Acta
Technica
Jaurinensis

Vol. 7., No.2., pp. 193-206, 2014
DOI: 10.14513/actatechjaur.v7.n2.278
Available online at acta.sze.hu

Domain Decomposition Algorithms for Edge Element Based Parabolic Type Problems

D. Marcsa[1,2]

[1]Széchenyi István University, Department of Automation
Egyetem tér 1., H-9026, Győr, Hungary
E-mail: marcsad@sze.hu

[2]Research Center of Vehicle Industry
Egyetem tér 1., H-9026, Győr, Hungary

Abstract: Today, a huge amount of interest in new actuators and a correspondingly huge pressure on companies and research centres to develop the most efficient, cost-effective electric design of the actuators for electric vehicles. However, the design and analysis of actuators is very complex task, thus need the help of multiprocessor computers to run efficient computations. This work analyzes the use of two non-overlapping domain decomposition methods (DDMs) in order to improve the calculation behaviour of finite element method (FEM) with edge element approximation. In this case, the DDMs under investigation are the Schur complement method and the Lagrange multiplier based Finite Element Tearing and Interconnecting (FETI) method with their algorithms. The performances of these methods and solvers have been investigated in detail for two-dimensional parabolic type problems as case studies.

Keywords: Domain decomposition, Iterative algorithm, Finite element method, Edge element based parabolic type problem

1. Introduction

The numerical design of electromechanical actuators of electric vehicles is a very complex task, because a lot of different physical aspects should be considered. The performances of electrical equipments are not defined only by their electromagnetic field, because the electromagnetic field has strong interaction between the following quantities: electromagnetic field distribution, mechanical equation, external circuits, etc. But, the analysis of complex system, e.g. analysis of motors of electric vehicle is very resource-intensive and time consuming, wherein the resources and time of the simulation plays an important role for designers and researchers. Therefore, the solution of a complex system must be parallelised in order to speedup the numerical computations with less computer requirement. The parallelisation of computation can be approached with the use of a domain decomposition method (DDM) [1-8]. To solve

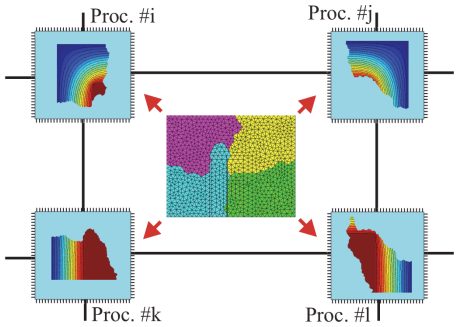

Figure 1. Demonstration of mesh partitioning and distributed computation.

large scale problems, a domain has been divided into sub-domains that are fit into the computer memory. 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 numerical design a computer with parallel processors, computations should be distributed across processors (see in fig. 1). The used DDMs are the Schur complement method and the Finite Element Tearing and Interconnecting (FETI) method.

The Schur complement method [1-3], as sequential algorithm was started to use many decades ago, when computer RAM was extremely small. Nevertheless, nowadays, this method is a very popular parallel domain decomposition technique among engineers [3].

In the last decade, the Finite Element Tearing and Interconnecting (FETI) method [4-6] has seemed as one of the most powerful and one of the most popular solvers for numerical computation. The FETI requires fewer interprocess communication, than the Schur complement method, while is still offers the same amount of parallelism [1].

The parallel finite element based numerical analysis on massively parallel computers or on clusters of Personal Computers (PCs) needs the efficient partitioning [5] of the finite element mesh. This is the first and the most important step of parallelization with the use of domain decomposition methods.

Many domain decomposition or graph-partitioning algorithms can be found in the literature. Gmsh [7] combined with METIS [8] algorithm has been used for the discretization of the domain of problem and for the mesh partitioning.

The finite element method [9, 10] is an important technique for the solution of a wide range of problems in science and engineering. Generally use vector and scalar potential functions to describe the field quantities. Finite element techniques using nodal based functions to approximate both scalar and vector potentials were first to emerge [10]. However, if the vector potential has two (in 2D) or three components (in 3D), the uniqueness of the vector potential is not so evident and it can be prescribed by implicit enforcement of Coulomb gauge [9, 10]. This problem is avoided if the vector potentials are approximated by edge finite elements and taking care about the representation of source current density [9, 10].

This paper presents an edge element based parallel approach for the solution of two-dimensional parabolic type problems by finite element method. The problems are benchmarks to show the steps of the edge element based finite element technique with DDMs. The comparison focused on the speedup and iteration of solvers of methods.

2. Problems Definition

The paper presents the steps of parallel finite element method through two 2D parabolic type problems. The studied parabolic type problems are separated into two parts: the conducting region (laminated core, steel plate) denoted by Ω_c, and the nonmagnetic and nonconducting domain (air, windings) denoted by Ω_n. The Γ_B boundary denotes that the normal component of the magnetic flux density is vanishing. The Γ_H boundary denotes that the tangential component of the magnetic field intensity is zero or a known surface current density $\boldsymbol{K}$ [9, 10].

