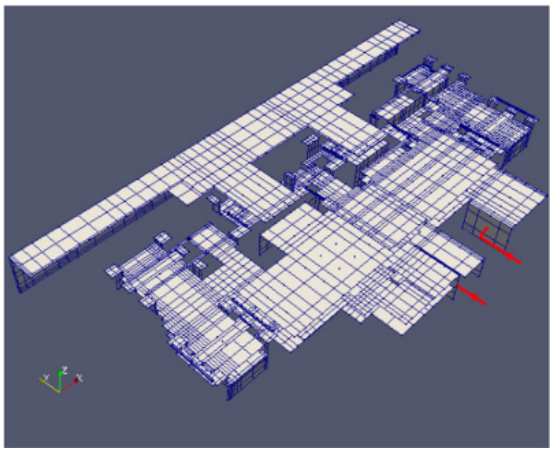


Fig. 5. PEEC model of the studied bus bars.

In the modeled bus bars, a current source with the amplitude of 1A is connected between two ports, which are marked two arrows which point to the connection ports. The simulation has been performed from 1 Hz to 10 MHz using 10 frequency steps. The total resistance and inductance of the bars has been extracted from the simulation. Two different test cases of the same model with different mesh densities have been analyzed. In these tests, the dense solver utilizes the Intel MKL library, while the sparse solver uses the reluctance method together with the sparse direct solver MUMPS. All simulations have been run on a Linux 64-bit cluster, equipped with two quad-core Intel Xeon E5520 2.2 GHz processors with 24 GB of RAM installed and a Tesla M2050 card with 3 GB of on-board memory.

Table 1: Bus bar solution

Model	Solver	Num. of unknowns	Mem. [GB]	Sparse [%]	Time [hh:mm:ss]
BB1	Dense	33 296	17.7	0	06:34:10
	Sparse		13.3	0	01:56:50
	Sparse		0.8	95	01:00:22
	Sparse		0.5	98	00:40:00
BB2	Dense	58 668	55	0	-*
	Sparse		41.5	0	-*
	Sparse		6.6	95	10:12:02
	Sparse		2.8	98	08:10:28

-*: Not available due to memory

Table 1 shows the simulation results of each test case using different levels of sparsification. Figures 6 and 7 present the total resistance and inductance of the bus bars as the frequency increases. It should be noted that for the first test case, the results for the case which the reluctance matrix is sparsified up to 95% almost overlaps the results without any sparsification. Moreover, results from the second test model reveal that as the problem size increases, the error rate decreases when the reluctance matrix gets sparser. Figures 6 and 7 also demonstrates the skin and proximity effects along the bus bars, where, as the frequency increases resistance increases and inductance decreases to a certain level.

From Figs. 6 and 7 and Table 1, several conclusions can be drawn:

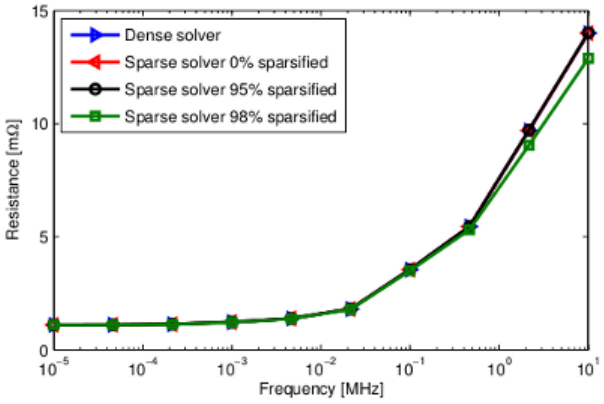
- The highest error in the results occurs when the sparsification is up to 98%. However, this error is still adequate and less than 8%.
- Although the reluctance matrix is sparsified up to 98%, the error is still quite low. Moreover, despite this acceptable error in the results, for the smallest model, the speed and memory usage has been improved by factors of approximately 10 and 35, respectively. It is also evident that problems that could not be solved before, due to the lack of the memory (i.e. BB2), can now be solved with small approximations involved in the solution.
- For reluctance calculations, the GPU solution results in a remarkable speedup, compared to the CPU-only solution. The speedup increases even more as the problem size increases. For the first test case, reluctance calculation took 110 and 57 seconds for CPU and GPU respectively and the second test case, reluctance calculation was carried out in 437 and 120 seconds for CPU and GPU respectively. As stated before, all matrix operations have also been done on GPU which improved the overall solution. It is expected that higher speedup will be achieved when graphic cards with higher internal memories are manufactured. Furthermore, at each time, only one block of the inductance matrix shown in Fig. 3 is transferred to the GPU. Thus, the memory peak is always decreased to the largest block in the inductance matrix.
- In Table 1, the dense solver indicates the solver which uses only CPU resources while the sparse solver uses CPU together with GPU resources, since it performs all BLAS operations on GPU. Comparison between dense and sparse solvers with no sparsity, reveals that the GPU have contributed to gain a speedup around 3.4 to solve exactly the same problem.
- Smaller problems loose the accuracy faster than larger problems when the reluctance matrix is sparsified in the same level. Comparison between two analyzed structures, yields that BB1 showed 8% error when the reluctance matrix is 98% sparsieifed while BB2 had only 3% error with the same level of sparsification, comparing to the cases when 95% sparsification is applied.
- The parallel sparse direct solver shows better performance than the dense solver even when the reluctance matrix is not sparsified at all. This performance increase can be due to the nature of the coefficient matrix in the MNA formulation in the PEEC, which is relatively sparse, and the efficiency of the sparse solver for sparse systems.

