Evaluation of Computational Complexity of Finite Element Analysis Using Gaussian Elimination

Petro Shmigelskyi, Ihor Farmaga, Piotr Spiewak, Lukasz Ciupinski

Abstract— **This paper describes the evaluation of computational complexity of software implementation of finite element method. It has been used to predict the approximate time in which the given tasks will be solved. Also illustrates the increasing of computational complexity in transition from two to three dimensional problem.**

Index Terms— **Finite element methods, Computational complexity, Interpolation, Linear approximation.**

I. Introduction

THE issue of computational complexity of FEM is especially critical for the analysis of bodies with a very heterogeneous structure [1], described by a huge amount of mesh nodes. Having answered the question and knowing the size of the input data, we can determine whether the task can be solved using available computer, and whether the solution will be obtained in a reasonable time.

II. Asymptotic notation

The function of computing time complexity in some cases can be determined accurately. In most cases it is not required to find its exact value. The exact value of the time complexity depends on determining the elementary operations (e.g., the complexity can be measured in the number of arithmetic operations, bit operations or operations of Turing machine). When increasing the size of the input data, the contribution of constant factors and terms of lower order, which appear in the expression, is quite small for the exact work time. Mathematical notation, which allows to reject details of the algorithm analysis, is called asymptotic notation and is denoted by $O(f(N))$; it is the notation that will be used to describe the complexity of algorithms [2].

Manuscript received April 20, 2011. This work was supported by Department of Computer-Aided Design Systems (Lviv Polytechnic National University) and the Faculty of Materials Science and Engineering (Warsaw University of Technology).

Petro Shmigelskyi is with the Lviv Polytechnic National University, Ukraine (e-mail: petro.shmigelskyi@gmail.com).

Ihor Farmaha is with the Lviv Polytechnic National University, Ukraine (e-mail: ifarmaga@gmail.com).

Piotr Spiewak is with the Warsaw University of Technology, Poland (e-mail: pspiewak@inmat.pw.edu.pl).

Lukasz Ciupinski is with the Warsaw University of Technology, Poland (e-mail: lukas@inmat.pw.edu.pl).

III. Algorithm analysis

Finite Element Method Algorithm

There are many algorithms for the implementation of the FEM, but they all contain the basic steps shown in Fig. 1.

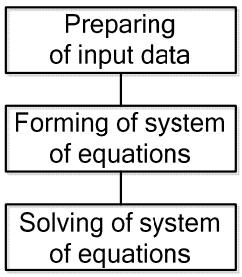

Fig.1 Stages of FEM.

Preparation of input data includes the formation of finite element mesh. We will not evaluate its complexity, as it depends heavily on its generation algorithms: in some cases it may be a simple task, in other its complexity exceeds the complexity of solving the remaining phases of the FEM, as well as in most tasks the mesh is created once and used in many simulations.

Computational complexity

For instance, to conduct the analysis of algorithm complexity, we take the one described in [3]. Here the banded stiffness matrix with bandwidth W is used. The number of nodes is denoted by N, and the number of elements – E.

The formation of global matrices of stiffness and forces is done via the recording of values obtained for individual elements, taking into account boundary conditions. The amount of operations needed for this purpose equals $C \cdot E$, where $C=const$ – the number of operations for the formation of local matrices of one element. In the

asymptotic notation the constant factors are not taken into account, so it will look like:

$$O(E). \quad (1)$$

Global matrices need modification to incorporate prescribed nodal values. In the worst case the complexity of this phase will be:

$$O(NW). \quad (2)$$

The next step solves the system of equations. Because of its huge size, the use of FEM without a computer is not reasonable. To solve this problem many different methods are used. In the tested program Gaussian elimination is used, which allows accurate solution of the system. The method implementation is divided into two subroutines. The first one reduces the matrix to upper triangular, its asymptotic complexity is:

$$O(NW^2). \quad (3)$$

The second finishes the solution of the system, and its complexity:

