Organization of the Mesh Structure

Tomasz Jurczyk and Barbara Głut

AGH University of Science and Technology, Cracow, Poland,
{jurczyk,glut}@uci.agh.edu.pl

Abstract. The subject of this article concerns some aspects of the generation of 3D surface meshes, widely used in many scientific applications. This work contains a description of data structure used in the developed mesh generator and issues related to the management of this structure. The proper organization of this structure coupled with the appropriately selected algorithms, allow to achieve the satisfactory efficiency of the mesh generation process.

1 Introduction

In many areas such as numerical computation or computer graphics there are required surface meshes of three-dimensional objects. The developed mesh generator can create unstructured triangular, quadrangular or mixed meshes, basing on the geometrical description of the given objects. The quadrangular meshes are created from the triangular ones via the indirect techniques[1].

When creating meshes containing very large number of elements, the computational complexity and the memory requirements of the generator can prove to be a very important factor. Several data structures to represent and manipulate unstructured meshes have been published over the years [2] [3]. Such structures usually offer efficient solutions to represent and recover mesh topology and modify its geometric parameters. In this article we describe some aspects regarding the organization of the unstructured meshes, which allows obtaining practically linear time complexity of finite mesh generation.

2 Mesh Structure and Management

Finite element mesh is represented as a structure of linked entities such as vertices, edges or elements (triangles or quadrangles). The selection of the represented mesh entities and theirs interconnection are motivated by the diversity of employed procedures, which can treat the mesh as a graph of vertices and edges, a graph of connected elements, etc.

The process of finite mesh generation consists of a sequence of mesh transformation (mostly local), associated with such operations as insertion of the new node into the triangulation, or with conversion of two triangles into a single quadrilateral element. Depending upon the phase of the generation process, the

number of various mesh entities may gradually increase, decrease or be practically constant. The typical character of quantity changes for various mesh entities in different phases of the generation process (triangulation, conversion to quadrilaterals, mesh improvement) presents Fig. 1.
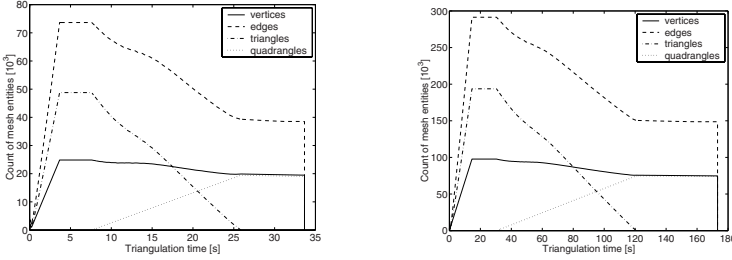

Fig. 1. Number entities during the process of mesh generation (mesh with NT=49k and NT=194k)

In order to optimize algorithms which relies on the removing/inserting of mesh entites (e.g. edges or triangles), there is being used an adequately adjusted procedure of allocation and deallocation of mesh elements. Instead of many single memory operations, a set of entities is allocated at once. Additionally, during the routines of mesh generation, where the quantity of elements has an inclination to grow, removed entities are not deallocated, but they are put aside for further use.

3   Prediction of the Mesh Size

In general case, the final size of the mesh isn't known a priori. However, such information can be very useful for precise establishing of the sizes of the dynamically created structures used for storing the mesh entities, thus increasing the efficiency of memory management – and the whole generation procedure. In the presented approach we try to assess the final size of the finite element mesh during the process of its creation, basing upon the property of the incremental Delaunay triangulation algorithm. During this procedure, all triangular elements are organized in the heap-structure, according to some quality criterion (the root of the heap contains the element with the lowest quality)[4]. The selected quality criterion is responsible for assessing how good the given triangle conforms to the prescribed control space (according both to geometrical shape and element size). It can be evaluated from the following formula:

$$Q_\triangle = \frac{S_\triangle}{\frac{3\sqrt{3}}{4\pi}S_\circ} \tag{1}$$

where $S_\triangle$ is the area of the ideal triangle (according to the current metric), and $S_\circ$ is the area of the circle circumscribed on the evaluated triangle.