The first problem is the quarter of a single-phase transformer (see in fig. 2), where the problem domain Ω is split into conducting Ω_c and nonconducting Ω_n part. The detailed description about the geometry of this problem you can find in [11]. The partial differential equations of this problem are [9, 10]

$$\left.\begin{aligned} \nabla \times \nabla \times \boldsymbol{T}_0 &= \nabla \times \boldsymbol{J}_0 \quad \text{in } \Omega \\ \boldsymbol{T_0} \times \boldsymbol{n} &= \boldsymbol{0} \quad \text{on } \Gamma_H \end{aligned}\right\} \rightarrow \mathbf{K}_E \mathbf{a}_E = \mathbf{b}_E, \tag{1}$$

$$\left.\begin{aligned} \nabla \times \left(\frac{1}{\mu_0} \nabla \times \boldsymbol{A}\right) &= \nabla \times \mathrm{T}_0, \quad \text{in } \Omega_n, \\ \nabla \times \left(\frac{1}{\mu} \nabla \times \boldsymbol{A}\right) + \sigma \frac{\partial \boldsymbol{A}}{\partial t} &= 0, \quad \text{in } \Omega_c, \\ \boldsymbol{A} \cdot \boldsymbol{n} &= 0, \quad \text{on } \Gamma_{B,} \end{aligned}\right\} \rightarrow \mathbf{K}_N \mathbf{a}_N = \mathbf{b}_N, \tag{2}$$

where $\boldsymbol{J}_0$ = 63902 A/m^2 is the source current density, $\boldsymbol{T}_0$ is the impressed current vector potential, $\boldsymbol{A}$ is the magnetic vector potential, $\mu = \mu_0 \mu_r$ is the permeability, $\mu_0 = 4\pi 10^{-7} \mathrm{H/m}$ is the permeability of vacuum, $\mu_r = 4500$ is the relative permeability,

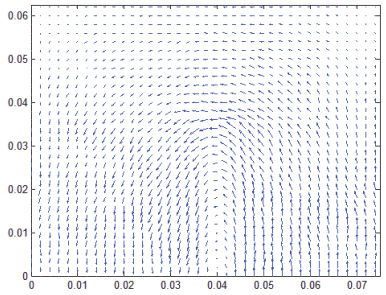

Figure 2. The solution of $\boldsymbol{T}_0$ impressed current vector potential at 10Hz.

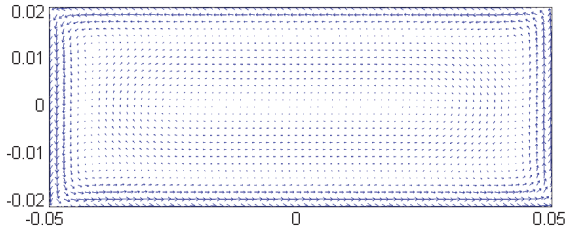

Figure 3. The solution of (3) when the frequency is 5Hz.

$\sigma = 1780$ S/m is the conductivity, and $\boldsymbol{n}$ is the outer normal unit vector. The subscript E and N denotes, the linear system of equation ($\mathbf{Ka} = \mathbf{b}$) from edge or nodal element approximation.

At this problem, the impressed current vector potential the $\boldsymbol{T}_0$ computation has been performed by edge finite element approximation, and the magnetic vector potential $\boldsymbol{A}$ has been approximated by nodal shape function, because it has only z-component, i.e. A_z. The (2) problem has been used to validate the results of parallel edge element based finite element technique.

The second problem is a steel plate (0.1 x 0.04 m) around a coil (see in fig. 3). In this problem, the conductivity and relative permeability are $\sigma = 2 \cdot 10^5$ S/m and $\mu_r = 4000$. This problem has only eddy current region Ω_c, because the excitation is the surface current density $\boldsymbol{K}$ = 200 A/m as boundary condition with the appropriate direction. The partial differential equation and boundary condition of the steel plate problem are the following [9, 10],

$$\left.\begin{array}{l} \nabla \times \left(\frac{1}{\mu} \nabla \times \boldsymbol{A}\right) + \sigma \frac{\partial \boldsymbol{A}}{\partial t} = 0, \quad \text{in } \Omega, \\ \boldsymbol{T_0} \times \boldsymbol{n} = \boldsymbol{K} \quad \text{on } \Gamma_H \end{array}\right\} \rightarrow \mathbf{K}_E \mathbf{a}_E = \mathbf{b}_E, \tag{3}$$

The above mentioned problems have been solved by finite element method, i.e. the $\Omega = \Omega_c \cup \Omega_n$ region has been discretized by FEM mesh. The weighted residual method with the Galerkin formulation has been applied to give the weak formulations of the (1), (2) and (3).