VI. CONCLUSION AND FURTHER WORK

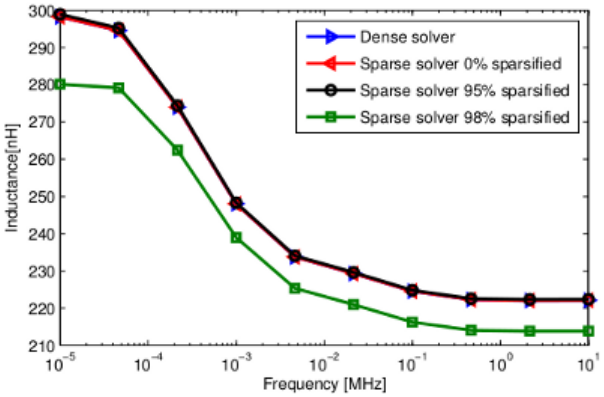
The improved solver presented in this paper makes it possible to solve very large problems with limited computational resources. Using the reluctance method, the final coefficient matrix in the system equation can be sparsified without the risk of an unstable solution, due to the diagonal dominance of the reluctance matrix. In the new solver, the bottleneck of the solution is shifted to the reluctance matrix calculation. Therefore, utilizing GPU technology together with grouping algorithms is the key to improving the performance in this phase of the solution. A numerical test case was studied which proved that even by sparsifying the reluctance matrix up to a high level (i.e. 98%) the required accuracy was still satisfied. In addition, memory usage and the grouping algorithms were analyzed, showing that, using grouping strategy, the reluctance matrix can be computed by inverting sub-matrices, which is less expensive and more efficient than inverting the whole L_p matrix. It was also observed that, even when no sparsification is applied to the system, the direct sparse solver performs more efficiently than conventional dense solvers. Because of the relatively sparse matrices involved in MNA formulations, which make sparse solvers more appropriate for this purpose. For the next step, the iterative solvers can be considered for use with the PEEC-based solver. Using iterative solvers, the time complexity of the solution can be reduced, but because of the severely ill conditioned matrices that are involved in the PEEC method, the development of a proper preconditioner will be a challenging and critical task. Finally, the iterative PEEC-based solver can be enhanced to employ GPU hardware. The workload of preconditioning and Krylov subspace solver can be shared between the host and GPU to acquire maximum efficiency and the parallelism.

ACKNOWLEDGMENT

The authors would like to thank ABB for providing bus bars models for this work.

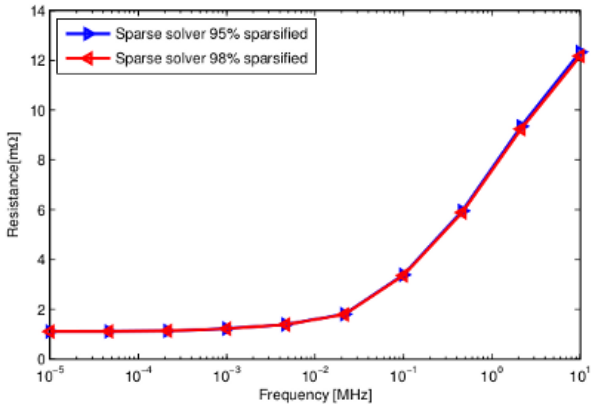


(a)

