Domain Decomposition Coupled with Delaunay Mesh Generation

Tomasz Jurczyk[1] and Barbara Głut[1]

University of Mining and Metallurgy, Cracow, Poland,
{jurczyk,glut}@uci.agh.edu.pl

Abstract. This paper presents a domain decomposition method for finite element meshes created using the Delaunay criterion. The decomposition is performed at the intermediate stage of the generation process and is successively refined as the number of nodes in the mesh increases. The efficiency of the algorithm and the quality of the partitioning is evaluated for different partition heuristics and various finite element meshes.

1 Introduction

The mesh partitioning problem is to partition the vertices (or elements) of a mesh in k roughly equal partitions, such that the communication overhead during the parallel simulation is minimized. The quality of the partition is a function of several criteria, depending upon the computer architecture and the problem being computed. In most cases, the cut-size of the partition (the number of edges connecting vertices in different clusters) is assumed as the main quality coefficient. The graph partitioning problem is NP-complete (e.g. [1]), regardless of the number of partitions. However, there have been developed many algorithms calculating reasonably good partitionings[2, 3].

Recently a class of multilevel algorithms[4, 5] has emerged as a highly effective method for computing a partitioning of a graph. The basic structure of a multilevel partitioning algorithm consist of three phases: coarsening, initial partitioning and refinement. In the first phase the multilevel algorithm repeatedly reduces the size of the graph by joining pairs of adjacent nodes. At each step the *maximum matching* is found by applying some strategy of edge collapsing, the graph is reduced and the weights of nodes and edges are updated. The second phase partitions the reduced graph into required number of clusters. Since the number of nodes is small, the partitioning can be effectively computed. The only requirement for the partitioning algorithm is to be able to handle graphs with weighted nodes and edges. In the third step the graph is repeatedly uncoarsened until it reaches the original size. Because of the weighted nodes and edges, the balanced partition of the smaller graph corresponds to a balanced partition of the larger graph with the same cut size. At each step of uncoarsening the graph can be further improved by local refinement heuristics.

In this article we present a similar method applied to the decomposition of finite element meshes generated with the Delaunay triangulation. In the process

P.M.A. Sloot et al. (Eds.): ICCS 2002, LNCS 2329, pp. 353–360, 2002.

of triangulation using the Delaunay criterion the mesh is created by an incremental refinement of initial simple triangulation. First phase consists of successive insertion of all boundary nodes. Then, inner nodes are added until the quality measure (e.g. size or geometric quality) for all triangles in the mesh is satisfactory. Inner nodes are inserted in the currently worst areas of the triangulation, which results in the uniform refinement of the mesh. In the presented method, instead of coarsening the complete mesh, an intermediate mesh is selected as a coarse graph and is initially partitioned into the required number of clusters. The process of the further construction of the mesh is similar to the uncoarsening stage in the multilevel method. Each newly inserted node becomes a part of some cluster, depending upon the partition numbers of adjacent nodes. Periodically, after a specified number of nodes are included into the mesh, a smoothing process is started. The smoothing method should be fast and improve the quality of the partitioning locally.

The remainder of the paper is organized as follows. Section 2 defines the graph partitioning problem and presents shortly the most important heuristics for both the partitioning and smoothing task. Section 3 presents the description of the domain decomposition during the Delaunay triangulation. In section 4 various parameters of the presented method are evaluated.

2 Graph Partitioning

Given a graph $G = (V, E)$ with nodes V ($n = |V|$ is the number of nodes) and undirected edges E, let $\pi : V \to \{0, 1, \ldots k-1\}$ be a *partition* of a graph G, that distributes the nodes among k clusters $V_0, V_1, \ldots V_{k-1}$. The *balance* of π is defined as follows: $bal(\pi) := \max |V_i|; 0 \leq i < k - \min |V_j|; 0 \leq j < k$; and the *cut-size*: $cut(\pi) := |\{\{v, w\} \in E; \pi(v) \neq \pi(w)\}|$.

2.1 Decomposition Algorithms

There are several different criteria a heuristic can focus on. Many methods are very powerful with regard to one criterion, but don't satisfy others. The following description summarizes the most popular schemes.

The *greedy* strategy[6] starts with one cluster holding the whole graph, others being empty. Then, nodes are repeatedly moved to the successive underweighted clusters in such way, that each movement minimally increases the cut size. If the number of clusters $k > 2$ both direct approach and recursive bisection can be applied.

The *layer* method[7] starts with a random initial node and repeatedly identifies successive layers of adjacent nodes. After all nodes are visited, the graph is accordingly split with the first cluster containing $\frac{n}{2}$ nodes from successive layers.