The weak formulation of the (1) and (2) are [9, 10]

$$\int_\Omega (\nabla \times \boldsymbol{W}_k) \cdot (\nabla \times \boldsymbol{T}_0^\kappa) d\Omega = \int_{\Omega_n} (\nabla \times \boldsymbol{W}_k) \cdot \boldsymbol{J}_0 d\Omega, \tag{4}$$

and

$$\int_{\Omega_c \cup \Omega_n} \frac{1}{\mu} (\nabla N_k) \cdot (\nabla A_z^\kappa) d\Omega + \int_{\Omega_c} N_k \cdot \sigma \frac{\partial A_z^\kappa}{\partial t} d\Omega = \int_{\Omega_n} (\nabla \times \boldsymbol{W}_k) \cdot \boldsymbol{T}_0^\kappa d\Omega, \tag{5}$$

where $\boldsymbol{W}_k$ is the vector weighting function of the k^{th} element, N_k is the nodal weighting function of the k^{th} element. $\boldsymbol{T}_0^\kappa$, A_z^κ denote the approximated unknown potential functions, e.g. [9]

$$\begin{aligned} \boldsymbol{T}_0 &\cong \boldsymbol{T}_0^{\kappa} = \boldsymbol{T}_{0D} + \sum_{k=1}^{n} \boldsymbol{W}_k T_{0k}, \\ \mathrm{A}_z &\cong \mathrm{A}_z^{\kappa} = \mathrm{A}_{zD} + \sum_{k=1}^{n} N_k A_{zk}\,, \end{aligned} \tag{6}$$

where $\boldsymbol{T}_{0D}$ and A_{zD} prescribe the Dirichlet boundary condition.

The weak formulation of (3) is [9, 10]

$$\int_{\Omega_c} \frac{1}{\mu} (\nabla \times \boldsymbol{W}_k) \cdot (\nabla \times \mathbf{A}^{\kappa}) \mathrm{d}\Omega + \int_{\Omega_c} \boldsymbol{W}_k \cdot \sigma \frac{\partial \boldsymbol{A}^{\kappa}}{\partial t} \mathrm{d}\Omega = \int_{\Gamma_H} \boldsymbol{W}_k \cdot \boldsymbol{K}\, \mathrm{d}\Gamma. \tag{7}$$

3. Parallel Finite Element Method with Domain Decomposition

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

The general form of a linear algebraic problem arising from the discretization of a parabolic type problems defined on the domain Ω can be written as [12]

$$\mathbf{K}\mathbf{a} = \mathbf{b}, \tag{8}$$

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

In the following, we introduced the extensive review of the Schur complement method and the FETI method for the parallel solution of edge element based problems. However, the presented techniques also useful for the solution of nodal element based problems.

3.1. Schur Complement Method

After the problem is partitioned into a set of N_S disconnected sub-domains, it can be seen in fig. 4, and equation (8) has been split into N_S particular blocks [1-3]

$$\begin{bmatrix} \mathbf{K}_{jj} & \mathbf{K}_{j\Gamma_j} \\ \mathbf{K}_{\Gamma_j j} & \mathbf{K}_{\Gamma_j \Gamma_j} \end{bmatrix} \begin{bmatrix} \mathbf{a}_j \\ \mathbf{a}_{\Gamma_j} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_j \\ \mathbf{b}_{\Gamma_j} \end{bmatrix}, \tag{9}$$

where $j = 1 \ldots N_S$, $\mathbf{K}_{jj}$ is the positive definite sub-matrix of the j^{th} sub-domain, $\mathbf{a}_j$ is the vector of the right hand side defined inside the sub-domain. The sub-matrix $\mathbf{K}_{j\Gamma_j} = \mathbf{K}_{\Gamma_j j}^{\mathrm{T}}$ contains the value of j^{th} sub-domain, which connect to the interface boundary unknowns of that region. The superscript T denotes the transpose. $\mathbf{K}_{\Gamma_j \Gamma_j}$, and $\mathbf{a}_{\Gamma_j}$ expresses the coupling of the interface unknowns.

Each sub-domain will be allocated to an independent processor core, because the sub-matrix $\mathbf{K}_{jj}$ with the $\mathbf{K}_{j\Gamma_j}$, $\mathbf{K}_{\Gamma_j j}$ and the right-hand side $\mathbf{b}_j$ are independent, i.e. they can be assembled in parallel on distributed memory. Only the $\mathbf{K}_{\Gamma_j \Gamma_j}$, and $\mathbf{b}_{\Gamma_j}$ are not independent. The matrix $\mathbf{K}_{\Gamma\Gamma}$ and the vector $\mathbf{b}_{\Gamma}$ are assembled after interprocess data

