VERSITA
Electrical, Control and Communication Engineering
doi: 10.2478/ecce-2013-0011
2013 / 3

Finite Element Tearing and Interconnecting Method and its Algorithms for Parallel Solution of Magnetic Field Problems

Dániel Marcsa (*PhD Student, Széchenyi István University*),
Miklós Kuczmann (*Professor, Széchenyi István University*)

***Abstract* – Because of the exponential increase of computational resource requirement for numerical field simulations of more and more complex physical phenomena and more and more complex large problems in science and engineering practice, parallel processing appears to be an essential tool to handle the resulting large-scale numerical problems. One way of parallelization of sequential (single-processor) finite element simulations is the use of domain decomposition methods. Domain decomposition methods (DDMs) for parallel solution of linear systems of equations are based on the partitioning of the analyzed domain into sub-domains which are calculated in parallel while doing appropriate data exchange between those sub-domains. In this case, the non-overlapping domain decomposition method is the Lagrange multiplier based Finite Element Tearing and Interconnecting (FETI) method. This paper describes one direct solver and two parallel solution algorithms of FETI method. Finally, comparative numerical tests demonstrate the differences in the parallel running performance of the solvers of FETI method. We use a single-phase transformer and a three-phase induction motor as two-dimensional static magnetic field test problems to compare the solvers.**

***Keywords* – High performance computing, parallel processing, finite element analysis, magnetic fields.**

I. Introduction

The finite element method [1]-[2] is an important technique for the solution of a wide range of problems in science and engineering. In electromagnetic computation, it is based on the weak formulation of the partial differential equations, which can be obtained by Maxwell's equations and the weighted residual method [1]. The most time consuming part of finite element computations is the solution of the large sparse system of equations. Therefore, the solution of a large system of equations must be parallelized in order to speedup the numerical computations [3]-[5].

Different applications of domain decomposition method [3]-[4] have a long history in computational science. The reason for employing the sub-structuring technique was the small memory of computers. To solve large scale problems, domain has been divided into sub-domains that fit into the computer memory. Despite grow of computer memory, the demand for solution of large real life problems is always ahead of computer capabilities. The large scale computations and simulations performed with finite element method (FEM) [1]-[2] often require very long computation time. While limited progress can be reached with improvement of numerical algorithms, a radical time reduction can be made with multiprocessor computation. In order to perform finite element analysis by means of a computer with parallel processors, computations should be distributed across the processors (see in Fig 1.).

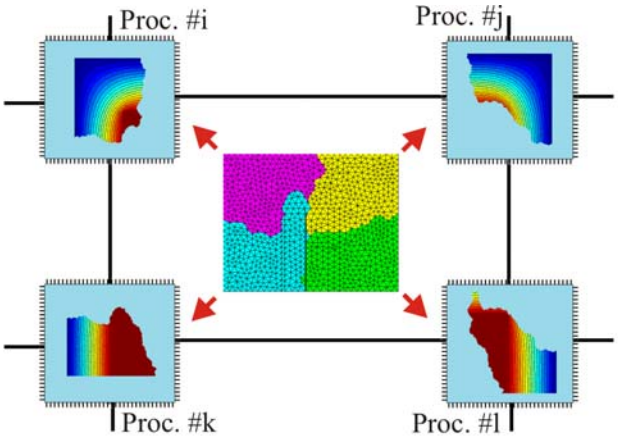

Fig. 1. The geometry based distributed computation.

The Finite Element Tearing and Interconnecting (FETI) method [3]-[6] was introduced by Farhat and Roux in Reference [4]. In the last decade, the FETI method [3]-[6] has seemed as one of the most powerful and the most popular solvers for numerical computation. The FETI requires fewer interprocessor communications, than traditional domain decomposition algorithm, as Schur or Schwarz method [3], while it still offers the same amount of parallelism. The best solvers for FETI method are iterative solvers. The conjugate gradient (CG) based iterative methods [7]-[8] have emerged as a widely used method. However, the direct solver sometimes is faster, more efficient and tends to be more robust than the iterative solver [6].

The parallel finite element based numerical analysis on supercomputers or on clusters of PCs (Personal Computers) needs the efficient partitioning of the finite element mesh. This is the first and the most important step of parallelization with the use of domain decomposition methods [9].