At the subsequent steps of the triangulation algorithm, elements with the worst quality (i.e. too large or badly shaped) are taken from the root of the heap structure. For each taken triangle a refinement procedure is performed, which consists of insertion of a new node in the circumcentre of the triangle and local reconfiguration of the mesh according to the selected Delaunay criterion. The quality criterion (1) is mainly used for determining the order of triangles, selected for the refinement operation. However, this formula can be additionally used for predicting the final number of triangles in the mesh being created. Since this criterion always selects elements with the highest ratio of the current size to the expected size, the process of mesh refinement (according to the prescribed control space) is performed gradually throughout the whole mesh area.

The prediction of the final size of the mesh is calculated from the current number of elements in the mesh and the assessed number of triangles, which should be created instead of some average triangle from the current mesh:

$$NT_{\text{final}} = NT_{current} \frac{Q^*}{Q_\triangle} \tag{2}$$

where NT_{final} is the predicted number of triangles, $NT_{current}$ is the current number of triangles in the mesh, Q^* is the quality coefficient for the selected (average) triangle, and $Q_\triangle$ is the quality coefficient of the ideal triangle.

One of the most important factors influencing the accuracy of the prediction of the final number of elements in the mesh is the selection of triangle (or triangles), for which the quality criterion is inspected and used in the formula (2). Because of the efficiency aspects, the number of inspected triangles can't be too large – the sampling should be restricted to one or few elements.
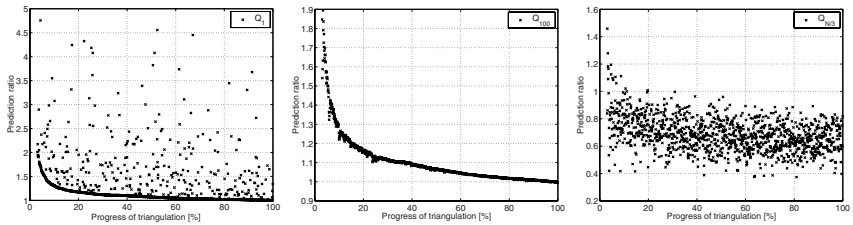

Fig. 2. Accuracy of prediction (ratio of assessed final number of elements to the real value) against the triangulation progress, for cases Q_1, Q_{100}, $Q_{\frac{N}{3}}$

An influence of the selection of the sampled triangle on the prediction accuracy was tested for several meshes. Figure 2 presents the prediction accuracy, evaluated as the ratio of the assessed final number of triangles to the real mesh size after the triangulation process. The best results of prediction were obtained using triangle near the root of the heap (the example triangle Q_{100} is taken from the sixth level of the heap), but not too close. Oscillations for the elements
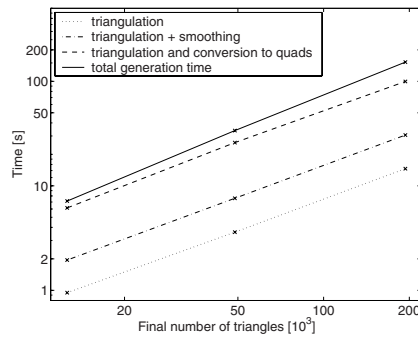
Fig. 3. Computational complexity for meshes with the final number of triangles: NT=12580, NT=48796, and NT=193672

Q_1 (taken from the root of the heap) and $Q_{\frac{N}{3}}$ follow from the variation of the quality of elements at these places.

4   Computational Complexity

The presented structure of the mesh coupled with the appropriately selected algorithms, allow to achieve practically linear computational complexity of the mesh generation. Figure 3 presents the time of generation of three meshes with sizes from 12.5 to 200 thousand of triangles.

References

1. T. Jurczyk and B. Głut. Generation of Good Quality Quadrilateral Elements for Anisotropic Surface Meshes. In N.-E. Wiberg and P. Diez, editors, Proc. of the ADMOS2003 Conference on Adaptive Modeling and Simulation, Göteborg, Sweden, 29 September - 1 October 2003. Chalmers University of Technology, Sweden.
2. M.W. Beall and M.S. Shephard. A general topology-based mesh data structure. International Journal for Numerical Methods in Engineering, John Wiley & Sons, Ltd., 40:1573–1596, 1997.
3. R.V. Garimella. Mesh data structure selection for mesh generation and FEA applications. International Journal for Numerical Methods in Engineering, John Wiley & Sons, Ltd., 55(4):451–478, 2002.
4. T. Jurczyk and B. Głut. Generation of triangular meshes for complex domains. Computer Science, Annual of University of Mining and Metallurgy, Cracow, 3:71-93, 2001.