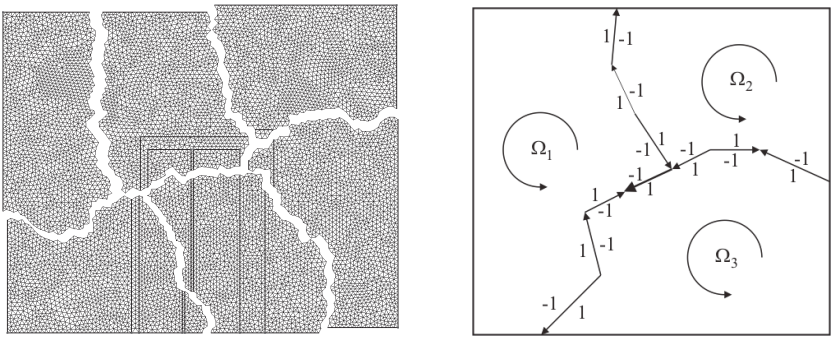

Figure 4. Example of partitioned of domain Ω, and we show the edge directions of the sub-domain interface boundaries, which is very important in the edge element based DDMs.

transfer, because they are the assembly of $\mathbf{K}_{\Gamma_j\Gamma_j}$ and $\mathbf{b}_{\Gamma_j}$, where j is the index of sub-domains.

The assembly and the solution of the sub-matrices can be performed parallel by independent processors. However, the solution requires exchange of interface values between the processes in charge of the various sub-domains. In many practical applications, the preconditioned conjugate gradient (PCG) method is used because of its simplicity and efficiency. The parallel implementation of the preconditioned conjugate gradient method can be presented algorithm 1 [2].

Algorithm 1. Parallel PCG Algorithm.

1 Initialization: $\mathbf{a_0} = \mathbf{0}$,
2 $\mathbf{r_0} = \mathbf{b}_s$
3 Assembly local $\mathbf{r_0}$ with $\mathbf{r}_{\Gamma_{\mathbf{int}}}$ entries → $\bar{\mathbf{r}}_\mathbf{0}$,
4 **for** i = 0, 1, … **do**
5 $\mathbf{w}_i = \mathbf{M}^{-1}\bar{\mathbf{r}}_i$,
6 $\gamma_i = \mathbf{r}_i^{\mathbf{T}}\mathbf{w}_i$,
7 Assembly $\boldsymbol{\gamma_i}$,
8 Assembly local $\mathbf{w}_i$ with $\mathbf{w}_{\Gamma_{\mathbf{int}}}$ entries → $\bar{\mathbf{w}}_i$,
9 **if** i = 0 **then**
10 $\bar{\mathbf{p}}_i = \bar{\mathbf{w}}_i$,
11 **else**
12 $\bar{\mathbf{p}}_i = \bar{\mathbf{w}}_i + (\gamma_i/\gamma_{i-1})\bar{\mathbf{p}}_{i-1}$,
13 $\mathbf{w}_i = \mathbf{K}_s\bar{\mathbf{p}}_i$,
14 $\beta_i = \mathbf{p}_i^{\mathbf{T}}\bar{\mathbf{w}}_i$,
15 Assembly β_i,
16 Assembly local $\mathbf{w}_i$ with $\mathbf{w}_{\Gamma_{\mathbf{int}}}$ entries → $\bar{\mathbf{w}}_i$,
17 $\bar{\mathbf{a}}_i = \bar{\mathbf{p}}_{i-1} + (\gamma_i/\beta_i)\bar{\mathbf{p}}_i$,
18 $\bar{\mathbf{r}}_i = \bar{\mathbf{r}}_{i-1} + (\gamma_i/\beta_i)\bar{\mathbf{w}}_i$,
19 **if** $\gamma_i/\gamma_0 < \varepsilon$ **then**
20 **return**

In the parallel PCG algorithm, $\mathbf{K}_S$ and $\mathbf{b}_S$ are the mass matrix and right-hand side of sub-domain, $\mathbf{a}$ is the unknown potentials, $\mathbf{r}$ is the residual vector, subscript Γ_{int} denotes the external interface entries from neighbouring sub-domains, $\mathbf{M} = \text{diag}(\mathbf{K}_S)$ is the diagonal preconditioning matrix [12], $\mathbf{w}$ and $\mathbf{p}$ are working vectors, and ε is the specified error tolerance.

3.2. Finite Element Tearing and Interconnecting Method

After mesh partitioning (see in fig. 4), the FETI method consists in transforming the original problem, equation (8) with the equivalent system of sub-domain equations [1, 4-6]

$$\mathbf{K}_j\mathbf{a}_j = \mathbf{b}_j - \mathbf{B}_j^{\mathrm{T}}\boldsymbol{\Lambda}, \tag{10}$$

with the compatibility of the magnetic vector potentials at the sub-domain interface [1, 4-6]

$$\sum_{j=1}^{N_S} \mathbf{B}_j\mathbf{a}_j = \mathbf{0}, \tag{11}$$

where $j = 1 \ldots N_S$, the number of sub-domains, $\mathbf{K}_j$, $\mathbf{b}_j$ and $\mathbf{a}_j$ are respectively the system matrix, the representation of the excitation and the unknown potentials of j^{th} sub-domain. The vector of Lagrange multipliers $\boldsymbol{\Lambda}$ introduced for enforcing the constraints (11) on the sub-domain interface, and $\mathbf{B}_j$ is a signed ($\pm$) Boolean mapping matrix, which is used to express the compatibility condition at the j^{th} sub-domain interface Γ_j