Many domain decomposition or graph-partitioning algorithms can be found in the literature [10]. Gmsh [11] combined with METIS algorithm [10] has been used for the discretization of the domain of problem and for the mesh

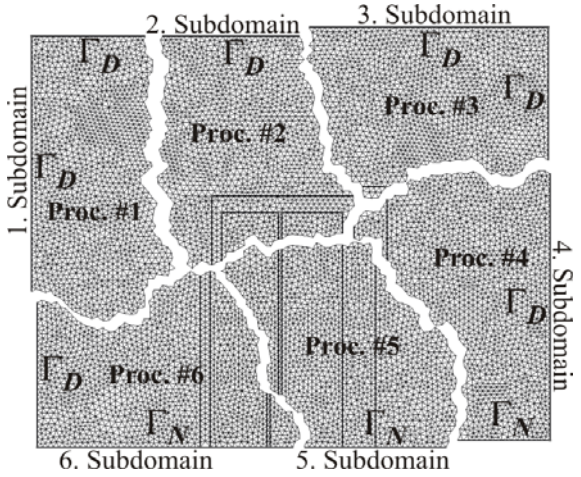

Fig. 2. Partitioned two-dimensional problem.

partitioning. The parallel finite element program has been implemented in the Matlab environment [12].

The paper presents a parallel approach to the solution of two-dimensional static magnetic field problems by parallel finite element method. These problems are case studies to show the steps of the Finite Element Tearing and Interconnecting method with finite element technique. The comparison focused on the time, speedup and numerical behavior of solvers of FETI method.

II. Finite Element Tearing and Interconnecting Method and its Algorithms

The main idea of the domain decomposition method is to divide the domain Ω into several sub-domains in which the unknown potentials could be calculated simultaneously, i.e. in parallel.

The general form of a linear algebraic problem arising from the discretization of a static field problem defined on the domain Ω can be written as [7]-[8]

$$\mathbf{K}\mathbf{a} = \mathbf{b}\,, \tag{1}$$

where $\mathbf{K} \in R^{n\times n}$ is the symmetric positive definite matrix, $\mathbf{b} \in R^n$ on the right hand side of the equations represents the excitation, and $\mathbf{a} \in R^n$ contains the unknown nodal potentials. Here n is the number of unknowns.

If domain Ω is partitioned into a set of N_S disconnected sub-domains (Fig. 2), the FETI method consists of replacing equation (1) with the equivalent system of sub-structure equations (where $j = 1, \ldots, N_S$) [3]-[6]

$$\mathbf{K}_j\mathbf{a}_j = \mathbf{b}_j - \mathbf{B}_j^{\mathrm{T}}\boldsymbol{\lambda}\,, \tag{2}$$

with the compatibility of the magnetic vector potentials at the sub-domain interface [3]-[6]

$$\sum_{j=1}^{N_S}\mathbf{B}_j\mathbf{a}_j = \mathbf{0}\,, \tag{3}$$

where $\mathbf{K}_j$ is the j^{th} sub-domain mass matrix, $\mathbf{b}_j$ is the j^{th} vectors of right-hand side, $\boldsymbol{\lambda}$ is a vector of Lagrange multipliers [3]-[4] introduced for enforcing the constraint (3) on the sub-domain interface boundary Γ_j, and $\mathbf{B}_j$ is a signed (±) Boolean mapping matrix [5], which is used to express the compatibility condition at sub-domain interface Γ_j. The superscript T denotes the transpose.

Usually, the partitioned problem may contain $N_f \le N_S$ floating sub-domain, where matrices $\mathbf{K}_j$ from being singular [9]. The floating sub-domain is a sub-domain without enough Dirichlet boundary conditions [6]. In Fig. 2, the Sub-domain 5 is a floating sub-domain, because the outer boundary is not Dirichlet boundary condition (Γ_D), but Neumann boundary condition (Γ_N). In this case N_f of local Neumann problems are singular. To guarantee the solvability of these sub-problems, we require that [4]-[5]

$$\left(\mathbf{b}_j - \mathbf{B}_j^{\mathrm{T}}\boldsymbol{\lambda}\right) \perp \mathrm{Ker}\left(\mathbf{K}_j\right), \tag{4}$$

and compute the solution of equation in (2) as [3]-[6]

$$\mathbf{a}_j = \mathbf{K}_j^{+}\left(\mathbf{b}_j - \mathbf{B}_j^{\mathrm{T}}\boldsymbol{\lambda}\right) + \mathbf{R}_j\boldsymbol{\alpha}_j\,, \tag{5}$$

where $\mathbf{K}_j^{+}$ is a pseudo-inverse of $\mathbf{K}_j$, $\mathbf{R}_j$=Ker($\mathbf{K}_j$) is the kernel or null space [7] of $\mathbf{K}_j$, and $\boldsymbol{\alpha}_j$ is the set of amplitudes that specifies the contribution of the null space $\mathbf{R}_j$ to the solution $\mathbf{a}_j$. Instead of a pseudo-inverse of matrix, the Moore-Penrose matrix inverse [7] or generalized inverse have been used here. The introduction of the $\boldsymbol{\alpha}_j$ is compensated by the additional equations resulting from (4) [3],[5]

$$\mathbf{R}_j^{\mathrm{T}}\left(\mathbf{b}_j - \mathbf{B}_j^{\mathrm{T}}\boldsymbol{\lambda}\right) = \mathbf{0}\,. \tag{6}$$

