Parallel Implementations of the PEEC Method

Jonas Ekman and Peter Anttu

EISLAB

Department of Computer Science & Electrical Engineering

Luleå University of Technology

SE-971 87 Luleå, Sweden

Abstract—This paper presents a parallel implementation of a partial element equivalent circuit (PEEC) based electromagnetic modeling code. The parallelization is based on the GMM++ and ScaLAPACK packages. The parallel PEEC solver was successfully implemented and tested on several high performance computer systems. Large structures containing over 50 000 unknown current and voltage basis functions were successfully analyzed and memory, performance, and speedup results are presented. The numerical examples are both of orthogonal and nonorthogonal type with analysis in the time- and frequency-domain.

I. INTRODUCTION

The partial element equivalent circuit (PEEC) method [1], [2] is widely used for solving mixed circuit and electromagnetic problems. The method gives a framework for creating electric equivalent circuit representations for three-dimensional structures and calculating self and mutual inductances and capacitances. The resulting equivalent circuits can be solved in SPICE-like solvers or, for the full-wave case, by creating and solving the fully coupled circuit equations.

As for all the methods within computational electromagnetics (CEM), the problem system size that can be solved is increasing with more efficient computer implementations [3] and more powerful computer systems. However, the desired problem sizes to be solved are also increasing and there is a clear gap between desired and possible problem size to be solved. Fast solutions for CEM problems have been treated for a long time, i.e. [4] where both differential and integral equation solvers were discussed. The next step after faster implementations are to improve the computing power running the algorithms. One solution is to use GRID computing on different levels. For example using a local area network of interconnected computers to speed up calculations or by porting the code to parallel architectures. Recent publications on the extension to parallel implementations are for example [5] where a nesting combination of the finite element domain decomposition method and the algebraic multigrid method is presented, [6] on the implicit FDTD method, and [7] for a parallel version of the numerical electromagnetics code (NEC).

Until now, no parallel implementation on the PEEC method have been reported except for in [8] where a sequential code was parallelized for LANs using a freeware. In this paper, parallel implementations of the PEEC method is presented for supercomputers featuring different architectures using ScaLAPACK [9]. Implementation issues and examples are presented for the time- and frequency- domain PEEC method. It is shown

that with this type of parallel PEEC solvers the problem sizes can be increased considerably and new application areas arise.

II. FORMULATION OF PEEC CIRCUIT EQUATION

The classical PEEC method is derived from the equation for the total electric field at a point [10] written as

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

where $\mathbf{E}^i$ is an incident electric field, $\mathbf{J}$ is a current density, $\mathbf{A}$ is the magnetic vector potential, and ϕ is the scalar electric potential, all at observation point $\mathbf{r}$. 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 Lp_{11} and Lp_{22} in Fig. 1 are defined as

$$Lp_{\alpha\beta} = \frac{\mu}{4\pi} \frac{1}{a_\alpha a_\beta} \int_{v_\alpha} \int_{v_\beta} \frac{1}{|\mathbf{r}_\alpha - \mathbf{r}_\beta|} dv_\alpha dv_\beta. \quad (2)$$

Figure 1 also shows the node capacitances which are relates to the diagonal 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

$$p_{ij} = \frac{1}{S_i S_j} \frac{1}{4\pi\epsilon_0} \int_{S_i} \int_{S_j} \frac{1}{|\mathbf{r}_i - \mathbf{r}_j|} dS_j dS_i \quad (3)$$

and a resistive term between the nodes, defined as

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

In eq. (2) and eq. (4), a represents the cross section of the rectangular volume cell normal to the current direction, γ , and l is the length in the current direction. Further, v represents the current volume cells and S the charge surface cells. For a detailed derivation of the method, including the nonorthogonal formulation, see [11].

The discretization process of the EFIE in eq. (1) and the successive Galerkin's weighting leads to generate an equivalent circuit. When Kirchhoff's voltage and current laws are