Usually, the partitioned problem may contain $N_f \leq N_S$ floating sub-domains, where sub-matrix $\mathbf{K}_f$ ($f = 1 \ldots N_f$) is singular. The floating sub-domain means a sub-domain without any constraints on boundary, e.g. Dirichlet boundary condition. To guarantee the solvability of these generally singular problems, we require that [4, 5]

$$\left(\mathbf{b}_j - \mathbf{B}_j^{\mathrm{T}}\boldsymbol{\Lambda}\right) \perp \mathbf{R}_j, \tag{12}$$

and compute the solution of equation in (10) as [4, 5]

$$\mathbf{a}_j = \mathbf{K}_j^{+}\left(\mathbf{b}_j - \mathbf{B}_j^{\mathrm{T}}\boldsymbol{\Lambda}\right) + \mathbf{R}_j\boldsymbol{\alpha}_j, \tag{13}$$

where $\mathbf{K}_j^{+}$ is a pseudo-inverse of $\mathbf{K}_j$ if $\mathbf{K}_j$ is singular, else $\mathbf{K}_j^{+} = \mathbf{K}_j^{-1}$. $\mathbf{R}_j = \text{Ker}\left(\mathbf{K}_j\right)$ is the null space of $\mathbf{K}_j$ [12], and $\boldsymbol{\alpha}$ 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 has been used here [12]. The introduction of the $\boldsymbol{\alpha}_j$ is compensated by the additional equations resulting from equation (12) [4, 5]

$$\mathbf{R}_j^{\mathrm{T}}\left(\mathbf{b}_j - \mathbf{B}_j^{\mathrm{T}}\boldsymbol{\Lambda}\right) = \mathbf{0}. \tag{14}$$

Substituting equation (13) into equation (11), and exploiting the solvability condition (14) lead after some algebraic manipulations to the following interface problem [1, 4-6]:

$$\begin{bmatrix} \mathbf{F}_{\mathrm{I}} & -\mathbf{G}_{\mathrm{I}} \\ -\mathbf{G}_{\mathrm{I}}^{\mathrm{T}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\Lambda} \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} \mathbf{d} \\ -\mathbf{e} \end{bmatrix}. \tag{15}$$

where

$$\begin{aligned} \boldsymbol{F}_I &= \sum_{j=1}^{N_S} \boldsymbol{B}_j \boldsymbol{K}_j^{+} \boldsymbol{B}_j^T, \\ \boldsymbol{G}_I &= [\boldsymbol{B}_1 \boldsymbol{R}_1 \quad \dots \quad \boldsymbol{B}_{N_S} \boldsymbol{R}_{N_S}], \\ \boldsymbol{d} &= \sum_{j=1}^{N_S} \boldsymbol{B}_j \boldsymbol{K}_j^{+} \boldsymbol{b}_j, \\ \boldsymbol{e} &= \left[\boldsymbol{b}_1^T \boldsymbol{R}_1 \quad \dots \quad \boldsymbol{b}_{N_s}^T \boldsymbol{R}_{N_s}\right]. \end{aligned} \tag{16}$$

In order to solve equation in (15) for the Lagrange multiplier vector$\boldsymbol{\Lambda}$, the following splitting of $\boldsymbol{\Lambda}$ is performed [4, 5]

$$\boldsymbol{\Lambda} = \boldsymbol{\Lambda}_0 + \mathbf{P}(\mathbf{Q})\bar{\boldsymbol{\Lambda}}, \tag{17}$$

where $\boldsymbol{\Lambda}_0$ is a particular solution of $\mathbf{G}_\mathrm{I}^\mathrm{T}\boldsymbol{\Lambda} = \mathbf{e}$ and $\mathbf{P}(\mathbf{Q})$ is a projection operator [4], where $\mathbf{Q}$ is an arbitrary positive definite matrix, in this case a unit matrix. The projection matrix is $\mathbf{P}(\mathbf{Q}) = \mathbf{I} - \mathbf{Q}\mathbf{G}_\mathrm{I}\left(\mathbf{G}_\mathrm{I}^\mathrm{T}\mathbf{Q}\mathbf{G}_\mathrm{I}\right)\mathbf{G}_\mathrm{I}^\mathrm{T}$, which the following relationship hold $\mathbf{G}_\mathrm{I}^\mathrm{T}\mathbf{P}(\mathbf{Q}) = \mathbf{0}$.

The classical preconditioned conjugate gradient method cannot be used because it is derived for systems with symmetric positive definite matrices and the matrix $\boldsymbol{F}_I$ does not fulfil this requirement [1]. Therefore a modification which is based on the application of an additional condition, $\mathbf{G}_\mathrm{I}^\mathrm{T}\boldsymbol{\Lambda} = \mathbf{e}$ must be included. Additional condition must always be satisfied during the iteration process.