Substituting equation (5) into the equation (3) and exploiting the solvability condition (6), after some algebraic manipulations leads to the following FETI interface problem [3]-[6]

$$\begin{bmatrix}\mathbf{F}_I & -\mathbf{G}_I\\ -\mathbf{G}_I^{T} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{\lambda}\\ \boldsymbol{\alpha}\end{bmatrix} = \begin{bmatrix}\mathbf{d}\\ -\mathbf{e}\end{bmatrix}, \tag{7}$$

where [4]-[6]

$$\mathbf{F}_I = \sum_{j=1}^{N_S}\mathbf{B}_j\mathbf{K}_j\mathbf{B}_j^{\mathrm{T}}\,,$$

$$\mathbf{G}_I = \left[\mathbf{B}_1\mathbf{R}_1 \ldots \mathbf{B}_{N_S}\mathbf{R}_{N_S}\right],$$

$$\mathbf{d} = \sum_{j=1}^{N_S}\mathbf{B}_j\mathbf{K}_j^{\mathrm{T}}\mathbf{b}_j\,,$$

$$\mathbf{e} = \left[\mathbf{b}_1^{\mathrm{T}}\mathbf{R}_1 \ldots \mathbf{b}_{N_S}^{\mathrm{T}}\mathbf{R}_{N_S}\right]^{\mathrm{T}}.$$

In order to solve equation (7) for the Lagrange multiplier vector $\boldsymbol{\lambda}$, the following splitting of $\boldsymbol{\lambda}$ is performed [4]-[6]

$$\boldsymbol{\lambda} = \boldsymbol{\lambda}_0 + \mathbf{P}(\mathbf{Q})\bar{\boldsymbol{\lambda}}\,. \tag{8}$$

where $\boldsymbol{\lambda}_0 = \mathbf{Q}\mathbf{G}_I\left(\mathbf{G}_I^{\mathrm{T}}\mathbf{Q}\mathbf{G}_I\right)^{-1}\mathbf{e}$, which is a particular solution of $\mathbf{G}_I^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{e}$, and $\mathbf{P}(\mathbf{Q})$ is a projection operator [3], [5], which is for any matrix $\mathbf{Q}$, usually the unit matrix, $\mathbf{G}_I^{\mathrm{T}}\mathbf{P}(\mathbf{Q}) = \mathbf{0}$ by $\mathbf{P}(\mathbf{Q}) = \mathbf{I} - \mathbf{Q}\mathbf{G}_I\left(\mathbf{G}_I^{\mathrm{T}}\mathbf{Q}\mathbf{G}_I\right)^{-1}\mathbf{G}_I^{\mathrm{T}}$.

A. Direct Solver

If $\bar{\boldsymbol{\lambda}} = \mathbf{F}_I^{+}\mathbf{d}$ in (8) and $\mathbf{Q} = \mathbf{F}_I^{+}$ [6], [13], and after some algebraic manipulations equation (8) leads to the following equation [6], [13]

$$\boldsymbol{\lambda} = \mathbf{F}_I^{+}\left(\mathbf{d} + \mathbf{G}_I\boldsymbol{\alpha}\right), \tag{9}$$

where $\boldsymbol{\alpha} = -\left(\mathbf{G}_I^{\mathrm{T}}\mathbf{F}_I^{+}\mathbf{G}_I\right)^{-1}\left(\mathbf{G}_I^{\mathrm{T}}\mathbf{F}_I^{+}\mathbf{d} - \mathbf{e}\right)$ [6], [13]. After obtaining $\boldsymbol{\lambda}$ and $\boldsymbol{\alpha}$, the results can be calculated by (5).

This direct solver is handling the singularity of floating sub-domain.

B. Modified Conjugate Gradient Method

The conjugate gradient method is the most commonly used algorithm to solve systems of linear equations [8]. However, the coarse problem is not positive definite and therefore the classical conjugate gradient cannot be used [3]. The modified conjugated gradient method [4]-[5], [14]-[15] is used for the solution of the reduced system (7).

The solution to the system of equations (7) is identical to the problem of minimizing the quadratic form [3], [7]-[8]

$$\Phi = \frac{1}{2}\boldsymbol{\lambda}^{\mathrm{T}}\mathbf{F}\boldsymbol{\lambda} - \boldsymbol{\lambda}^{\mathrm{T}}\mathbf{d}. \tag{10}$$

with the additional condition $\mathbf{G}_I^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{e}$ [3]. Therefore a modification which is based on the application of the additional condition must be included. Additional condition must always be satisfied during the iteration process [9].

