Miklós KUCZMANN

"Széchenyi István" University

Parallel Finite Element Method

Abstract. *The paper presents the domain decomposition method as a tool to cut the finite element mesh into smaller sub-meshes. Smaller mass matrices can be assembled and solved in a parallel way, i.e. the solution time can be decreased by this method. The finite element method has been chosen as a numerical tool to present the applicability of the technique.*

Streszczenie. *Artykuł przedstawia metodę dekompozycji obszaru jako narzędzie do podzielenie siatki elementów skończonych na mniejsze obszary. Mniejsze macierze układu mogą być składane rozwiązywane w sposób równoległy – czas rozwiązywania może być w tym podejściu zmniejszony. Do przedstawienia skuteczności metody została wybrana metoda elementów skończonych, (**Równoległa metoda elementów skończonych**)*

Keywords: finite element method, parallel computing, domain decomposition method.
Słowa kluczowe: metoda elementów skończonych, obliczenia równoległe, metoda dekompozycji obszaru.

Introduction

The finite element method (FEM) is a numerical tool to solve electromagnetic field problems [1-4]. It is based on the weak formulation of the partial differential equations obtained from Maxwell's equations and on the finite element discretization of the geometry of the problem to be analyzed. The system of linear equations can be solved by specialized solvers according to the symmetry of the sparse mass matrix. Anyway, the solution of three dimensional problems with a huge number of unknowns can be very time consuming.

The finite element mesh can be decomposed into smaller sub-meshes with equal number of finite elements (Fig. 1). These are also called subdomains. The sub-meshes can be handled by the independent processors of a supercomputer or the independent computers of a computer grid. The equations according to the sub-meshes can be assembled in an independent, i.e. parallel way. There are nodes and edges shared by two or more subdomains. The number of such nodes and edges should be decreased, because of the communication between two connecting subdomains. After assembling the system of equations, the solution of them can be worked out in a parallel way [6-7].

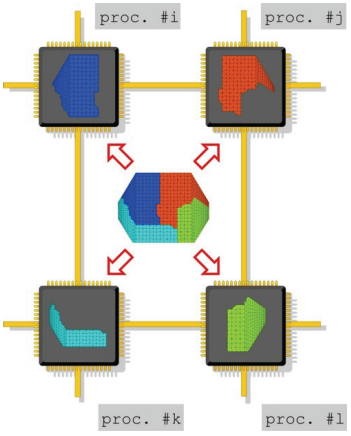

Fig. 1. Mesh partitioning and distributed computation [8]

The paper presents a simple two dimensional case study to show the steps of the parallel finite element technique.

Problem definition

This paper presents a simple case study, a two dimensional electrostatic field problem with the Laplace-Poisson equation [1,2]

$$(1) \qquad -\nabla \cdot (\varepsilon \nabla \Phi) = \rho \,,$$

where ε and ρ are the permittivity and the volume charge density, and Φ is the unknown potential, from which the electric field intensity vector can be calculated by $\boldsymbol{E} = -\nabla\Phi$. The problem region Ω is a rectangle, and homogeneous Dirichlet boundary condition has been prescribed along the perimeter of the domain Γ as it can be seen in Fig. 2.

$$\Omega: \; -\nabla \cdot (\varepsilon \nabla \Phi) = \rho$$

$$\Gamma: \; \Phi = 0$$

Fig. 2. Problem definition

The problem has been solved by the finite element technique, i.e. the region has been discretized by small triangles, and the weighted residual method with the Galerkin formulation has been applied to present the weak formulation [1-3],

$$(2) \quad \int_{\Omega} \varepsilon \, \nabla N \cdot \nabla \widetilde{\Phi} \, \mathrm{d}\Omega = \int_{\Omega} N \rho \, \mathrm{d}\Omega \,.$$

Here N is the weighting function as well as the basis function to approximate the unknown potential function by $\widetilde{\Phi}$,

$$(3) \quad \Phi \cong \widetilde{\Phi} = \Phi_D + \sum_{i=1}^{I} N_i \Phi_i \,,$$

where the first term Φ_D prescribes the homogeneous Dirichlet boundary condition, moreover N_i and Φ_i are the shape functions of the approximation and the nodal values of the approximating function, I=3 and I=6 in the case of linear and quadratic approximation.

Domain Decomposition in FEM

The geometry and the finite element mesh of the problem have been implemented in the frame of `COMSOL Multiphysics` [4], then the mesh has been exported as a text file, finally, the solver has been realized in the `C` language applying `MPI` functions [9,10].

The finite element mesh of the problem can be seen in Fig. 3. Large scale numerical simulations on parallel computers or on networks containing connected computers, such as those based on finite element methods, require the distribution of the finite element mesh to the processors or to the computers.