The algorithm of the preconditioned modified conjugate gradient method is summarized in algorithm 2 [1, 4].

Algorithm 2. PMCG Method.

1 Initialization: $\boldsymbol{\Lambda_0} = \mathbf{Q}\mathbf{G}_\mathbf{I}\left(\mathbf{G}_\mathbf{I}^\mathbf{T}\mathbf{Q}\mathbf{G}_\mathbf{I}\right)^{-\mathbf{1}}\mathbf{e}$,
2 $\mathbf{r_0} = \mathbf{d} - \mathbf{F}_\mathbf{I}\boldsymbol{\Lambda_0}$,
3 $\mathbf{w_0} = \mathbf{P}(\mathbf{Q})^\mathbf{T}\mathbf{r_0}$,
4 $\mathbf{h_0} = \mathbf{P}(\mathbf{Q})\mathbf{F}_\mathbf{I}^\mathbf{L}\mathbf{w_0}$,
5 $\mathbf{s_0} = \mathbf{h_0}$,
6 **while** $\|\mathbf{r}\|_2 < \varepsilon$ (k = 1, …) **do**
7 $\alpha_k = \frac{\mathbf{h}_k^\mathbf{T}\mathbf{s}_k}{\mathbf{s}_k^\mathbf{T}\mathbf{F}_\mathbf{I}\mathbf{s}_k}$,
8 $\boldsymbol{\Lambda_{k+1}} = \boldsymbol{\Lambda_k} - \alpha_k\mathbf{s_k}$,
9 $\mathbf{r_{k+1}} = \mathbf{r_k} - \alpha_k\mathbf{F_I}\mathbf{s_k}$,
10 $\mathbf{w_{k+1}} = \mathbf{P}(\mathbf{Q})^\mathbf{T}\mathbf{r_{k+1}}$,
11 $\mathbf{h_{k+1}} = \mathbf{P}(\mathbf{Q})\mathbf{F}_\mathbf{I}^\mathbf{L}\mathbf{w_{k+1}}$,
12 **for** $0 \le j \le k$, **do**
13 $\beta_k^j = -\frac{\mathbf{h}_{k+1}^\mathbf{T}\mathbf{F_I}\mathbf{s_j}}{\mathbf{s}_j^\mathbf{T}\mathbf{F_I}\mathbf{s_j}}$,
14 $\mathbf{s_{k+1}} = \mathbf{h_{k+1}} + \sum_{j=1}^{k}\beta_k^j\,\mathbf{s_j}$.

In this paper, the also called lumped preconditioner $\mathbf{F}_\mathrm{I}^\mathrm{L}$ has been used [1, 4]. In this case, each sub-domain mass matrix is partitioned as in equation (9),

$$\mathbf{K}_j = \begin{bmatrix} \mathbf{K}_j^{ii} & \mathbf{K}_j^{i\Gamma} \\ \mathbf{K}_j^{\Gamma i} & \mathbf{K}_j^{\Gamma\Gamma} \end{bmatrix}, \tag{18}$$

where the superscript i and Γ designate the sub-domain interior and interface boundary DoF, respectively. The lumped preconditioner is given by [4, 5]

$$\mathbf{F}_{\mathrm{I}}^{\mathrm{L}} = \sum_{j=1}^{N_S} \boldsymbol{B}_j \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_j^{\Gamma\Gamma} \end{bmatrix} \boldsymbol{B}_j^T. \tag{19}$$

The PMCG algorithm is run sequential to solve the equation (15) interface problem. After obtaining the solution of Lagrange multipliers, $\boldsymbol{\Lambda}$, $\boldsymbol{\alpha}$ can be calculated as [4]

$$\boldsymbol{\alpha} = \left(\mathbf{G}_{\mathrm{I}}^{\mathrm{T}}\mathbf{Q}\mathbf{G}_{\mathrm{I}}\right)^{-1}\mathbf{G}_{\mathrm{I}}^{\mathrm{T}}\mathbf{r}, \tag{20}$$

where $\mathbf{r}$ is the final residual term which is equivalent the jump of potentials between sub-domains, and the sub-solution can be calculated parallel by equation (13).

4. 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) 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.

The first parabolic type problem is a shell-type transformer (see in fig. 4), and it contains 120994 edge elements, which means 81232 3-node triangular elements and 39763 nodes. At this problem, the impressed current vector potential [9] computation has been performed by edge finite element approximation, and the magnetic vector potential [9, 10] has been approximated by nodal shape function, because it has only z-component. This problem has been used to validate the results of parallel edge element based finite element technique. The second parabolic type problem is the plate problem (see in fig. 5), and it contains 73984 edge elements, which means 49152 3-node triangular elements and 24833 nodes.