The algorithm of the modified conjugate gradient method (MCG) can be written as [3]-[5], [14]

1. Initialize

$$\begin{aligned}
\boldsymbol{\lambda}_0 &= \mathbf{G}_I\left(\mathbf{G}_I^{\mathrm{T}}\mathbf{G}\right)^{-1}\mathbf{e},\\
\mathbf{r}_0 &= \mathbf{d} - \mathbf{F}_I\boldsymbol{\lambda}_0,\\
\mathbf{w}_0 &= \mathbf{P}(\mathbf{Q})\mathbf{r}_0,\\
\mathbf{s}_0 &= \mathbf{w}_0.
\end{aligned}$$

2. Iteration k = 0, 1, ... until convergence

$$\begin{aligned}
\beta_k &= \frac{\left(\mathbf{w}_k\right)^{\mathrm{T}}\mathbf{w}_k}{\left(\mathbf{s}_k\right)^{\mathrm{T}}\mathbf{F}_I\mathbf{s}_k},\\
\boldsymbol{\lambda}_{k+1} &= \boldsymbol{\lambda}_k + \beta_k\mathbf{s}_k,\\
\mathbf{r}_{k+1} &= \mathbf{r}_k - \beta_k\mathbf{F}_I\mathbf{s}_k,\\
\mathbf{w}_{k+1} &= \mathbf{P}(\mathbf{Q})\mathbf{r}_{k+1},\\
\eta_k &= \frac{\left(\mathbf{w}_{k+1}\right)^{\mathrm{T}}\mathbf{w}_{k+1}}{\left(\mathbf{w}_k\right)^{\mathrm{T}}\mathbf{w}_k},\\
\mathbf{s}_{k+1} &= \mathbf{w}_{k+1} + \eta_k\mathbf{s}_k.
\end{aligned}$$

The drawback of this algorithm is that it cannot handle the singularity of floating sub-domain. This problem can be solved by a suitable preconditioner.

C. Preconditioned Modified Conjugate Gradient Algorithm

With the preconditioned modified conjugate gradient algorithm (PMCG) [3], [9], [15] the sub-domain problems with invertible matrices are solved in the initialize step and related sub-domain with singular matrices are solved in the iteration steps.

In this paper, the so called lumped preconditioner $\mathbf{F}_I^{\mathrm{L}}$ [3], [16] has been used. In this case, each sub-domain matrix is partitioned as [3], [5]

$$\mathbf{K}_j = \begin{bmatrix} \mathbf{K}_j^{ii} & \mathbf{K}_j^{i\Gamma} \\ \mathbf{K}_j^{\Gamma i} & \mathbf{K}_j^{\Gamma\Gamma} \end{bmatrix}, \tag{11}$$

where the superscript i and Γ designate the sub-domain interior and interface boundary DoF, respectively. The lumped preconditioner is given by [3], [16]

$$\mathbf{F}_I^{\mathrm{L}} = \sum_{j=1}^{N_S}\mathbf{B}_j\begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_j^{\Gamma\Gamma} \end{bmatrix}\mathbf{B}_j^{\mathrm{T}}. \tag{12}$$

The algorithm of the preconditioned modified conjugate gradient method (PMCG) can be summarized as [3], [9]

1. Initialize

$$\begin{aligned}
\boldsymbol{\lambda}_0 &= \mathbf{Q}\mathbf{G}_I\left(\mathbf{G}_I^{\mathrm{T}}\mathbf{Q}\mathbf{G}\right)^{-1}\mathbf{e},\\
\mathbf{r}_0 &= \mathbf{d} - \mathbf{F}_I\boldsymbol{\lambda}_0,\\
\mathbf{w}_0 &= \mathbf{P}(\mathbf{Q})^{\mathrm{T}}\mathbf{r}_0,\\
\mathbf{h}_0 &= \mathbf{P}(\mathbf{Q})\mathbf{F}_I^{\mathrm{L}}\mathbf{w}_0,\\
\mathbf{s}_0 &= \mathbf{h}_0.
\end{aligned}$$

2. Iteration k = 0, 1, ... until convergence

$$\begin{aligned}
\beta_k &= \frac{\left(\mathbf{h}_k\right)^{\mathrm{T}}\mathbf{s}_k}{\left(\mathbf{s}_k\right)^{\mathrm{T}}\mathbf{F}_I\mathbf{s}_k},\\
\boldsymbol{\lambda}_{k+1} &= \boldsymbol{\lambda}_k + \beta_k\mathbf{s}_k,\\
\mathbf{r}_{k+1} &= \mathbf{r}_k - \beta_k\mathbf{F}_I\mathbf{s}_k,\\
\mathbf{w}_{k+1} &= \mathbf{P}(\mathbf{Q})^{\mathrm{T}}\mathbf{r}_{k+1},\\
\mathbf{h}_{k+1} &= \mathbf{P}(\mathbf{Q})\mathbf{F}_I^{\mathrm{L}}\mathbf{w}_{k+1},\\
\mathbf{s}_{k+1} &= \mathbf{h}_{k+1} - \sum_{j=0}^{k}\frac{\left(\mathbf{h}_{k+1}\right)^{\mathrm{T}}\mathbf{F}_I\mathbf{s}_j}{\left(\mathbf{s}_j\right)^{\mathrm{T}}\mathbf{F}_I\mathbf{s}_j}\mathbf{s}_j.
\end{aligned}$$