In the simple *geometric* bisection the longest expansion of any dimension is determined and the nodes are split according to their coordinate in this dimension. The *inertial* method[8] considers nodes as mass points and cuts the graph with the line (plane) orthogonal to the principle axis of this collection of mass

points. In the *shortest cut* strategy several lines (planes) containing the mass center of the graph are created. All edges of the graph are inspected, the *cut size* for each of these lines (planes) is evaluated, and the line (plane) with the smallest number of crossing edges is selected.

The *spectral* bisection heuristic[7, 8] is based on algebraic graph theory. The *Laplacian matrix* is constructed on the basis of the adjacency information of the graph and the eigenvector $\bar{y}$ of the second smallest eigenvalue of this matrix is used to bisect the graph. This method gives good results, unfortunately the time and space required to compute accurately the eigenvalues and eigenvectors of the *Laplacian matrix* is very high.

The *multilevel* class[4, 5]) of graph partitioning methods employs the reduction of large graphs both to shorten the partitioning time and to improve the solution quality. First the graph is successively reduced by collapsing vertices and edges, the smaller graph is partitioned and finally uncoarsened level by level with addition of local refinement.

2.2 Refinement Algorithms

The initial (global) partitioning can be further improved by applying some local smoothing method, which concentrates on the refining of the current partitioning in order to obtain a lower *cut size* (or to improve others desirable properties).

The *Kernighan-Lin* heuristic[9, 10] is the most frequently used local refinement method. Originally, it uses a sequence of node pair exchanges to determine the changes in the initial partitioning. After the given number of steps is executed, the history of changes is checked and the step with the best incremental gain is selected. The whole procedure continues until there is no improvement. Several improvements and generalization were proposed since the original publication of this algorithm such as moving single nodes, limiting the maximum number of moved nodes in a single pass, using cost metrics other than cut-size, refining k-way partition.

3 Simultaneous Mesh Generation and Decomposition

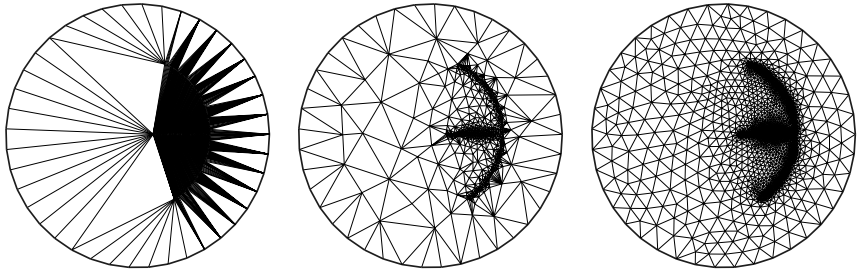
The preparation stage of the parallel numerical simulation of physical processes using finite element method consists usually of several phases: definition of the process domain, generation of the finite element mesh, decomposition of this mesh in the given number of clusters and mapping these clusters to the processor-nodes of the parallel machine. In the approach presented in this paper, the mesh generation procedure (based on the Delaunay property) is coupled with the mesh decomposition. By running the partitioning procedure at an early phase of the mesh construction, both time complexity and the quality of the overall process can be improved. It should be noted, that addition of the decomposition procedures doesn't influence the geometrical structure of the final finite element mesh.

The domain decomposition can be incorporated into the mesh generation scheme as follows:

```
[First stage]
 Create simple mesh (two triangles) containing all boundary nodes.
 Insert all boundary nodes into the mesh
   (respecting the Delaunay property).
[Second stage]
 Evaluate the quality coefficients for all triangles.
 While the worst triangle is not good enough:
   insert a new node inside this triangle,
   retriangulate the mesh locally.
   (*) if(node_count == N1) initial decomposition
   (*) else if(node_count == N1 + k*N2) decomposition smoothing
```

The initial decomposition is performed when the number of nodes (both boundary and inner ones) reaches N1. After this decomposition the smoothing procedure is called in regular intervals (every N2 number of inserted nodes).

In Fig. 1 there are presented successive steps of the mesh generation process using the Delaunay property. Figure 1a shows the mesh containing boundary nodes only. Figures 1b,c present intermediate mesh with a number of inner nodes already inserted and finally the complete mesh.



