Multiblock method for database generation in finite element programs

DANIELA CÂRSTEA
High School Group of Railways Craiova
Str. Doljului nr. 14, bl. C8c, sc.1, apt.7, Craiova
ROMANIA

Abstract. A pre-processor for the generation of the mesh 2D in a CAD product based on the finite element method (FEM) is presented. Our software product is based on the multiblock method and is implemented in C language. A friendly interface user-program is included and some communication languages are available in the communication protocol. In our software product the database consists in a set of files (text or binary files). These files contain both geometrical data of the elements and physical properties (field sources, material properties, boundary conditions etc). The database can be used by well known software products for graphics processing and post processing stages of a finite element program.

We present some aspects of parallel implementation of the pre-processor.

Keywords: Finite-element method; CAD.

1 Introduction

In many engineering applications in the area of field computation, the numerical models are based on the finite element method (FEM). The FEM programs have a modular form in accordance the stages of the method: pre-processing, solution (processing) and postprocessing. The greatest task in any finite element program is generally the preparation of the input data. There is a large amount of input data that consists in both geometrical and physical data.

Our work concerned to pre-processing of the FEM programs. In order to eliminate much of the effort involved in data preparation for finite element programs, an automatic or semi-automatic mesh generation can be employed. In this way the data errors which inevitably occur during manual preparation are eliminated. There are a lot of methods for mesh generation but some of them are not efficient. A semi-automatic approach is very efficient.

The mesh generator can be implemented both in conventional and parallel computers. On parallel computers the speedup of the computation can be improved. The conventional mesh generator can be used both in the case of convex domains and for domains with shapes not too different from a convex. In “twisted” geometry a suitable transport function must be defined but it is difficult to find a transport function for non-convex domains.

2 The multiblock method

The multiblock method [3], sometimes referred to as super-element method, has some advantages over many methods. One main advantage consists in the fact that it is well suited for parallel computation. This is a top research area and our program was created with this idea in mind. It was tested for many applications for conventional computers but can be extended for advanced architectures, like parallel computers.

The basic idea consists of partitioning the domain of field into a set of curved-sided general quadrilateral blocks. Each block of this coarse partitioning is then meshed in triangles. Although the method is efficient for parallel architectures, it includes two kinds of difficulties:

- The creation of elementary blocks and their meshes
- The management of the block interfaces

As input data, the mesh generator requires data for the coarse mesh of the domain in terms of blocks of quadrilateral nature and, in addition, data for point's distribution on the block edges. The number of the points must be consistent (i.e., two logically connected edges must have the same number of control points). The control points can be generated automatically in the case of “straight edges”, but for “curved edges” the control points are given explicitly in order to describe the shape of the “curve” accurately.

3 A conventional algorithm

The idea of the multiblock method is simple [2]: the spatial domain is divided into a few large blocks or zones (the input data), and the subdivision proceeds automatically. The algorithm has four steps presented in the figures 1- 4.

In a semi-automatic generator the block is specified by:

- The number of the elements of division on the directions ξ and η, called n_ζ and n_η.
- The weighting factors.

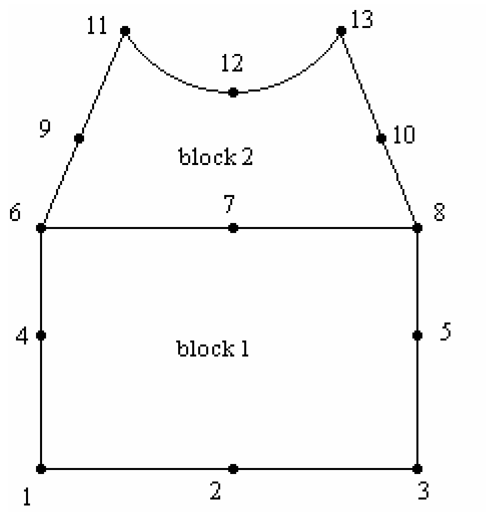

Fig. 1 - The definition of blocks

The algorithm in pseudo-code has the following form [1]:

1. *The construction of a coarse mesh. It consists in:*
   - *Definition of the structural blocks*
   - *Definition of the control points on the edge of the blocks*
2. *The subdivision of each block in more blocks in accordance with the position and the number of points on its edges*
3. *The connection of individual meshes with nodal renumbering*
4. *The subdivision of each quadrilateral element in triangles to obtain the final mesh*

Each step of the algorithm involves some operations that are not easy to implement. For example, the step 2 of the algorithm uses transport-deformation functions that depend by the domain (convex or non-convex), and the block nature [3].