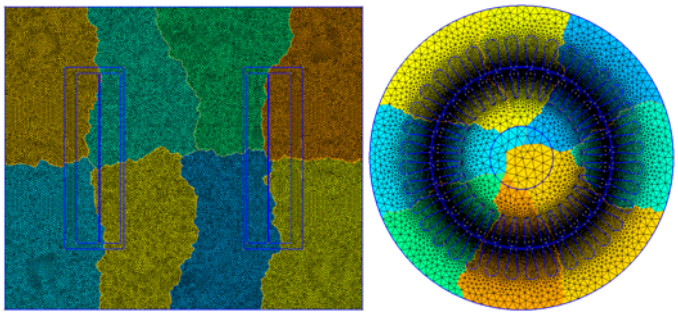

Fig. 3. The partitioned finite element mesh of the test problems.

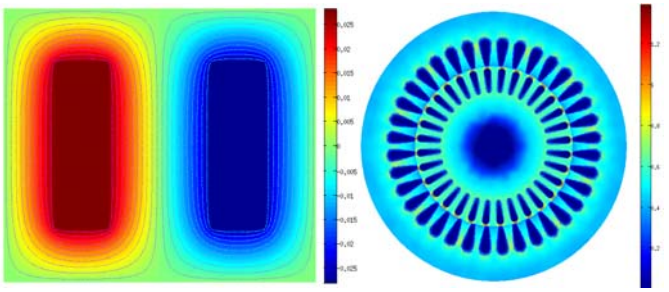

Fig. 4. The magnetic vector potential and equipotential lines of the transformer and the magnetic flux density distribution of the induction motor.

III. Test Problems

Two test problems have been used for the comparison. The first benchmark is a single-phase transformer [17], and the second one is a three-phase induction motor. Fig. 3 shows the partitioned finite element mesh of the problems. The solution of these problems can be seen in Fig. 4. The chosen test problems are static magnetic field problems, where the partial differential equations are of elliptic type [1]. The problems are discretized by triangle elements and linear nodal shape functions [1]-[2] have been used for the test problems.

IV. Results and Discussion

In order to compare the iteration counts and speedup of the methods, we have run a number of test cases using a research code that has been developed for that purpose on the Matlab computing environment. This code simulates the state of the art techniques used to implement the discussed DDMs in lower level programming languages (e.g. Fortran, C, C++) for high performance application.

The computations have been carried out on a massively parallel computer (SUN Fire X2250). This computer works with a shared memory topology with two Quad-Core Intel® Xeon® processors. The parallel programs have been implemented under the operating system Linux.

Three problems have been used to compare the time and the speedup of the function of the number of the applied processors. The 55933 number of unknowns (DoF) problem is the single-phase transformer problem. The 107828 DoF problem is the induction motor problem. The 71655 DoF problem is a quarter of the transformer, because in this case the problem contains floating sub-domain.

In order to use the same stop criterion for the iterative solvers, $\varepsilon = 10^{-9}$. The speedup has been calculated by the following formula, $Speedup = Time_1 / Time_n$ [18], where $Time_1$ is the running time of the sequential algorithm or the running time with least processor number, and $Time_n$ is the running time of the parallel algorithm executed on n processor [18].

The next figures (Fig. 5. to Fig. 10.) show the speedup and time of the function of the number of the applied processor cores. The number of processor cores is equal to the number of sub-domains during all simulations.

The parallel performance results of the solvers of the FETI method for the full transformer problem are summarized in Fig. 5 and Fig. 6. The speedups are computed using the wall clock time of sequential calculation as the reference point. Fig. 6 shows the time of the solvers is nearly same but the speedup of the solvers (see in Fig. 5.) clearly shows that the direct solver is faster than the iterative solvers.

The running performances of the solvers of FETI method for the three-phase induction motor problem are summarized in Fig. 7. and Fig. 8., when the number of processor cores is varied between 4 and 8. The speedups are computed using N_P=4 as the reference point. The direct solver also achieves good speedup, and the modified conjugate gradient solver achieves reasonable one. However, when the problem is the largest (4 processors with 26837 DoF) the PMCG is the fastest.