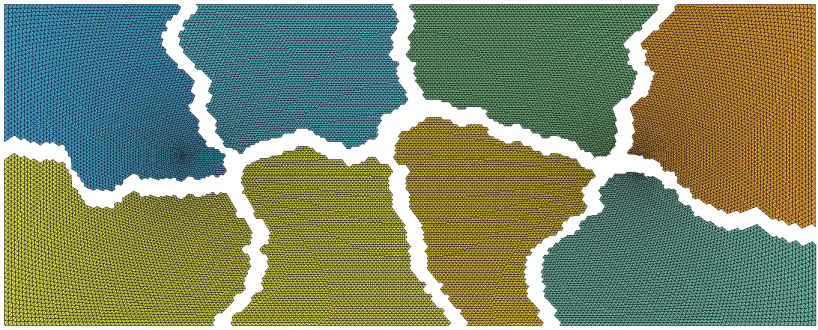

Figure 5. Irregular decomposition of mesh into 8 sub-domain.

In order to use the same stop criterion for the methods, $\varepsilon = 10^{-9}$. The speedup has been calculated by the following formula, $speedup = Time_1/Time_n$, 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 [1, 13].

In the tables are summarized the number of applied processor cores, N_p, the number of degree of freedom of one sub-domain, DoF, the number of interface unknown, $C.DoF$, the computation time (wall clock time) in seconds, $Time$, and the number of iteration of the iterative solvers, $NoIt$.

The parallel performance results of Schur complement method and the FETI method for the first parabolic type problem summarized in fig. 6, 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 Schur complement method achieves good speedups, and the FETI

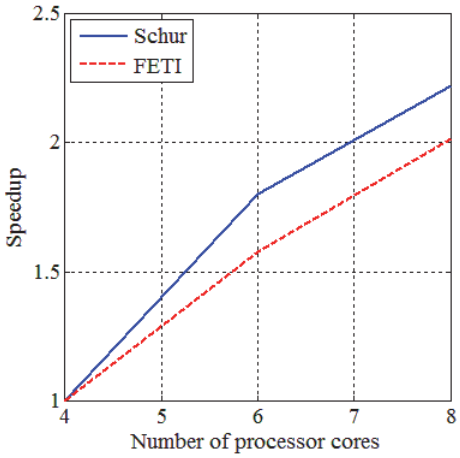

Figure 6. The speedup of the first parabolic type problem.

Table 1. Performance comparison – First parabolic type problem at 10Hz.

N_p		4	6	8
	DoF	30477	20355	15297
	$C.DoF$	424	650	827
Schur Complement Method	$Time$	152.332	84.555	68.495
	$NoIt$	1128	1134	1146
FETI Method	$Time$	801.086	508.481	397.266
	$NoIt$	204	192	201

method solver achieves reasonable ones. However, the total wall clock times ($Time$) reported in table 1 show the big difference of the running performance. The reason of this difference, the calculation of null space and pseudo-inverse are very time consuming in edge element case, because the mass matrices of sub-domains are singular.

For $1 \leq N_p \leq 8$, the performance results of the Schur complement method and FETI method are reported in fig. 7 for steel plate problem. The speedups are computed using the wall clock time of sequential calculation (197.004 sec) as the reference point. The Schur complement method achieves good speedups, and the FETI method solver achieves reasonable ones. Furthermore, the total wall clock times ($Time$) reported in table 2 show nearly same running performance, because this problem is relatively small as the transformer problem.

It seems to be, the Schur complement method solved the problem faster, than the

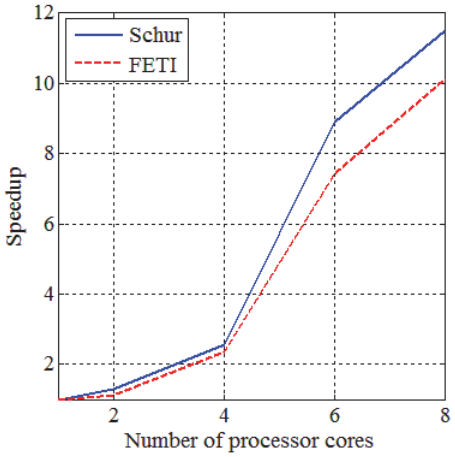

Figure 7. The speedup of the second parabolic type problem.

Table 2. Performance comparison – Second parabolic type problem at 5Hz.

N_p		2	4	6	8
	DoF	37060	18594	12436	9342
	$C.DoF$	135	416	592	736
Schur Complement Method	$Time$	151.319	77.416	22.102	17.118
	$NoIt$	171	174	176	176
FETI Method	$Time$	176.428	84.291	26.561	19.422
	$NoIt$	67	68	72	73

FETI method. However, the number of iteration reported in table 1 and 2 show the solver of FETI method faster convergence rate as the solver of Schur complement method. This conclusion is also supported by the convergence curves gives in fig. 8 and 9. These figures show the convergence curves of iterative solvers at steel plate problem when using $N_p = 8$ and three different frequencies.