Step 1. Definition of the blocks

Our mesh generator starts with quadrilaterals with parabolic sides. Each block has the general form of a curved-sided quadrilateral and is represented by the 8-nodded quadratic isoparametric element described in [4]. A quadrilateral is defined by four vertices and four points located on each of the sides (see Fig. 1). For a quadrilateral domain, let *n* be the number of points describing edges 1 and 3 and *m* corresponding to edges 2 and 4. The method creates a mesh consisting of (*n-1*) (*m-1*) quadrilateral elements.

Step 2. The subdivision process

In the step 2 of the algorithm, each block is subdivided into quadrilateral elements according to a fineness of subdivision to be specified as data input (Fig. 2). This is a feature that can permit a parallel implementation of the algorithm. The subdivision process is performed individually for each block and element nodal points in each block are numbered separately. For this step of our software product, we use linear 4-noded quadrilateral elements. The elements are formed by linearly connecting grid points, as shown in Fig.2.

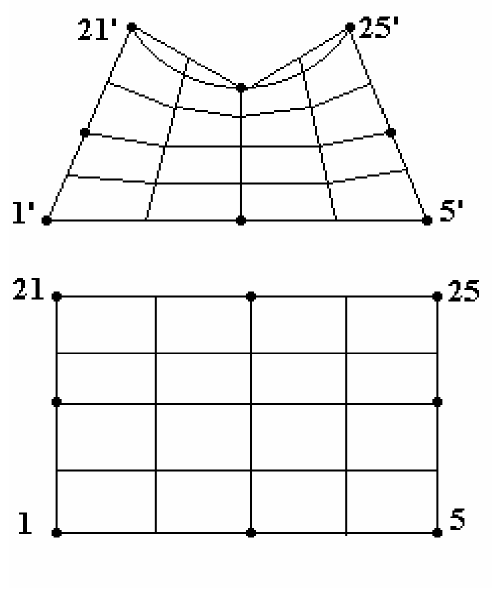

Fig. 2 - Subdivision of a structural block

The natural co-ordinate system ξ and η permits elements with curvilinear shapes to be considered. The co-ordinate values x and y at any point within the element are given by the expressions [4]:

$$x(\xi,\eta) = \sum_{i=1}^{n} N_i^{(e)}(\xi,\eta)\cdot x_i$$

$$y(\xi,\eta) = \sum_{i=1}^{n} N_i^{(e)}(\xi,\eta)\cdot y_i$$

with: x_i, y_i the co-ordinates of the node *i* and N_i – shape function. The natural curvilinear co-ordinates ξ and η range between the limits ± 1 and the direction of ξ is defined by the order of numbering of the first 3 element node numbers, following an anticlockwise sequence around the element.

The interpolation functions for quadrilateral elements are [4]:

- For a vertex:

$$N_i(\xi,\eta) = (1+\xi\xi_i)(1+\eta\eta_i)(\xi\xi_i+\eta\eta_i-1)/4$$

- For a middle point of an edge:

$$\xi_i = 0: \quad N_i(\xi,\eta) = (1-\xi^2)(1+\eta\eta_i)/2$$

$$\eta_i = 0: \quad N_i(\xi,\eta) = (1-\eta^2)(1+\xi\xi_i)/2$$

In a semi-automatic generator we must define the block. In our generator we specify:

- The number of the elements of division on the directions ξ and η, called n_x and n_y.
- For a non-uniform mesh we specify the weighting factors.

Denoting by $(p_\xi)_i$ and $(p_\eta)_i$ the weighting factors, the grid points can be located [2]. Thus:

1. Initiate the co-ordinates ξ and η
2. Compute the natural co-ordinates for the division:

$$\xi_i = \xi_{i-1} + \frac{2(p_\xi)_i}{p_{\xi T}\cdot c} \qquad (1)$$

$$\eta_i = \eta_{i-1} + \frac{2(p_\eta)_i}{p_{\eta T}\cdot c} \qquad (2)$$

where

$$p_{\xi T} = \sum_{i=1}^{n_x}(p_\xi)_i \,,\, p_{\eta T} = \sum_{i=1}^{n_y}(p_\eta)_i$$

The value for c in equations (1)-(2) is 1 for linear elements (defined by 4 nodes), and 2 for quadratic elements (defined by 8 nodes).