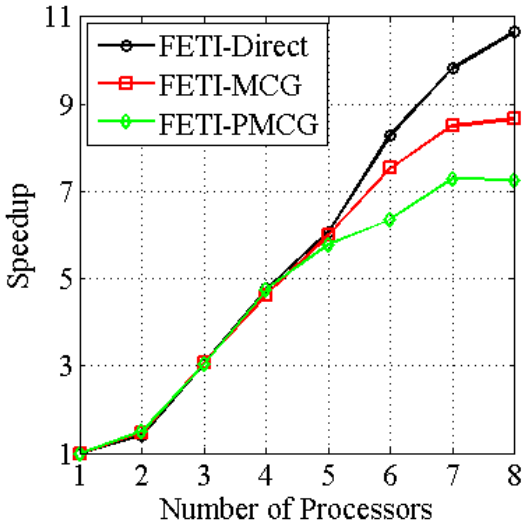

Fig. 5. The speedup of 55933 DoF problem.

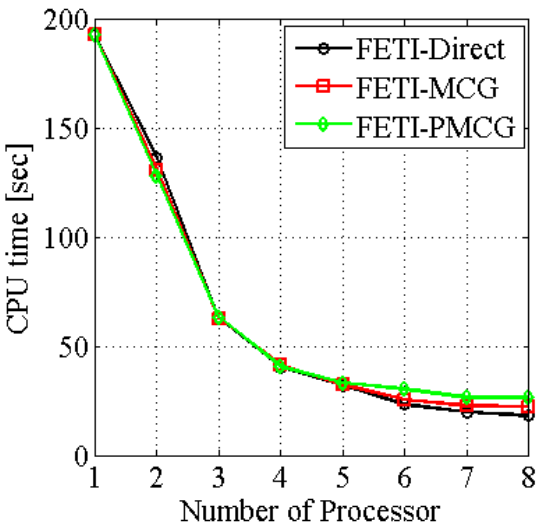

Fig. 6. The time test of 55933 DoF problem.

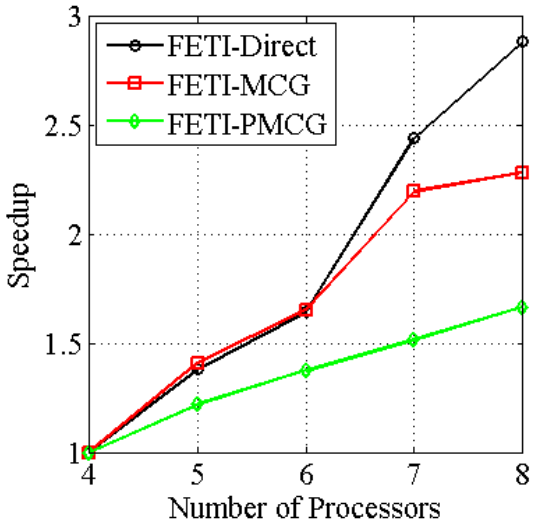

Fig. 7. The speedup of 107828 DoF problem.

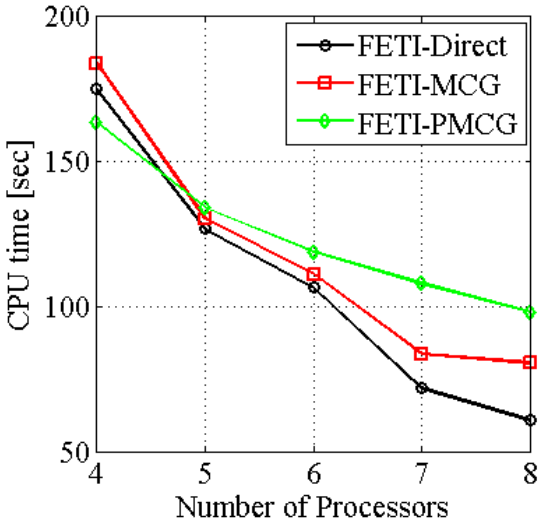

Fig. 8. The time test of 107828 DoF problem.

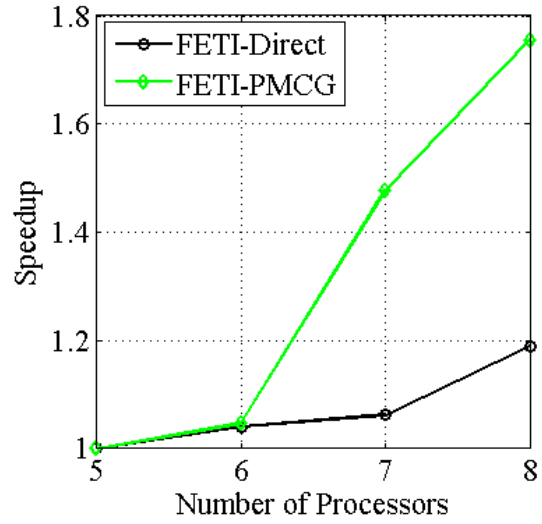

Fig. 9. The speedup of 71655 DoF problem.