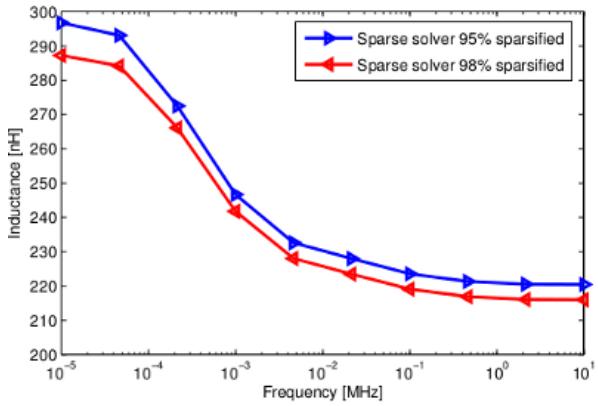


(b)

Fig. 6. Bus bar resistance (a) and inductance (b) of the BB1 model, simulated using dense and sparse solvers with different sparsification levels.



(a)



(b)

Fig. 7. Bus bar resistance (a) and inductance (b) of the BB2 model, simulated using dense and sparse solvers with different sparsification levels.

REFERENCES

[1] A. Nishizawa, M. Shimazaki, and S. Tanabe. "Ground Bouncing Analysis Using a Program Linking FDTD and SPICE," *In Proc. of Int. Symposium on Elctromagnetic Compatibility*, Seattle, WA, USA, 1999.

[2] A. E. Ruehli. "Equivalent Circuit Models for Three Dimensional Multiconductor Systems," *IEEE Transactions on Microwave Theory and Techniques*, MTT-22, no. 3, pp. 216-221, March 1974.

[3] A. E. Ruehli. "Inductance Calculations in a Complex Integrated Circuit Environment," *IBM Journal of Research and Development*, vol. 16, no. 5, pp. 470-481, September 1972.

[4] A. E. Ruehli, P. A. Brennan. "Efficient Capacitance Calculations for Three Dimensional Multiconductor Systems," *IEEE Transactions on Microwave Theory and Techniques*, MTT-vol. 21 no. 2 pp. 76-82, February 1973.

[5] Z. Song, F. Duval, D. Su, and A. Louis. "Stable Partial Inductance Calculation for Partial Element Equivalent Circuit Modeling," *Applied Computational Electromagnetics Society (ACES) Journal*, vol. 25, no. 9, 2010.

[6] G. Antonini, G. Miscione, and J. Ekman. "PEEC Modeling of Automotive Electromagnetic Problems," *Applied Computational Electromagnetics Society (ACES) Newsletter*, vol. 23, no. 1, pp. 39-50, March 2008.

[7] D. Daroui and J. Ekman. "Parallel Implementation of the PEEC Method," *Applied Computational Electromagnetics Society (ACES) Journal*, vol. 25, no. 5, pp. 410-422, May 2010.

[8] D. Daroui and J. Ekman. "Performance Analysis of Parallel Nonorthogonal PEEC-based Solver for EMC Applications," *Progress In Electromagnetics Research B*, vol. 41, pp. 77-100, 2012.

[9] H. Ji, A. Devgan, and W. Dai. "KSPICE: Efficient and Stable RKC Simulation for Capturing On-chip Inductance Effect," *Technical Report*, University of California, April 2000.

[10] Y. B. Tao, H. Lin, and H. J. Bao. "From CPU to GPU: GPU-based Electromagnetic Computing (GPUECO)," *Progress In Electromagnetic Research*, vol. 8, no. 1, pp. 1-19, 2008.

[11] M. J. Inman and A. Z. Elsherbeni. "GPU Acceleration of Linear Systems for Computational Electromagnetic Simulations," *In Proc. of the IEEE Int. Symposium on Antennas and Propagation Society*, Charleston, SC, USA, 2009.

[12] M. Ujaldon. "Using GPUs for Accelerating Electromagnetic Simulations," *Applied Computational Electromagnetics Society (ACES) Journal*, vol. 25, no. 4, 2010.

[13] M. J. Inman and A. Z. Elsherbeni. "FDTD Calculations using Graphical Processing Units," *In Proc. of the IEEE Conf. on Wireless Communications and Applied Computational Electromagnetics*, HI, USA, 2005.

[14] V. Demir and A. Z. Elsherbeni. "Compute Unified Device Architecture (CUDA) Based Finite-Difference Time-domain (FDTD) Implementation," *Applied Computational Electromagnetics Society (ACES) Journal*, vol. 25, no. 4, pp. 303-314, April 2010.