Step 3. The connection of individual blocks

The subdivision process is performed individually for each block. The element nodal points in adjacent blocks are numbered separately.

The nodal points along block interfaces will have common co-ordinate values and therefore should be uniquely numbered. Thus after subdividing each block in a mesh, it is necessary to search for nodal points along block interfaces, by comparing the co-ordinates of all nodal points, and then assigning a single nodal number to the nodes with identical co-ordinate values.

Step 4. Generation of the triangular mesh

In the step 4 of the algorithm, a triangular mesh can be created. The quadrilaterals can subsequently be split into two "optimal" triangles (i.e. splitting with respect to the shortest diagonal).

In order to allow the mesh to be graded, the absolute or relative size of the elements in a particular block is defined or computed. In the first approach we compute the absolute size of the elements. The user defines the spacing at the edge ends *i* and *j*.

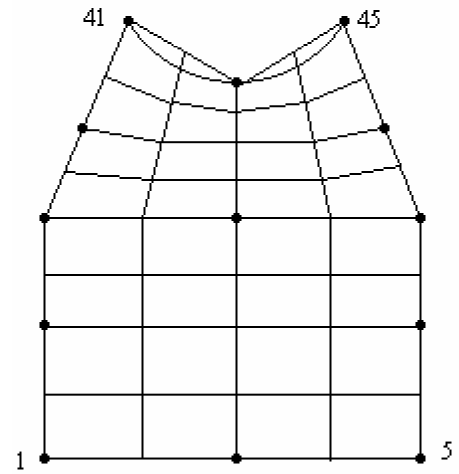

Fig. 3 - The connection of individual meshes

For adjacent blocks it is necessary to specify the spacing values so that the subdivisions along block interfaces are compatible.

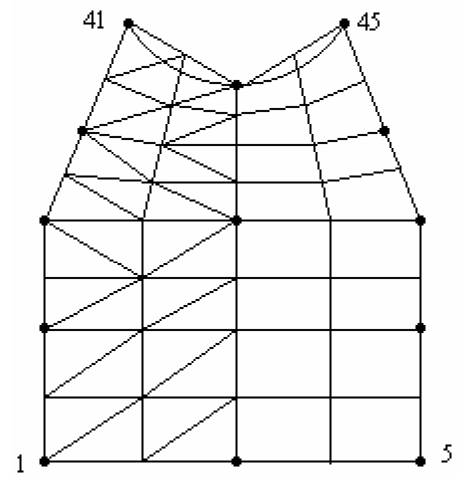

Fig.4- Final mesh with triangular elements

In another approach, the weighting factors, which prescribe the relative size of elements within a block, are generated automatically both in ξ-direction and η-direction. The user defines the spacing at the ends of an edge of the block. The formulae that we use for estimating the number of points that will be produced by our mesh generator are simple and cheap.

Finally, the generated mesh is processed so that a bandwidth minimisation is obtained. This procedure is based on a judicious ordering of the structural blocks and node numbers allocated.

4 Database generation

The scope of the generator is to create automatically the database necessary in the following stages of the FEM program: processing and post-processing. In our software product the database consists in a set of files (text or binary files). The relevant attribute of the database is the portability. For this requirement, the output data of the mesh generator are arranged in two types of the files: the first type of the files contains the geometrical properties of the mesh; another file contains the physical properties of the elements (field sources, material properties, boundary conditions etc).

The first file with geometrical properties contains the records with the following components: the number of the node, coordinates of the nodes, code for the boundary conditions. The nodes are labelled with integers ranging from 1 to the total number of nodes in the domain. The second file with geometrical properties contains element data: number of the element, the nodes of the element, label of the element, code of the edge for non-natural boundary conditions. The two files have no physical information about the domain. In this way the database is independently by the field problem.

The attributes relative to the physical problem can be assigned finally if the user desires. The process is done by the codes from the records of the geometrical files. For example, the element code is an index in a table with physical properties of the element. An alternative approach is to assign the physical properties in the assembly phase of the finite element program in processing stage. From practical considerations, all elements in a block have the same code.

The user has the access at the database and can link it with different programs of FEM or software products as Mathcad, Matlab etc. The advantage of our software consists in the possibility of the user to develop his solvers for special problems in engineering.