$$O(NW). \quad (4)$$

Having added all gained complexities, we obtain expression for the whole algorithm. Given large W, the function of the algorithm will converge to its third member, which is growing the fastest and therefore only considered asymptotic complexity of the whole FEM algorithm is equal to:

$$O(E)+O(NW)+O(NW^2)+O(NW) = O(NW^2). \quad (5)$$

IV. Research of Results

Solution time of two dimensional problem

Having computational complexity of the algorithm we can predict the approximate time in which the given task will be solved. We need to conduct a number of previous tests on the computer to be used. For more accurate prediction these launches are conducted with large input data. Now, knowing the time in which the problem has been solved and its dimensions, a time of solving other tasks can be provided proportionally, through asymptotic complexity. These survey results are presented in Tables I and II, where column *t exp* shows the time of solving of the tasks, obtained experimentally, and column *t pre* – the predicted time. In the Table I the third experiment has been taken as the basis of time prediction, in Table II – the fourth one.

As it can be seen from the Table I, high precision of time prediction is achieved for large values of input data, since the used asymptotic complexity does not consider members of the lower orders, and for large input their impact on the entire function is small. For more accurate prediction of the solution time, the results of the task, which dimension is the closest to the explored task dimension, has to be taken as the basis. For small input data the full expression of complexity Eq. (5) can be used, and previously rejected factors must be taken into account within each member. However, this assessment does not guarantee high predicting accuracy.

TABLE I
PREDICTED SOLUTION TIME

№	*N*	*W*	*t exp, sec*	*t pre, sec*	δ, %
1	251001	502	536.42	546.93	1.92
2	75 651	502	165.20	164.84	0.22
3	38 160	361	43.00	basis	
4	27 391	302	14.35	21.60	50.52
5	7 360	161	1.12	1.65	47.32

TABLE II
PREDICTED SOLUTION TIME

№	*N*	*W*	*t exp, sec*	*t pre, sec*	δ, %
1	251001	502	536.42	363.34	32.27
2	75 651	502	165.20	109.51	33.71
3	38 160	361	43.00	28.56	33.58
4	27 391	302	14.35	basis	
5	7 360	161	1.12	1.10	1.79

Evaluation of memory usage

Most memory in the program is needed to store the system of equations, which consists of stiffness matrix K, the vector of desired values Φ and vector of forces F (Fig. 2). To store the system we need M_G memory cells:

$$M_G = N(W+2L) \quad (6)$$

where L is an amount of unknown values in one node.

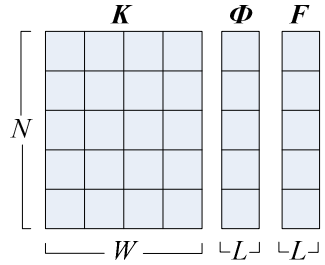

Fig. 2. Presentation of the system of equations in memory

For elements storage using an array that stores numbers of its nodes, the size equals:

$$M_E = n \cdot E, \quad (7)$$

where n is an amount of nodes in one element. The second array stores the coordinates of nodes; its size is equal to:

$$M_N = d \cdot N, \quad (8)$$

where d is the dimension of space.

Other expenses of memory are not taken into consideration as they are much smaller and do not depend on input data.

For example, when solving the problem of deformation of plates with one million elements and 500 000 nodes, with the bandwidth of 500, triangular elements with three nodes are used. To store the nodes we use *Long* data type with the size of 4 bytes, and for the coefficients of equations and nodes coordinates - *Double* type with the size of 8 bytes. Then, to store the described arrays we need the following amount of memory:

$$N(W+2L+d)\cdot 8 + n\cdot E\cdot 4 = 5\cdot 105\cdot(500+2\cdot 2+2)\cdot 8 + 3\cdot 106\cdot 4 = \\ = 2024\cdot 106\text{B} \approx 1{,}89\ \text{GB}.$$

Comparing of computational complexities of two and three dimensional problems

Using the equations obtained from previous sections, we will conduct a comparison of computational complexities for two and three dimensional problems.