[15] T. Topa, A. Karwowski, and A. Noga. "Using GPU with CUDA to Accelerate MoM-based Electromagnetic Simulation of Wire-grid Models," *IEEE Antennas and Wireless Propagation Letters*, vol. 10, pp. 324-345, 2011.

[16] S. Ramo, J. R. Whinnery and T. Van Duzer, Fields and Waves in Communication Electronics, John Wiley and Sons, 1994.

[17] A.E. Ruehli, G. Antonini, J. Esch, A. Mayo J. Ekman, and A. Orlandi. "Non-orthogonal PEEC Formulation for Time and Frequency Domain EM and Circuit Modeling," *IEEE Transactions on Electromagnetic Compatibility*, vol. 45, no. 2, pp. 167-176, May 2003.

[18] C. Ho, A. Ruehli, P. Brennan. "The Modified Nodal Approach to Network Analysis," *IEEE Transactions on Circuits and Systems*, pp. 504-509, June 1975.

[19] G. Antonini, J. Ekman, and A. Orlandi. "Full Wave Time Domain PEEC Formulation using a Modified Nodal Analysis Approach," *In Proc. of EMC Europe*, Eindhoven, The Netherlands, 2004.

[20] T.-H. Chen, C. Luk, H. Kim, C C.P. Chen. "Inductwise: Inductance-wise Interconnect Simulator and Extractor," *In Proc. of the IEEE Int. Conf. on Computer Aided Design*, pp. 215-220, San Jose, CA, Nov 2002.

[21] H. Ji, A. Devgan, and W. Dai. "Ksim: A Stable and Efficient RKC Simulator for Capturing On-chip Inductance Effect," *In Proc. of Design Automation Conference*, CA, USA, 2001.

[22] C. Luk. "Efficient Inductance Extraction for On-chip Interconnect," *Technical report*, University of Wisconsin-Madison, 2003.

[23] S. Zenga, W. Yu, J. Shi X, Hong, and C. Cheng. "Efficient Partial Reluctance Extraction for Large-scale Regular Power Grid Structures," *IEICE Transactions on Fundamentals*, E92-A, no. 6, June 2009.

[24] G. Zhong, C. Koh, V. Balakrishnan, and K. Roy, "An Adaptive Window-based Susceptance Extraction and its Efficient Implementation," *In Proc. of Design Automation Conference*, IN, USA, 2003.

[25] Intel Math Kernel Library (MKL). Online: http://software.intel.com/en-us/articles/intel-mkl/.

[26] CULA A set of GPU-accelerated linear algebra libraries. Online: http://www.culatools.com/.

[27] MUMPS A parallel sparse direct solver. Online: http://graal.ens-lyon.fr/MUMPS/.

[28] M. L. Zitzmann. "Fast and Efficient Methods for Circuit-based Automotive EMC Simulation", *PhD thesis*, University of Erlangen-Nürnberg, February 2007.

[29] P. R. Amestoy, T. A. Davis, and I. S. Duff. "An Approximate Minimum Degree Ordering Algorithm," *SIAM Journal on Matrix Analysis and Applications*, vol. 17, pp. 886-905, 1996.

[30] METIS A Software Package for Partitioning Unstructured Graphs, Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices. Online:http://glaros.dtc.umn.edu/gkhome/views/metis/.

[31] Iain S. Duff. "Developments in Matching and Scaling Algorithms," *Proc. of Appl. Math. Mech.*, vol. 7, no. 1, Aug 2007.

[32] The Tesla M class GPU modules. Online: http://www.nvidia.com/object/preconfigured-clusters.html.

[33] K. C. Huang, P. C. Wu, and F. J. Wang. "Parallelizing a Level 3 BLAS Library for LAN-Connected Workstations," *In Proc. of Second Symposium on Autonomous Decentralized Systems*, Arizona, USA, April 1995.

[34] S. Van Acker, P. Salenbien, and A. Cosaert. "Predictability of the Behavior of Power Distribution Components in Power Conversion Applications," *In PCIM Conference*, Shanghai, China, March 2005.

[35] A. Rodríguez, J. S. Lai, and F. Z. Peng. "Multilevel Inverters: A Survey of Topologies Controls, and Applications," *IEEE Transactions on Industrial Electronics*, vol. 49, no. 4, August 2002.