5 Semi-automatic facilities

The mesh generator has a lot of manual facilities for refinements. The initial mesh can be extended in the zones of the interest. Some transformations as bisection, trisection, quadri-section, symmetry etc. were included in our software product and can be used by the users in the development of the database in the manual effort involved in data preparation [5]. An automatic extension involves the design of efficient error estimators and this task is difficult. An engineer can estimate the zones with accuracy problems so that a manual extension is a good approach in the development of the final mesh.

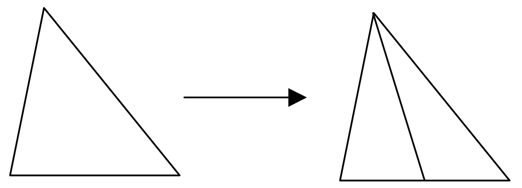

Fig. 5- The bisection

The bisection method involves the division of the triangular element in two elements using the middle of a side in an element (Fig. 5).

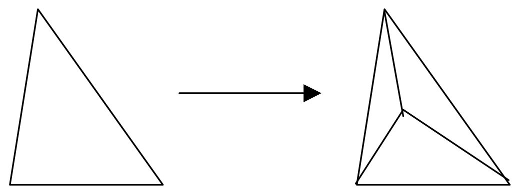

Fig. 6- The trisection

The trisection method involves the division of the triangle in three triangles (Fig.6). The new vertex is the intersection of the medians.

The quadrisection method involves the division of a triangle in four triangles by the middle lines of the selected triangle (Fig. 7).

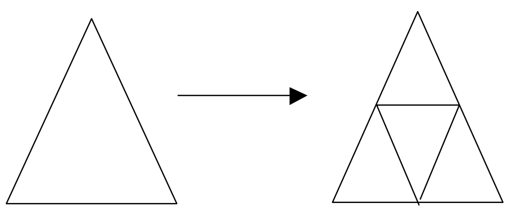

Fig. 7- The quadri-section

We considered only triangular elements but our generator can create different element types. The permitted element types are:

- Linear 3 nodded triangular elements. The procedure for mesh generation was presented in the above discussion.
- Linear 4-noded quadrilateral elements. In this case elements are formed by linearly connecting grid points.
- Quadratic 8-noded isoparametric elements. For this case each element spans over a 2 x 2 mesh of grid points.

6 Conclusions

In this work we described an efficient program for generating finite element meshes based on automatic element subdivision of a few large domains or blocks which are defined as input data. Our mesh generator can be implemented both in conventional and parallel computers. On parallel computers the speedup of the computation can be improved. The FE mesh itself can be generated in parallel, with each processor generating a portion of the mesh, which it was allocated during the partitioning phase. Refinement techniques can be used if it is necessary.

The conventional mesh generator can be used both in the case of convex domains and for domains with shapes not too different from a convex. In "twisted" geometries a suitable transport function must be defined and it is difficult to find a transport function for non-convex domains [3].

The mesh generator developed by the author is easy to use. The first version of this mesh editor was a semi-automatic generator [5]. We can control the density of elements in a given region. As input data, our mesh generator requires the input of a coarse discretization of the domain in terms of blocks of quadrilateral nature and, in addition, a distribution on the edges of the blocks. The number of the points must be consistent (i.e., two logically connected edges must have the same number of control points). The control points can be generated automatically in the case of "straight edges" but for "curved edges" the control points are given explicitly in order to describe the shape of the "curve".

In an adaptive mesh the following strategies can be used: h-strategy (when elements are refined); p-strategy (when the order of the polynomial approximations of the unknowns is increased); h-p strategy (a combination of the previous strategies).

References:

[1]. Cârstea, D. "CAD tools for magneto-thermal and electric-thermal coupled fields". *Research Report in a CNR-NATO Grant.* University of Trento. Italy, 2004.

[2]. Hinton, E., Owen, D.R., *An introduction to finite element computations*. Academic Press, New York, San Francisco, 1977, US.

[3]. George, P.L., *Automatic Mesh Generation. Application to finite element methods*. John Willey & Sons, Paris 1991.

[4]. Olaru, V., Brătianu, C., *Modelarea numerică cu elemente finite.* Editura tehnică, Bucureşti, 1986, Romania.

[5]. Cârstea, D., Cârstea, I., "A semi-automatic mesh editor for finite element programs". In: *Proceedings of the Second International Workshop CAD in electromagnetism and electrical circuits, CADEMEC 99.* pg. 138-141. 7-9 September 1999, Cluj, Romania, 1999