Fig. 1. Metal strip with 3 nodes and 2 cells (a) and 3 node PEEC circuit (b)

enforced to the N_i independent loops and N_ϕ independent nodes of the PEEC equivalent circuit we obtain:

$$-\mathbf{A}\dot{\Phi}(t) - \mathbf{R}\mathbf{i}_L(t) - \mathbf{L}_p\dot{\mathbf{i}}_L(t) = \mathbf{v}_s(t) \quad (5a)$$

$$\mathbf{P}^{-1}\dot{\Phi}(t) - \mathbf{A}^T\mathbf{i}_L(t) = \mathbf{i}_s(t) \quad (5b)$$

where

- $\Phi(t) \in \mathbb{R}^{N_\phi}$ is the vector of node potentials to infinity; $\mathbb{R}^{N_\phi}$ is the node space of the equivalent network;
- $\mathbf{i}_L(t) \in \mathbb{R}^{N_i}$ is the vector of currents including both conduction and displacement currents; $\mathbb{R}^{N_i}$ is the current space of the equivalent network;
- $\mathbf{L}_p$ is the matrix of partial inductances describing the magnetic field coupling;
- $\mathbf{P}$ is the matrix of coefficients of potential describing the electric field coupling;
- $\mathbf{R}$ is the matrix of resistances;
- $\mathbf{A}$ is the connectivity matrix;
- $\mathbf{v}_s(t)$ is the vector of distributed voltage sources due to external electromagnetic fields or lumped voltage sources;
- $\mathbf{i}_s(t)$ is the vector of lumped current sources.

A program for electromagnetic analysis, based on the theory and references outlined above, has been developed [12]. The solver can handle both the traditional orthogonal PEEC model and the newly introduced nonorthogonal formulation [11]. The solver creates an equivalent circuit and calculates the corresponding resistances, (partial) inductances, capacitances, and coupled voltage and current sources (to account for electromagnetic couplings) for the given geometrical layout (CAD-data as specified in an input file). The user adds external electronic (sub-)systems and analysis mode as described by the SPICE syntax. The actual solution of the resulting circuit equations, eqs. (5a) and (5b), in either the time- or frequency-domain, is performed in the solver and results are given as current- and voltage- distributions in the geometrical layout.

III. PARALLELIZATION OF THE PEEC SOLVER

A. Introduction

The development and performance-testing of the parallel code has mainly been performed on the Ingrid cluster at HPC2N [13]. This Linux cluster consists of 100 nodes with Intel Pentium4 CPUs running at 2.8GHz giving each CPU a peak floating point performance of 5.6 Gflops/s. Each node is equipped with 2 GB of memory and fast ethernet. As the nodes only have fast ethernet the efficiency of ScaLAPACK is put to test.

The choice to use ScaLAPACK [9] was based on previous experience and the fact that it is aimed at the solution of dense matrix equations. In addition it is a stable, well tested, and efficient package. ScaLAPACK uses block cyclic data distribution to achieving good load balancing. This means that matrices are divided into blocks in two dimensions and these blocks are assigned to a set of processors. The parallel solver follows these steps:

- The discretization is entirely serial and duplicated on all processors.
- The partial element calculations are easily parallelized as no communication is required between nodes.
- The main difficulty lies in the mapping between global and local matrix coordinates. An overview of how to do this is given in [7].
- The matrix formulation and solution parts are performed by ScaLAPACK.

The new parallelized solvers uses the direct LU factorization approach of solving the linear equation systems. In addition Cholesky factorization is used where possible.

B. Parallelization of partial element computations

The partial element calculations are easily parallelized as is seen in Table I. The table is collected from frequency domain simulations in which the full matrices are calculated and therefore the time consumption is approximately doubled between the serial and one processor parallel version. There is also a cost associated with the conversion between global and local matrix coordinates. The test case "Reactor" is a orthogonal PEEC model utilizing analytical routines to evaluate partial inductances ($< 1\mu s/\text{element}$). The "Chassis" is a nonorthogonal PEEC model for which partial inductances are evaluated using numerical integration. In the tabulated case "8th GI" means 8th order Gauss-Legendre numerical integration and "FMF" means fast multi function (FMF) approach [14], [15]. The examples are described more in detail in Sec. IV.

The partial element calculation times for the time domain solver are actually much closer to the serial, as there is no duplication of effort. Only the upper (or lower) part of the matrices need to be calculated as this is sufficient for the Cholesky factorization mentioned earlier. Due to the properties of the matrices in the full-wave, frequency domain case, only an LU factorization is possible and therefore the need to fill in the whole matrix.

TABLE I
PARTIAL ELEMENT CALCULATION TIMES.

Numbers of processors	Partial inductance calculation	Chassis [s]		
		Reactor	8 th GI	FMF
Serial		6	12200	2480
1		13	28100	5460
4		3	7120	1860
16		1	1740	417

C. Increasing the number of processors

The performance gain of increasing the number of processors can be studied by using the speedup factor

$$S(n) = \frac{t_s}{t_{p_n}} \quad (6)$$

where t_s is the time taken by the serial code and t_{p_n} is the time taken by the parallel code using n processors. Various test results are shown in Fig. 2. In the figure, the following abbreviations are used: frequency domain (FD), time domain (TD), quasi-static (QS), and full-wave (FW). Further, "ndiv" is an easy way to increase the number of current and voltage basis functions as shown in Table II.