(a) Triangulation of boundary nodes (b) Intermediate step (c) Final mesh

Fig. 1. Successive steps of Delaunay mesh generation

3.1 Initial Decomposition

The decomposition of the mesh can take place at any time during the second phase of the Delaunay triangulation algorithm, while the mesh is being refined by insertion of inner nodes. At the beginning of this phase the mesh may substantially differ from the final mesh. As the refinement of the mesh progresses, the mesh structure stabilizes. The initial partitioning of the mesh should be performed when the overall variance of the mesh density is reasonably uniform. The

"resemblance" of the intermediate mesh to the final form is increasing with each inserted inner node. Unfortunately, so does the time required for the partitioning process.

The choice of the step when the initial partitioning is run (the parameter N1) influences noticeably the final quality of the mesh decomposition. Partitioning of the mesh at an early stage allows to concentrate on the main characteristics of the mesh structure. Unfortunately, if the final mesh density is variegated, splitting the mesh too soon can result in highly unbalanced clusters.

The selection of the partitioning heuristic is arbitrary (e.g. any method described in the section 2.1 can be used). The obtained partitioning, however, should be of good quality (with connected clusters), so that any further smoothing would adjust the boundaries of clusters locally only.

3.2 Decomposition Smoothing

After the initial partitioning of the mesh, the generation process is continued. As the number of inserted nodes increases, the structure of the mesh can alter and an adequate adjustment of the decomposition separators should be performed. Additionally, finer mesh has more degrees of freedom, and the quality of decomposition can be further improved.

In order to meet these needs, a smoothing method is periodically executed (every N2 inserted nodes and when the mesh generation process ends). A class of local refinement algorithms that tend to produce good results are those based on the Kernighan-Lin algorithm. In the variation used in this work the vertices are incrementally swapped among partitions to reduce the edge-cut and balance of the partitioning.

The selected smoothing method influences greatly both the quality of the obtained decomposition and the running time of the whole process. Both frequency of smoothing and the complexity of the method itself should be carefully selected. The complexity aspect is especially important as the number of partitions increases, which makes the complexity of this method much higher.

3.3 Introduction of Inner Nodes

During the continued process of the generation of a preliminary partitioned mesh, all newly inserted nodes must be assigned to the proper cluster. The decision is based on the nodes adjacent to the node in question. If all these nodes belong to a single cluster (Fig. 2a) the new node is assigned to the same cluster. However, if there are more available clusters (Fig. 2b,c) several criteria can be considered:

- *cut-size* – cluster with the most number of edges incident to this node should be selected.
- *balance* – prefers the smallest of incident clusters.
- other quality properties (e.g. avoiding nodes adjacent to several different clusters).

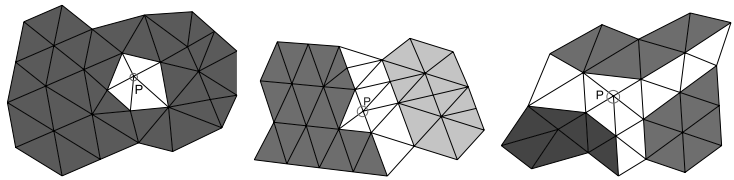


Fig. 2. Identification of the inner nodes

4 Results

The described method was tested on several domains with various control spaces, which resulted in generation of meshes with different shape and the variation of the density of elements. All meshes had roughly equal number of nodes and were partitioned into eight clusters. In [11] these meshes are presented along with a detailed analysis (including several aspects of the decomposition quality) of the results obtained with different partitioning algorithms.

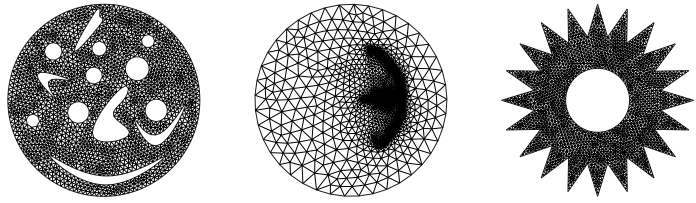
In this work we tested the efficiency of the proposed algorithm and the influence of different values of parameters N1 and N2 (along with other parameters of the decomposition process). The results were evaluated using several quality coefficients such as cut-size, balance, boundary balance, local cut-size, connectivity, multiconnected nodes, and running time.

Table 1 presents the results obtained for a three selected meshes (Fig. 3) containing about 8000 nodes partitioned into 8 clusters. The cut-size was chosen as a representative coefficient of the decomposition quality. The Roman numbers denote partitioning algorithms used to perform the initial decomposition: (I) direct greedy, (II) recursive greedy, (III) recursive layer, (IV) recursive inertial, and (V) recursive shortest-cut. In each case the partitioning was further improved by a Kernighan-Lin smoothing method.