For illustrative comparison of complexities consider cubic body (Fig. 3). This will simplify our calculations, but will clearly illustrate the complexity of the transition to three-dimensional problem using Gaussian elimination. Body divided into a uniform grid with h nodes per each edge. Denote the number of nodes needed to solve two dimensional problems through N_{2D} and bandwidth through W_{2D}. For three dimensional problems these values denote respectively N_{3D} and W_{3D}, each of h times larger than its two-dimensional analogue (9),(10).

$$N_{3D} = hN_{2D} \tag{9}$$

$$W_{3D} = hW_{2D} \tag{10}$$

By substituting of obtained number of nodes and bandwidth for three-dimensional problem to (5) and dividing to complexity of two dimensional problem (5) we obtain an expression that shows how many times the three-dimensional problem is more complex of its two dimensional analogue (11).

$$O(N_{3D}W_{3D}^2)/O(N_{2D}W_{2D}^2) = \\ = O(hN_{2D}(hW_{2D})^2)/O(N_{2D}W_{2D}^2) \approx h^3 \tag{11}$$

Now try to show how increased complexity of calculations in the solution of three dimensional problems, in comparison of two dimensional. For example, we consider square area. Uniform mesh is constructed so that every edge accounts 100 nodes.

Then in transition to three dimensional problem which describes the cube, according to (11) computational complexity will increase in 100^3 = 1 million times. Even if such a two dimensional problem will be calculated in 1 second, the three dimensional solution takes about 12 days.

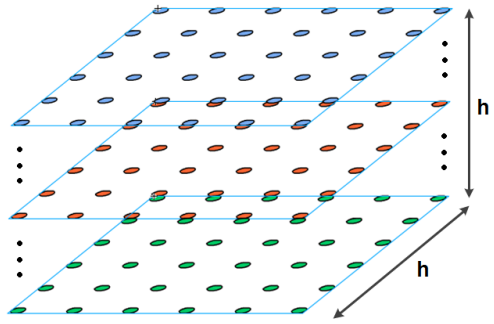

Fig. 3. Nodes location for three dimensional cubic body

Now conduct an approximate evaluation of machine memory using for example of a cubic body. According to (9), (10) the number nodes and bandwidth increase in the matrix in h times. In this evaluation we do not take into account the expressions of lower orders, so by substituting (9), (10) into (6) and dividing by (6) we obtain an approximate evaluation of increasing of memory using in the transition to three dimensional problems:

$$N_{3D}(W_{3D}+2L)/N_{2D}(W_{2D}+2L)) = \\ = hN_{2D}(hW_{2D}+2L)/(N_{2D}(W_{2D}+2L)) \approx h^2 \tag{12}$$

For example described above, obtained value shows that the memory usage will grow in almost 10 000 times. So even if the solution of two dimensional problem used only about 8 MB of memory, now this number will reach 80 GB which are not available for modern personal computers.

Perform an approximate evaluation of what size of three-dimensional cubic body problem our program can solve. Take the time limit in 10 hours. In calculating we based in the results from Table I from the first row. They obtained for a square body described by uniform grid on each side of which h_{2D} = 501 nodes. Number of nodes in the grid $N_{2D} = h_{2D}^2$, bandwidth of conductivity matrix $W_{2D} = h_{2D} + 1$. This problem was solved in T_{2D} = 536.42 seconds. For the three-dimensional grid $N_{3D} = h_{3D}^3$, $W_{2D} = h_{3D}^2$. Now use evaluation of complexity (5) to determine how many nodes can be on edge of three-dimensional grid (Fig. 3) the solution of the problem lasted for 10 hours:

$$N_{3D}W_{3D}^2/T_{3D} = N_{2D}W_{2D}^2/T_{2D}$$

$$h_{3D}^7/T_{3D} = h_{2D}^4/T_{2D}$$

$$h_{3D}^7 = h_{2D}^4 T_{3D}/T_{2D}$$

$$h_{3D} = \sqrt[7]{h_{2D}^4 T_{3D}/T_{2D}} = \sqrt[7]{502^4 \cdot 36000/536} \approx 63$$

If the number of nodes is increased only by one to 64, the solution time will increase by 18 minutes.