Fig. 2. Speedup factor $S(n)$ for various numbers of processors n for a few different test cases.

Ideally the performance increase would be a linear function of the number of processors. That is, $S(n) = kn$, where k should be as close to 1 as possible. This is not the case, but from the figure it can be seen that as the problem size is increased, the speedup factor increases. This is due to the fact that as processors have more to calculate, computation and communication can be more overlapped [16].

If instead the number of processors are fixed at 36 and the problem is scaled by increasing the discretization, the results are as in Table III. As the time taken by matrix formulation and time per sample have a time complexity of $O(N^3)$ the lower increase in time between the different test cases suggests the better use of the computational power.

TABLE II
REACTOR CHARACTERISTICS.

ndiv	Number of		nodes
	surface cells (charge basis functions)	volume cells (current basis functions)	
1	1604	802	1004
2	2404	1602	1804
4	4004	3202	3404
8	7204	6402	6604
16	13604	12802	13004
32	26404	25602	25804

TABLE III
FREQUENCY DOMAIN, QUASI-STATIC SIMULATION OF REACTOR. NUMBER OF PROCESSORS FIXED AT 36. PROBLEM SIZE INCREASED BY USING NDIV.

Number of unknowns ($\propto$ ndiv from Tab. II)	Matrix formulation [s]	Matrix solution [s]
2 406	5	3
4 006	9	9
7 206	25	27
13 606	101	113
26 406	524	558
52 006	3120	3440

D. Memory usage

In this section, the memory usage of the parallel solver is studied based on the following aspects:

- memory usage as a function of number of processors;
- memory usage as a function of problem size.

1) *Memory usage as a function of the number of processors:* The total memory usage M should be a function of the number of processors, p , and the problem size. This can be expressed as

$$M = ap + b. \quad (7)$$

Fig. 3 shows the total memory usage M as a function of processors. The increase is clearly linear and eq. (7) is verified.

Another way of expressing this is as average memory usage per processor m as

$$m = b/p + c. \quad (8)$$

This behavior for the parallel PEEC solver is verified by Table IV.

TABLE IV
MEMORY USAGE FOR VARYING NUMBER OF PROCESSORS. SIMULATION OF REACTOR.

Numbers of processors	Average memory usage / processor [MB]			
	Time domain	Frequency domain		
	Quasi-static ndiv=8	Quasi-static ndiv=4	Quasi-static ndiv=4	Full-wave ndiv=4
1	1920	510	1010	1540
4	500	113	264	399
9	228	56	122	183
16	131	34	73	107
25	88	24	49	71
36	63	21	37	52

Fig. 3. Memory usage as a function of processors.

2) *Memory usage as a function of problem size:* The expected memory usage is

$$m = bn^2 + cn + d \quad (9)$$

This is a simplification as in reality the actual increase is not as straightforward due to the number of surface and volume cells and nodes vary and the memory usage will be a more complicated function of these variables. Studying Table V and letting (m = memory usage) and (n = ndiv) in eq. (9) it is seen that the increase for time domain, $n = 16$ to $n = 32$ is slightly under 4. The same reasoning applies to the frequency domain simulation as well.

The expected memory usage can also be calculated with the help of the formulae in [16] for calculating the upper bounds for local rows and columns of matrices. Doing so for the frequency domain, quasi-static case when $ndiv = 32$ yields the following:

6 matrices are live at the same time. These are all approximately $25K \times 25K$ elements in size. Using the formulae gives local matrix of 4224×4224 . Multiplied by 16 for complex and 36 for number of processors gives the upper bound of 57 GB.

The actual memory usage is 60 GB. This amounts to an excess of 85 MB per processor.

TABLE V
AVERAGE MEMORY USAGE FOR INCREASING PROBLEM SIZE. 36 PROCESSORS. SIMULATION OF REACTOR.

ndiv	Average memory usage / processor [MB]		
	Time domain Quasi-static	Frequency domain	
		Quasi-static	Full-wave
1	7	8	12
2	9	15	20
4	21	37	52
8	63	121	167
16	227	444	606
32	865	1720	NA

IV. NUMERICAL TESTS

The initial testing of the parallel PEEC implementation have been completed in this section with two test examples. These examples have been analyzed with both the original serial PEEC solver, when possible, and the new parallel PEEC solver. The first example is a air-core reactor that has been used in the previous section for the general performance analysis. The second example is a nonorthogonal car chassis (body-in-white) example.