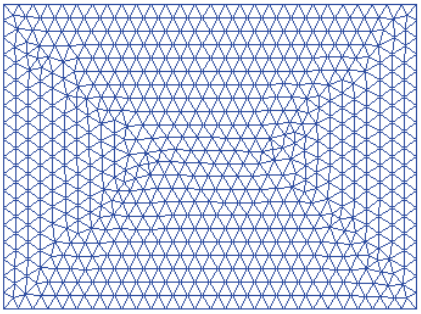

Fig. 3. Finite element mesh – an example

This distribution must be done so that the number of elements assigned to each processor is the same, and the number of adjacent elements assigned to different processors is minimized.

The goal of the first condition is to balance the computations among the processors. The goal of the second condition is to minimize the communication resulting from the placement of adjacent elements to different processors.

This task can be performed by different codes based on graph partitioning [8]. Here the `METIS` software package has been chosen [11], and the dual graph has been used. In the dual graph of the finite element mesh each finite element becomes a vertex of the graph.

Multilevel partitioning algorithms are implemented in the frame of `METIS`, i.e. the graph according to the mesh is first reduced in a coarsening phase, then an initial partitioning is performed, finally the final partition is ready after the refinement phase.

The finite element mesh can be decomposed easily into `n` parts by the `METIS` function `partdmesh`. The mesh in Fig. 3 has been cut into four submeshes, as it can be seen in Fig. 4, for illustration.

The system of equations obtained from (2) can be written as

$$\mathbf{K}\varphi = \mathbf{b}, \tag{4}$$

where **K** is the symmetric mass matrix, **b** is the right hand side of the equations and φ contains the unknown nodal potentials.

Equations in (4) have a special arrow-type pattern when applying mesh partitioning techniques, e.g. (4) has the following form when applying the mesh presented in Fig. 4:

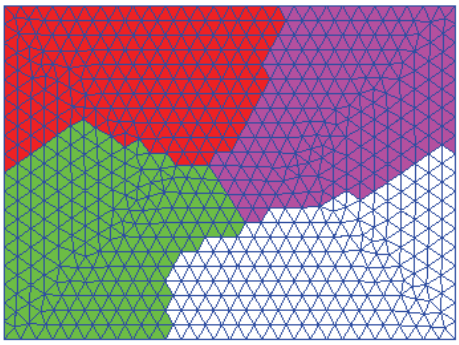

Fig. 4. Partitioned finite element mesh

$$\begin{bmatrix} \mathbf{K}_{11} & 0 & 0 & 0 & \mathbf{K}_{15} \\ 0 & \mathbf{K}_{22} & 0 & 0 & \mathbf{K}_{25} \\ 0 & 0 & \mathbf{K}_{33} & 0 & \mathbf{K}_{35} \\ 0 & 0 & 0 & \mathbf{K}_{44} & \mathbf{K}_{45} \\ \mathbf{K}_{51} & \mathbf{K}_{52} & \mathbf{K}_{53} & \mathbf{K}_{54} & \mathbf{K}_{55} \end{bmatrix} \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \\ \varphi_4 \\ \varphi_5 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \\ \mathbf{b}_4 \\ \mathbf{b}_5 \end{bmatrix}. \tag{5}$$

Here $\mathbf{K}_{11}$, $\mathbf{K}_{22}$, $\mathbf{K}_{33}$ and $\mathbf{K}_{44}$ are the symmetric sub-matrices of the four sub-regions, $\mathbf{b}_1$, $\mathbf{b}_2$, $\mathbf{b}_3$ and $\mathbf{b}_4$ are the according right hand sides assembled from the four sub-regions. The sub-matrix $\mathbf{K}_{i5}$ corresponds to the nodes of sub-region i, which are connected to the adjacent boundary of that region. The sub-matrix $\mathbf{K}_{i5}$ and $\mathbf{K}_{5i}$ are the transpose of each others, finally the sub-matrix $\mathbf{K}_{55}$ is corresponding to the nodes lying on the adjacent boundary. The same holds for $\mathbf{b}_5$, too.

The four sub-matrices $\mathbf{K}_{11}$, $\mathbf{K}_{22}$, $\mathbf{K}_{33}$ and $\mathbf{K}_{44}$ are independent, i.e. these can be assembled in a parallel way, as well as $\mathbf{K}_{i5}$, $\mathbf{K}_{5i}$, but only the particular block of the sub-matrix $\mathbf{K}_{55}$ is stored on the distributed memories. These parts are not independent. The same is true for the right hand side **b**. The assembly can be performed by the independent processors in the following way:

$$\begin{bmatrix} \mathbf{K}_{ii} & \mathbf{K}_{i5} \\ \mathbf{K}_{5i} & \mathbf{K}_{55}^{(i)} \end{bmatrix} \text{ and } \begin{bmatrix} \mathbf{b}_i \\ \mathbf{b}_5^{(i)} \end{bmatrix}, \tag{6}$$

i.e. the i^{th} processor store the data according to the domain i. The terms $\mathbf{K}_{55}^{(i)}$ are the interaction coefficients between degrees of freedom attached to nodes located on the according interface. The block $\mathbf{K}_{55}$ is the sum of these blocks as well as $\mathbf{b}_5$.