The results show that software implementation of FEM is still possible to use Gaussian elimination at solving of two dimensional problems. But this method is unacceptable

costly for solving of the three-dimensional problems with large amount of nodes. Because the number of equations in such problems is increasing rapidly. The complexity of the cubic Gauss entire task complexity grows very rapidly, making this method unsuitable for large problems.

V. Conclusion

On the basis of analysis of asymptotic complexity of algorithm, it is possible to determine its critical places that have the greatest impact on performance. For the considered example the subroutine solving system of equations is proper. When input data is huge, the complexity of the whole problem is close to its complexity. Gaussian elimination can be used for systems with thousands of equations and unknowns, but when their amount reaches several million, the cost of solution becomes too large. In such cases the special iterative methods are used. Analysis of such methods is more difficult because their work time depends on the needed accuracy of the solution.

Number of nodes (N) appears in all expressions of algorithm complexity, both computational and of memory usage, which indicates the extreme importance of careful preparation of input data to get the most simplified model. The factor of the obtained complexity, which depends on the bandwidth of the matrix, grows the fastest. So, when preparing a finite element mesh, one has to pay close attention to the numbering of nodes in order to achieve as small as possible bandwidth.

Acknowledgment

The authors would like to place on record the help TERMET project received from the Department of Computer-Aided Design Systems (Lviv Polytechnic National University) and the Faculty of Materials Science and Engineering (Warsaw University of Technology).

References

[1] K. Kurzydlowski, M. Lobur, I. Farmaga, O. Matviykiv. Data Processing Method for Determination Thermophysical Parameters of Composite Materials // IEEE MEMSTECH'2010. – Polyana, 2010. – PP. 264-266.

[2] I. Farmaga, P. Shmigelskyi, P. Spiewak, L. Ciupinski. Evaluation of Computational Complexity of Finite Element Analysis // IEEE CADSM'2011. – Polyana, 2011. – PP. 213 - 214.

[3] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms, Second Edition. MIT Press and McGraw-Hill, 2001. Section 3.1: Asymptotic notation, PP. 41–50.

[4] Larry J. Segerlind. Applied Finite Element Analysis. Second Edition. – John Wiley and Sons, 1984.

Petro Shmigelskyi was born in Drohobych, Ukraine, Lviv reg., in August 19 1988. In 2010 got Master's degree of computer science at Lviv Polytechnic National Univercity, Ukraine.

He starts his career in computer development in 2007 in Lvov. At 2011 become PhD student at Lviv Polytechnic National University at CAD system department. The field of study is mathematical models for modeling of thermal processing in nanocomposites.

Publications: I. Farmaga, P. Shmigelskyi, P. Spiewak, L. Ciupinski. Evaluation of Computational Complexity of Finite Element Analysis // IEEE CADSM'2011. – Polyana, 2011. – Pp. 213 – 214; Farmaha, U. Marikutsa, P. Shmigelskyi. Solving of heat transfer problem of composite materials by finite element method. XVII Polish - Ukrainian Conference on "CAD in Machinery Design - Implementation and Educational Problems". – Lviv: Publishing House Vezha&Co. 2010.- Pp. 124 – 125; Bilal Radi A'Ggel Al-Zabi, Serhiy Tkatchenko, Petro Shmigelskyi. Solving of Typification Problem by Selection of Isomorphic Subgraphs //IEEE MEMSTECH'2010.- Polyana-Svalyava, UKRAINE: Publishing House Vezha&Co. 2010.- Pp.111-112.