A. Air-core reactor

The reactor example is a part of a research project in which high frequency models for air-core reactors are being developed [17]. The test example is of rectangular type with four sides equal in length = 0.5 m. The windings (turns) are totally 65 and consist of copper tape with dimension 0.076 mm x 6.35 mm. The center to center spacing between the turns is 10 mm. This example can be studied using the original (orthogonal) formulation of the PEEC method and is therefore fast in the calculation of the partial elements. This can be seen in Table I where the partial inductances are calculated in 6.5 seconds in the serial implementation and in 3.38 seconds when using 4 processors for the parallel implementation.

In previous papers, i.e. [18], [19], the reactors have been studied in the time- and frequency- domain with regular (L_p, P, R, τ) PEEC models. However, the inclusion of Skin and proximity effects have not been possible through the volume filament approach, (VFI) PEEC, due to the excessive number of unknowns. The parallel implementation of the solver can now treat this type of problem and give a more correct model for the current distribution in the conductors. This is exemplified in Fig. 4 where a small section of the reactor is shown and arrows are giving the current distribution (and colors are giving the voltage distribution).

The second numerical test for the reactor is performed to study the frequency domain behavior when one end is

Fig. 4. Skin and proximity effects in reactor windings impacting on current distribution as shown with arrows. Voltage distributions shown with colors.

Fig. 5. Voltage distribution in air-core reactor at 4 MHz.

open (open-ended-test) and the other one is exciting the structure. When using a frequency sweep, the different voltage distributions in the windings, for different frequencies, can be plotted as in Fig. 5. This test combined with the (VFI)PEEC-model, gives a very accurate electromagnetic model for the reactor structure and is only possible to study with the parallel PEEC implementation.

B. Body-in-white

The second numerical test deals with a complex, nonorthogonal PEEC model for a chassis, also called body-in-white. The basic nonorthogonal PEEC model for the chassis structure is meshed (using quadrilateral patches) into 1 272 volume cells, 1 272 surface cells, and 359 (unique) nodes. This basic configuration results in 1 617 984 mutual inductances, 1 617 984 mutual coefficients of potential (capacitances).

This basic example can be studied using the sequential code using the nonorthogonal formulation of the PEEC method. However, the calculation of the partial elements are lengthy, see Table I for Chassis-8th GI. This type of partial element calculations (nonorthogonal) are facilitated by the FMF-approach [14], [15] as seen in the same table. When studying the chassis for higher frequencies, the mesh needs to be improved and the number of unknowns are exceeding the upper limit for the sequential implementation and computer system. The two subsequent test shows two model situations using a refined mesh in both the time- and frequency- domain. The refined mesh contain 4 276 volume cells, 3 114 surface cells, and 2 334 (unique) nodes and fulfils the condition $\frac{20 \text{ cells}}{\lambda_{min}}$ for 100 MHz.

1) *Time domain analysis:* The first test is carried out in the time domain with the chassis excited by a fast current pulse. The current source is a pulse waveform of Gaussian-type, $I(t) = e^{-x \cdot x}$ where $x = \frac{t - (150e-9)}{(50e-9)}$, injected in the front of the chassis. The back of the chassis is grounded using a 50

Fig. 6. Voltage distribution at 150 ns.

Ω resistor in order to have a current path through the chassis. This results the potential distribution in the chassis at time 150 ns seen in Fig. 6. The figure shows the higher voltages in the front of the chassis where the current is injected and the current distribution is as expected, flowing from the front end to the back end of the chassis.

2) *Frequency domain analysis:* The only addition for the frequency domain test is a 100 Ω resistor in parallel with the input current source (unitary for AC analysis). The result is given as the potential distribution in the chassis at 50 MHz in Fig. 7 (a) and for 72 MHz in (b).

V. CONCLUSIONS AND DISCUSSION

The presented parallel PEEC implementation open new doors for the solution of larger systems and the use of non-orthogonal partial element calculation routines. The calculation routines in particular are well parallelized and scale well, although there is room for improvement in efficiency. The solvers on the other hand can be much more efficient, but as long as problem size is increased as the number of processors are increased, performance is acceptable. Further, the testing platform was not the fastest available in terms of communication speed.

One part that improved the initial sequential code was the implementation of the routine for parallel-sparse-matrix handling (calculating $P^T A P$, where P is a sparse matrix and A is a dense matrix). However, this is more complicated to implement in a parallel environment. As the sparse matrices involved (the node reduction matrix and the connectivity matrix) are really sparse with very few elements these could be duplicated on every computational node. Doing so and following the sequential implementation could yield an acceptable solution.