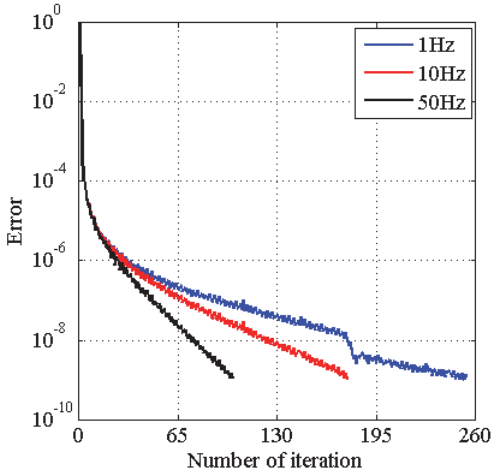

Figure 8. Convergence behaviour of PCG algorithm at different frequencies.

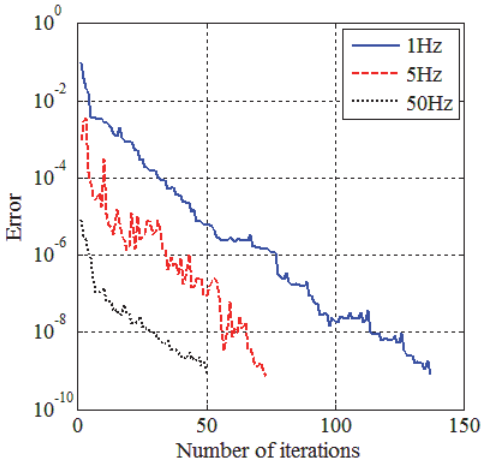

Figure 9. Convergence behaviour of PMCG algorithm at different frequencies.

5. Conclusions

In this paper, we have presented the description of Schur complement method and FETI method for edge element finite element method. We have illustrated the speedup of the methods and the convergence behaviour of solvers with the solution of two-dimensional parabolic type problems on massively parallel computer.

In all cases, the Schur complement method outperforms the FETI method. The computation of the null space and pseudo-inverse of the sub-domain mass matrices can become the Achille's heel for the FETI method in edge element case. The computation costs and the memory requirement are very high of these calculations. This is the reason, why the total times of Schur complement method are smaller. However the PMCG method convergence rate is better as the PCG method for Schur complement method.

We have to note that only two benchmarks have been used for the numerical tests. The tests with more complex three-dimensional problems will be the subject of a forthcoming work.

Acknowledgement

This paper is sponsored by "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] Kruis, J.: *Domain Decomposition Methods for Distributed Computing*, Saxe-Coburg Publications, Kippen, Stirling, 2006

[2] Nikishkov, G. P.: *Basics of the domain decomposition method for finite element analysis*, in Mesh Partitioning Techniques and Domain Decomposition Methods, Editor: Magoulés, F., Saxe-Coburg Publications, Kippen, Stirling, pp. 119-142, 2007

[3] Marcsa, D., Kuczmann, M.: *Comparison of domain decomposition methods for elliptic partial differential problems with unstructured mesh*, Przeglad Elektrotechniczny, vol. 2012, no. 12b, pp. 1-4, 2012

[4] Farhat, C., Pierson, K., Lesoinne, M.: *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, no. 2-4, pp. 333-374, 2000
DOI: 10.1016/S0045-7825(99)00234-0

[5] Farhat, C., Roux, F. X.: *Method of finite element tearing and interconnecting and its parallel solution algorithm*, International Journal for Numerical Methods in Engineering, vol. 32, no. 6, pp. 1205-1227, 1991
DOI: 10.1002/nme.1620320604

[6] Toselli, A., Vasseur, X.*: Robust and efficient FETI domain decomposition algorithms for edge element approximations*, COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering, vol. 24, no. 2, pp. 396-407, 2005
DOI: 10.1108/03321640510586033

[7] Geuzaine, C., Remacle. J. F.*: 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, no. 11, pp. 1309-1331, 2009
DOI: 10.1002/nme.2579

[8] *METIS - Serial Graph Partitioning and Fill-reducing Matrix Ordering*, available at: http://glaros.dtc.umn.edu/gkhome/views/metis (accessed 18 January 2014)
[9] Kuczmann M., Iványi, A.: *The Finite Element Method in Magnetics*. Akadémiai Kiadó, Budapest, 2008
[10] Bíró, O.: *Edge element formulation of eddy current problems*, Computer Methods in Applied Mechanics and Engineering, vol. 169, no. 3-4, pp. 391-405, 1999 DOI: 10.1016/S0045-7825(98)00165-0
[11] Bianchi N.: *Electrical Machine Analysis Using Finite Elements*, Taylor & Francis, Boca Rotan, FL, USA, 2005
[12] Saad, Y.: *Iterative Methods for Sparse Linear Systems*, 2nd edition, SIAM, 2003
[13] Molnárka G., Varjasi N.: *A Simultaneous Solution for General Linear Equations on a Ring or Hierarchical Cluster*, Acta Technica Jaurinensis, vol. 3, no. 1, pp. 65-73, 2010