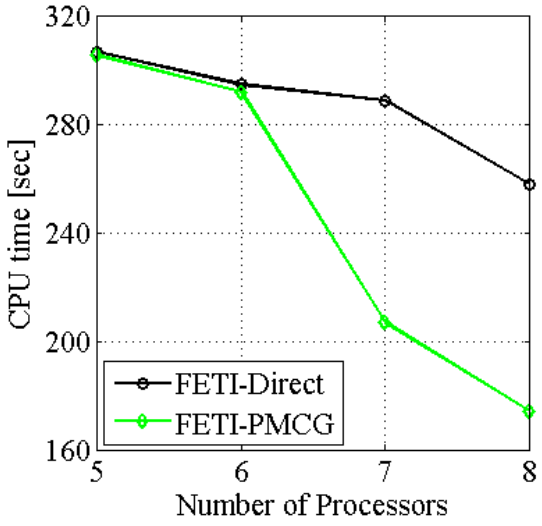

Fig. 10. The time test of 71655 DoF problem.

For $5 \leq N_P \leq 8$, the performance results of the solvers of FETI method are reported in Fig. 9 and Fig. 10 for the quarter of the transformer problem, i.e. when the problem contains floating sub-domain. In this case the preconditioned modified conjugate gradient method seems to be much better for this type of problem, than the direct solver.

Table I, Table II and Table III demonstrate the convergence properties of the iterative algorithms of FETI method for static field problems. These tables summarize: the number of processors, N_P; the number of unknowns on each sub-domains N_{DoF}; and the number of interface unknowns on each sub-domains N_{IDoF}. The number of iterations of solvers is increased, when the number of interface unknowns also is increased. Because, the number of Lagrange multipliers, i.e. the unknowns of (7) depends on the number of N_{IDoF}.

It seems to be that the modified conjugate gradient solver solved the problem faster, than the preconditioned modified conjugate gradient method. However, the number of iterations reported in Table I and Table II show that the PMCG solver of FETI method has faster convergence rate than the MCG solver of FETI method. The preconditioned FETI solver is shown to converge four times (Table I) and two times (Table II) faster than the modified conjugate gradient.

TABLE I

PERFORMANCE COMPARISON OF ITERATIVE ALGORITHMS AT 55933 DOF – TRANSFORMER PROBLEM.

N_P	N_{DoF}	N_{IDoF}	MCG N. of Iter.	PMCG N. of Iter.
2	27812	282	265	53
3	18586	406	379	78
4	13973	502	388	72
5	11205	632	576	95
6	9354	747	637	193
7	8033	811	703	135
8	7039	896	774	138

TABLE II

PERFORMANCE COMPARISON OF ITERATIVE ALGORITHMS AT 107828 DOF – INDUCTION MOTOR PROBLEM.

N_P	N_{DoF}	N_{IDoF}	MCG N. of Iter.	PMCG N. of Iter.
4	26837	525	679	337
5	21510	646	887	298
6	17950	794	1001	494
7	15337	913	1182	597
8	13494	1066	1198	541

TABLE III

DATA OF PRECONDITIONED FETI SOLVER AT 71655 DOF – QUARTER OF TRANSFORMER PROBLEM.

N_P	N_{DoF}	N_{IDoF}	PMCG N. of Iter.
5	14288	777	100
6	11987	1067	210
7	10278	1056	121
8	9007	1177	125

V. CONCLUSIONS

In this paper, we have presented the description of the Finite Element Tearing and Interconnecting method for static magnetic field problems. We have illustrated the speedup of the method and the convergence behavior of solvers with the

solution of two-dimensional static field problems on massively parallel computer.

For the three problems, the speedup achieves nearly 11-fold 3-fold and 2-fold speedup using 8 processors.

It can be concluded that the direct solver outperforms the modified and the preconditioned modified conjugate gradient algorithms. The computation costs and the memory requirement of the computation of the null space and pseudo-inverse of the sub-domain mass matrices are small, because the first two problems does not contain floating sub-domain. However, these computations can become the Achilles' heel for the direct solver in the third case. This is the reason, why the PMCG solver seems to be better than the direct solver for the quarter of the transformer problem.

The aim of the future research is to solve more complex, larger two dimensional and three dimensional problems, and to realize DP-FETI method, which is distinguished by low memory requirement and a shorter time is necessary to reach convergence.

ACKNOWLEDGEMENT

This paper was supported by the TÁMOP-4.2.2.A-11/1/KONV-2012-0012 : Basic research for the development of hybrid and electric vehicles - The Project is supported by the Hungarian Government and co-financed by the European Social Fund.

REFERENCES

[1] M. Kuczmann, A. Iványi, The Finite Element Method in Magnetics. Budapest, Academic Press, 2008.

[2] J. Luomi, Finite Element Methods for Electrical Machines (lecture Notes for postgraduate course in electrical machines). Chalmers University of Technology, Göteborg, 1993.

[3] J. Kruis, Domain Decomposition Methods for Distributed Computing. Kippen, Stirling, Scotland, Saxe-Coburg Publication, 2006.