First of all, the unknowns in φ_5 can be calculated by the solution of

$$\left[\mathbf{K}_{55} - \sum_{i=1}^{4} \mathbf{K}_{5i}\mathbf{K}_{ii}^{-1}\mathbf{K}_{i5}\right]\varphi_5 = \mathbf{b}_5 - \sum_{i=1}^{4} \mathbf{K}_{5i}\mathbf{K}_{ii}^{-1}\mathbf{b}_i\ , \tag{7}$$

which unknowns correspond to the adjacent boundary of the four sub-regions, anyway the size of this system of equations can be very small. This is the so-called coarse grid problem.

Then, the unknowns of all the other sub-regions can be calculated simultaneously, i.e. in a parallel way as

$$\mathbf{K}_{ii}\varphi_i = \mathbf{b}_i - \mathbf{K}_{i5}\varphi_5\ \text{, where } i = 1,2,3,4. \tag{8}$$

The possibility of parallel computation can decrease the computational time significantly. The assembly of the sub-matrices can be performed independently, but equation (6) uses the sub-matrices from the independent processors. It means communication between the processors. After obtaining φ_5, it must be sent back to the independent processors to calculate the sub-solutions in (8), which also can be performed independently. Anyway, there is a small amount of data to send and to receive while solving the problem.

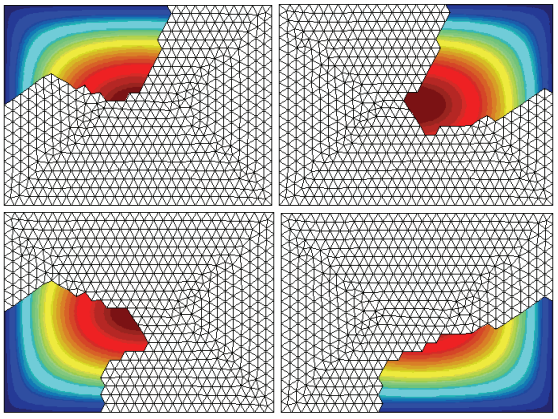

Fig. 5. The four sub-solutions

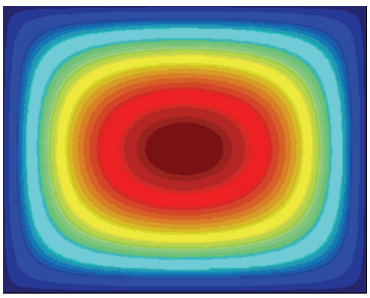

Fig. 6. The potential distribution inside the problem domain

It is important to note that equation (7) contains the inverse of the matrix $\mathbf{K}_{ii}$. The calculation of the inverse of a sparse matrix results in a dense one, and it can be a time consuming task. Instead of an inverse matrix, the $\mathbf{LDL}^T$ factorisation is computed and used.

The problem is quite small in this two dimensional example, so direct method has been chosen to solve the system of equations. The functions of the `CSparse` package have been used [12].

After obtaining φ_5, all the four sub-solutions can be calculated as it is illustrated in Fig. 5. The potential distribution of the problem can be seen in Fig. 6, which has been drawn after assembling the four sub-solutions.

Results

The problem region has been discretized using a very fine mesh to increase the number of unknowns, i.e. the size of the problem to be solved. The finite element mesh consists of 102.400 triangles, and it results in 51.601 unknowns. The approximation is first order.

In the first test, the mesh has been divided into two parts, the first domain contains 51.114 elements, the other domain contains 51.286 triangles. The number of unknowns is 25.648 and 25.729, the dimensions of the Schur complement are 224 by 224. The solution time is approximately 24 sec.

In the second test, the mesh has been divided into four parts consisting of 25.203, 25.846, 25.613 and 25.738 elements. The number of unknowns is 12.613, 12.742, 12.860 and 12.813, respectively. The dimensions of the Schur complement are 573 by 573. The solution time is approximately 15 sec.

As the last test, the mesh has been divided into six parts. The Schur complement contains 795 unknowns, and the time to catch the solution is 10 sec.