Table 1. Cut-size of meshes decomposed into 8 clusters with different methods (P0 denotes the decomposition of the final mesh, P1 – simultaneous mesh generation and decomposition with the parameters N1=1000, N2=1000, and P2 – with the parameters N1=1500, N2=500)

	cheese			mountain			gear		
	P0	P1	P2	P0	P1	P2	P0	P1	P2
I	659	769	596	804	868	834	599	805	685
II	632	647	610	819	808	795	553	595	650
III	542	572	513	915	782	674	519	855	590
IV	588	607	701	920	717	711	501	625	525
V	471	613	522	699	694	662	404	455	432

Coefficients presented in table 1 visualize the typical characteristic of obtained results. Depending upon the various parameters of the overall mesh generation and decomposition process, different variants gain prevalence. In the discussed example, the results were in most cases better when the value of parameter N1 (denoting the step at which the initial partitioning is performed) was increased from 1000 to 1500, and the frequency of smoothing (N2) was decreased from 1000 to 500. For different meshes the quality of decomposition calculated during the mesh generation as compared to the partitioning of the complete mesh has no regular tendency. E.g. the partitioning during the generation of the mesh (P2) is usually better than the partitioning of the final mesh (P0) in case of the mesh `mountain`, and consistently worse for the mesh `gear`.



(a) cheese (b) mountain (c) gear

Fig. 3. Tested meshes

Fig. 4 shows the intermediate phases and the final result of the simultaneous mesh generation and partitioning. As can be seen, the final partitioning is much alike the initial one. This property is very advantageous, especially with respect to the task of a parallel mesh generation, when the partitions should be preliminary distributed to the proper processors of a parallel machine and any further movement of these nodes results in the degradation of the overall performance.

(a) Initial phase (1500 nodes) (b) Intermediate step (5000 nodes) (c) Final result (8362 nodes)

Fig. 4. Example of generation and decomposition process (N1=1500, N2=500).

5 Summary

In our work we tested and compared the efficiency and quality of the decomposition of various algorithms applied to variegated finite element meshes. The experiments with the presented approach of combined mesh generation and partitioning have shown that these algorithms work quite well for many different meshes. The main requirements needed for a successful employment of this scheme are a good, global initial partitioning algorithm and fast locally smoothing method.

The future research will concentrate on improving the mesh refinement procedure in order to make it more efficient in terms of both the decomposition quality and the running time. The aspects of the presented approach connected with the problem of a parallel mesh generation will also be addressed.

Acknowledgments. Support for this work was provided by the Polish Committee for Scientific Research (KBN) Grant No. 8 T11F 01216.

References

1. T. Lengauer. *Combinatorial Algorithms for Integrated Circuit Layout*. B.G. Teubner, 1990.
2. R.A. Davey. *Decomposition of Unstructured Meshes for Efficient Parallel Computation*. PhD thesis, University of Edinburgh, 1997.
3. K. Schlogel, G. Karypis, and V. Kumar. Graph partitioning for high-performance scientific simulations. In J. Dongarra and et. al., editors, *CRPC Parallel Computing Handbook*. Morgan Kaufmann, 2000.
4. B. Hendrickson and R. Leland. A multilevel algorithm for partitioning graphs. Technical Report SAND93-0074, Sandia National Laboratories, Albuquerque, New Mexico 87185, 1993.
5. G. Karypis and V. Kumar. Multilevel k-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed Computing*, 48(1):96–129, 1998.
6. C. Farhat. A simple and efficient automatic FEM domain decomposer. *Computers and Structures*, 28(5):579–602, 1988.
7. H.D. Simon. Partitioning of unstructured problems for parallel processing. *Computing Systems in Engineering*, 2(2/3):135–148, 1991.
8. R.D. Williams. Performance of dynamic load balancing algorithms for unstructured mesh calculations. *Concurrency*, 3:457–481, 1991.
9. B.W. Kernighan and S. Lin. An effective heuristic procedure for partitioning graphs. *The Bell Systems Technical Journal*, pages 291–308, Feb. 1970.
10. C.M. Fiduccia and R.M. Mattheyses. A linear-time heuristic for improving network partitions. In *Proc. of the 19th IEEE Design Automation Conference*, pages 175–181, 1982.
11. T. Jurczyk, B. Głut, and J. Kitowski. An empirical comparison of decomposition algorithms for complex finite element meshes. Presented at Fourth International Conference on Parallel Processing and Applied Mathematics, Nałeczów, 2001.