[4] C. Farhat, F. X. Roux, "A method of Finite Element Tearing and Interconnecting and its parallel solution algorithm", International Journal for Numerical Methods in Engineering, vol. 32, pp. 1205–1227, 1991.

[5] D. J. Rixen, C. Farhat, R. Teuzar, J. Mandel, "Theoretical comparison of the FETI and algebraically partitioned FETI methods, and performance comparison with a direct sparse solver", International Journal for Numerical Methods in Engineering, vol 46, pp. 501-533, 1999.

[6] D. Marcsa, M. Kuczmann, "Comparison of domain decomposition methods for elliptic partial differential problems with unstructured meshes", Przeglad Elektrotechniczny, vol. 12b, pp. 1-4, 2012.

[7] Y. Saad, Iterative Methods for Sparse Linear Systems. 3rd edition, Philadelphia, PA, SIAM, 2003.

[8] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. M. Donato, J. Dongara, V. Eijkhout, R. Pozo, C. Romine, H. V. der Vorst, Templates for the Solution of Linear Systems: Boulding Blocks for Iterative Methods. Philadelphia, PA, SIAM, 1994

[9] C. Farhat, K. Pierson, M. Lesionne, "The second generation FETI methods and their application to the parallel solution of large-scale linear and geometrically non-linear structural analysis problems", Computer Methods in Applied Mechanics and Engineering, vol. 184, pp. 333-374, 2000.

[10] http://glaros.dtc.umn.edu/gkhome/views/metis. (Accessed November 27, 2012)

[11] C. Geuzaine, J.F. Remacle, "Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities", International Journal for Numerical Methods in Engineering, vol. 79, pp. 1309-1331, 2009.

[12] http://www.mathworks.com (Accessed November 20, 2012).

[13] P. Gosselet, C. Rey, D. J. Rixen, "On the initial estimate of interface forces in FETI methods", Computer Methods in Applied Mechanics and Engineering, vol. 192, pp. 2749–2764, 2003.

[14] F. Magoulés, F. X. Roux, "Algorithms and theory for substructuring and domain decomposition methods", in Mesh Partitioning Techniques and Domain Decomposition Methods, F. Magoulés, Eds, Kippen, Stirling, Scotland: Saxe-Coburg Publications, 2007, pp. 89-118.

[15] H. Kanayama, S.-I. Sugimoto, "Effectiveness of $\boldsymbol{A}$-φ method in a parallel computing with an iterative domain decomposition method", IEEE Transactions on Magnetics, vol. 42, pp. 539-542, 2006.

[16] Y. Fragakis, M. Papadrakakis, "The mosaic of high performance domain decomposition methods for structural mechanics: formulation, interrelation and numerical efficiency of primal and dual methods", Computer Methods in Applied Mechanics and Engineering, vol. 192, pp. 3799–3830, 2003.

[17] N. Bianchi, Electrical Machine Analysis Using Finite Elements, Taylor & Francis, Boca Rotan, FL, USA, 2005.

[18] A. F. P. Camargos, R. M. S. Batalha, C. A. P. S. Martins, E.J. Silva, G. L. Soares, "Superlinear speedup in a 3-D parallel conjugate gradient solver", IEEE Transactions on Magnetics, vol. 45, pp. 1602-1605, 2009.

Dániel Marcsa was born in Keszthely, 8th of August, 1984, in Hungary. He has become B.Sc. in Electrical Engineering in 2008, and M.Sc. in Mechatronics Enginnering in 2010 at the Széchenyi Istvan University. He is recently a Ph.D. student at the Széchenyi István University, Győr, Hungary.

His research is on static magnetic and eddy current potential formulations, numerical simulation of the electric machines by finite element method, and non-overlapping domain decomposition techniques in electromagnetic computation.

He is an active member of the Laboratory of Electromagnetic Field and Hungarian Electrotechnical Association, and member of IEEE and International COMPUMAG Society. He received the Best Presenter Award at the 6th International PhD & DLA Symposium in 2010.

Postal address: Széchenyi István University, Department of Automation, H-9026, Győr, Egyetem tér 1., Hungary.

E-mail: marcsad@sze.hu

Miklós Kuczmann was born in Kapuvár, 31st of May, 1977, in Hungary. He has become M.Sc. in Electrical Engineering in 2000, and Ph.D. in Electrical Engineering in 2005 at the Budapest University of Technology and Economics, Department of Electromagnetic Theory.

He is presently the head of the Department of Automation, Széchenyi István University, Győr, Hungary, where he is full professor since 2012.

Dr. Kuczmann has won the Bolyai János Scholarship from the Hungarian Academy of Sciences in 2006, and the "Best PhD Dissertation" award in 2006 from the same Institute.

Postal address: Széchenyi István University, Department of Automation, H-9026, Győr, Egyetem tér 1., Hungary.