The number of the finite elements in the different sub-domains is more or less the same. The size of the Schur complement is increasing when the number of sub-domains is increasing, because the number of communication nodes is increasing. The computation costs in (6) are also increasing, but the computation time in (7) is decreasing.

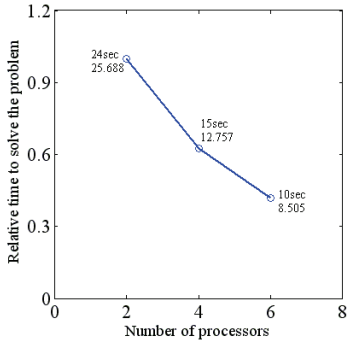

Fig. 7. Time to solve the problem vs. the number of applied processors

Fig. 7 presents a comparison between the different results. The total solution time is decreasing as the number of processors is increasing, because the number of unknowns per processors is decreasing.

The applied computer is a SUN Fire X2250 computer with the following data. CPU: 2x Quad-Core Intel Xeon L5420 @ 2,5GHz; RAM: 8x 4GB DDR2 ECC 800MHz; HDD: 2x SAMSUNG HD502IJ 500GB SATA II RAID1. This computer works with a shared memory topology. The parallel solver has been implemented under the operating system Linux (Fig. 8).

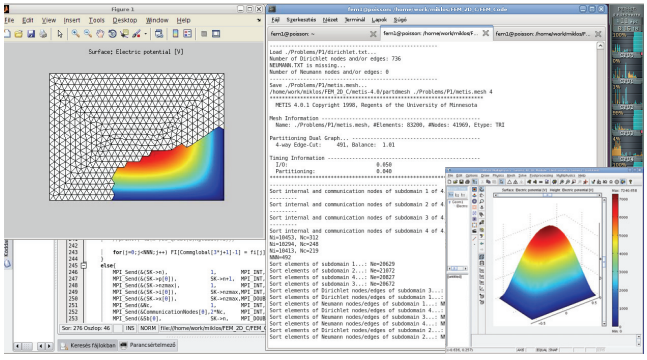

Fig. 8. The environment

Conclusions

A very simple two dimensional problem has been solved by parallel FEM. MPI functions have been implemented in the code, and it is not optimal at this moment; it must be improved in the future. More complex three dimensional problems are planned to solve.

The so called FETI method (Finite Element Tearing and Interconnecting) seems to be much better to solve the decomposed finite element problem, because the matrix $\mathbf{K}_{55}$ in (5), i.e. the matrix elements according to the communication nodes can be replaced by another technique based on Lagrangian multipliers.

Acknowledgments: This paper was supported by the Széchenyi István University (15-3210-02).

REFERENCES

[1] M. Kuczmann, A. Iványi, *The Finite Element Method in Magnetics*, Academic Press, 2008, Budapest.
[2] J. Jin, *The Finite Element Method in Electromagnetics*, John Wiley and Sons, New York, 2002.
[3] O. C. Zienkiewicz, R. Taylor, *The Finite Element Method*, McGraw-Hill, Maidenhead, 1991.
[4] Comsol Manual, COMSOL AB, 2007.
[5] F. Magoulés, *Mesh Partitioning Techniques and Domain Decomposition Methods*, Saxe-Coburg Publications, Kippen, Stirling, Scotlad, 2007.
[6] F. Magoulés, *Substructuring Techniques and Domain Decomposition Methods*, Saxe-Coburg Publications, Kippen, Stirling, Scotlad, 2010.
[7] J. Kruis, *Domain Decomposition Methods for Distributed Computing*, Saxe-Coburg Publications, Kippen, Stirling, 2006.
[8] `http://morpheus.pte.hu/~peteri`
[9] P. K. Jimack, N. Touheed, Developing Parallel Finite Element Software Using MPI, *High Performance Computing for Computational Mechanics*, ed. B.H.V. Topping and L. Lammer (Saxe-Coburg Publications), pp.15-38, 2000.
[10] P. K. Jimack, N. Touheed, An Introduction to MPI for Computational Mechanics, *Parallel and Distributed Processing for Computational Mechanics: Systems and Tools*, ed. B.H.V. Topping (Saxe-Coburg Publications), pp.24-45, 1999.
[11] `http://glaros.dtc.umn.edu/gkhome/views/metis`.
[12] T. A. Davis, Direct Methods for Sparse Linear Systems, Sociatey for Industrial and Applied Mathematics, Philadelphia, 2006.

***Author**: Dr.-Habil. Miklós Kuczmann, PhD, Széchenyi István University, Department of Telecommunications, Laboratory of Electromagnetic Field, Egyetem tér 1, H-9026 Győr, Hungary, E-mail: kuczmann@maxwell.sze.hu.*