The symmetry of the partial inductance L_p and coefficient of potential P matrices is not taken advantage of. This may seem like less of an issue as the number of processors is increased, but if nonorthogonal cells are involved then the partial element calculations are likely to take quite some time.

There is much to be done in the area of performance tuning of the parallel code. The creation of the processor grid is

Fig. 7. Voltage distribution at 50 MHz (a) and at 72 MHz (b).

the most simple and tries to form a square processor grid if possible. The addition of the possibility to specify an arbitrary grid should be very easily accomplished and could be useful in some situations. The blocking factors of matrices is currently defined in an include file and the possibility to specify them at run-time could be added.

In addition, the parallel solvers can solve problems faster by increasing the number of processors assigned to the task, but the efficiency may not be good for too small problems.

REFERENCES

- [1] A. E. Ruehli. Inductance calculations in a complex integrated circuit environment. *IBM Journal of Research and Development*, 16(5):470–481, September 1972.
- [2] A. E. Ruehli, P. A. Brennan. Efficient capacitance calculations for three-dimensional multiconductor systems. *IEEE Transactions on Microwave Theory and Techniques*, MTT-21(2):76–82, February 1973.
- [3] G. Antonini. Fast Multipole Formulation for PEEC Frequency Domain Modeling. *Journal of Applied Computational Electromag. Society*, 17(3), November 2002.
- [4] W. C. Chew, J. M. Jin, C. C. Lu, E. Michielssen, and J. M. Song. Fast solution methods in electromagnetics. *IEEE Transactions on Antennas and Propagation*, 45(3):533–543, 1997.
- [5] Y. Liu and J. Yuan. A finite element domain decomposition combined with algebraic multigrid method for large-scale electromagnetic field computation. *IEEE Transactions on Magnetics*, 42(4):655–658, April 2006.
- [6] T. Hanawa, M. Kurosawa, and S. Ikuno. Investigation on 3-D implicit FDTD method for parallel processing. *IEEE Transactions on Magnetics*, 41(5):1696–1699, May 2005.
- [7] A. Rubinstein, F. Rachidi, M. Rubinstein, and B. Reusser. A parallel implementation of NEC for the analysis of large structures. *IEEE Transactions on Electromagnetic Compatibility*, 45(2):177–188, May 2003.
- [8] F. Monsefi and J. Ekman. Optimization of peec based electromagnetic modeling code using grid computing. In *Proc. of EMC Europe*, Barcelona, Spain, 2006.
- [9] J. Choi, J. J. Dongarra, R. Pozo, and D. Walker. ScaLAPACK: A scalable linear algebra library for distributed memory concurrent computers. In *Proceedings of the Fourth Symposium on the Frontiers of Massively Parallel Computation (FRONTIERS '92)*, IEEE Computer Society Press, 1992.
- [10] S. Ramo, J. R. Whinnery and T. Van Duzer. *Fields and Waves in Communication Electronics*. John Wiley and Sons, 1994.
- [11] 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*, 45(2):167–176, May 2003.
- [12] LTU/UAQ PEEC solver. Available. Online: <http://www.csee.ltu.se/peec>.
- [13] HPC2N High-Performance Computing Center North Webpage. Available. Online: <http://www.hpc2n.umu.se>.
- [14] G. Antonini, J. Ekman, and A. Orlandi. Integral order selection rules for a full wave PEEC solver. In *Proc. of the Zürich Symposium on EMC*, pages 431–436, Zürich, Switzerland, 2003.
- [15] G. Antonini, J. Ekman, A. Ciccomancini Scogna, and A. E. Ruehli. A comparative study of PEEC circuit elements computation. In *Proc. of the IEEE International Symposium on EMC*, Istanbul, Turkey, 2003.
- [16] A. Cleary E. D'Azevedo J. Demmel I. Dhillon J. Don-garra S. Hammarling G. Henry A. Petitot K. Stanley D. Walker L. S. Blackford, J. Choi and R. C. Whaley. *ScaLAPACK Users' Guide*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 1997.
- [17] EISLAB Research Projects. Available. Online: <http://www.csee.ltu.se/peec>.
- [18] M. Enohnyaket and J. Ekman. High frequency models for air-core reactors using 3d equivalent circuit theory. In *Proc. of Nordic Distribution and Asset Management Conference (NORDAC'06)*, Stockholm, Sweden, 2006.
- [19] M. Enohnyaket and J. Ekman. Three dimensional high frequency models for air-core reactors based on partial element equivalent circuit theory. In *Proc. of EMC Europe*, Barcelona, Spain, 2006.