Layer-by-layer ordering in parallel finite element composition on shared-memory multiprocessors

A K Novikov, N K Piminova, S P Kopysov and YA Sagdeeva

Institute of Mechanics, Ural Branch of the Russian Academy of Sciences, 34
T. Baramzinoy, Izhevsk, Russia 426067

E-mail: sc_work@mail.ru

Abstract. In this paper, we present new partitioning algorithms for unstructured meshes that prevent conflicts during parallel assembling of FEM matrices and vectors in shared memory. These algorithms use a ratio which we introduce to determine if any two mesh cells are adjacent. This adjacency ratio defines mesh layers, which are combined into domains and assigned to different parallel processes/threads. The proposed partitioning algorithms are compared with the existing algorithms on quasi-structured and unstructured meshes by the number of potential conflicts and by the load imbalance.

1. Introduction

Modern parallel computations are performed on multi-core processors. At the same time, there is an intensive use of shared memory, which imposes additional restrictions on the mesh partitioning and mesh data related to localization and access synchronization. If shared memory is used, than a bottleneck of parallel algorithms for numerical methods are assembly operations (summation).

The finite element method such operations are: assembly of global matrices and / or vectors (in the element-by-element schemes) from local matrices and vectors, respectively, the numerical integration of local matrices and vectors. During the assembly operation of the vector components or matrix entries are summed, corresponding to the vertices shared to several finite elements. In parallel assembly, simultaneous summation vector components or matrix entries for several processes / threads leads to conflicts in the shared memory, and as a result — computational errors.

To eliminate these errors, different approaches are applied: at the level of technology of parallel programming — atomic operations and critical sections, memory allocation for storing shared data [1]; algorithm level — the change element-by-element assembly scheme at nodal [2] and the introduction of additional operations — the use of different variants for ordering mesh [3]. An ordering of mesh cells are usually carried out on the basis of the graphs coloring, which is traditionally used for the ordering of unknowns in parallel solving systems of algebraic equations [4].

Coloring algorithms allow to obtain sets of unconnected mesh cells, but are NP-complexity, since they depend not only on the number of graph vertices, but also on their degrees, and the choice of the initial vertex coloring.

In order to obtain a uniform distribution of mesh cells between parallel processes/threads after cells colorization balancing procedure is necessary, and reordering the mesh for access to the memory location.

An ordering of mesh and assembly are considered as interdependent parts of the finite element composition and use variations of layer-by-layer mesh partitioning, whereby during assembly operations parallel computing processes are not treated at the same time to the finite elements containing shared vertices.

Variants of this partition based on mesh partitioning at finite elements layers with the thick in one cell and then combining these layers into domains of finite elements assigned one thread or processor core. Layers of finite elements are defined by using introduced adjacency ratio of cells by the nodes. The algorithm of layers formation is NP-complexity.

In this paper we investigate the variants of layer ordering unstructured meshes with cells of arbitrary shape. The resulting ordering is compared with known algorithms on quasi-structured and unstructured meshes by the number of potential conflicts in the assembly and the distribution uniformity of mesh cells on the mesh subdomains.

2. Layer-by-layer partitioning of finite element meshes

Finite element meshes are unstructured meshes, that is the number of a node (cell) cannot be defined by incrementing the number of his neighbor node (cell). In [5] the following adjacency ratio was introduced for element-by-element schemes of FEM

$$\omega_k \in Adj(\omega_j), \qquad if\ V(\omega_j) \cap V(\omega_k) \neq 0, \tag{1}$$

where ω_j, ω_k are the cells of the mesh Ω; $Adj(\cdot)$ is the adjacency operator; $V(\omega_j)$ and $V(\omega_k)$ are the subsets of mesh nodes that correspond to the cells ω_j and ω_k.

It is important to note that for this adjacency ratio, the cell has a large number of adjacent cells m_e, even for a quasi-structured three-dimensional mesh as shown in figure 1.

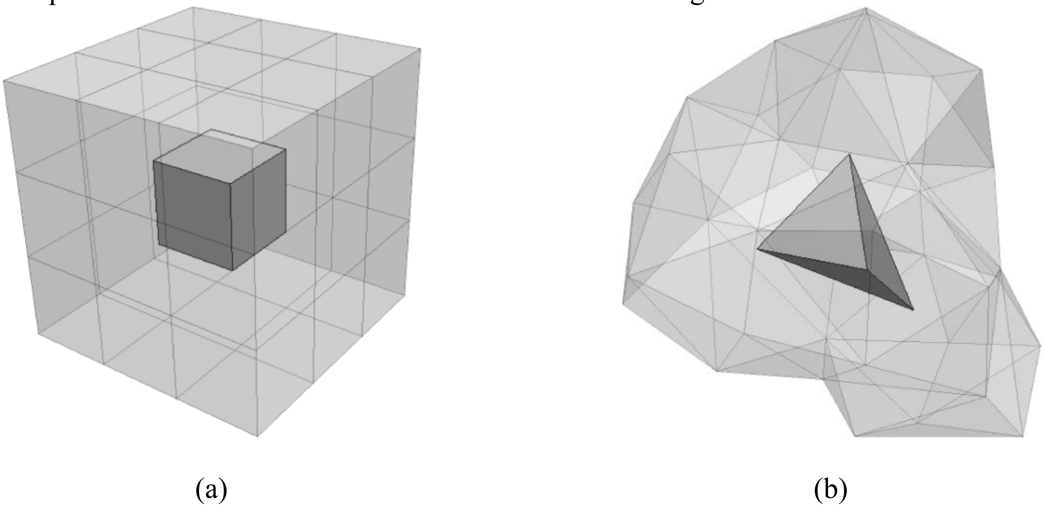

(a) (b)

Figure 1. Adjacent cells: (a) quasi-structured mesh, $m_e = 26$; (b) unstructured mesh, $m_e = 119$.

Thus, the algorithm of layer-by-layer partitioning of a finite-element mesh consists of two steps: 1) forming layers of mesh cells; 2) combining layers of mesh in subsets. Let us discuss these steps in more detail.

3. Forming layers of mesh cells

To form mesh layers by (1) the nodes are checked to belong to finite elements. Generally, unstructured meshes are stored so that the cell contains a mesh node, not the node belongs to a cell. To check which finite elements the node belongs to, two loops over mesh cells are need for $O(2 \cdot m \cdot n_e)$ operations, where m — the number of finite elements; n_e — the number of nodes in the final element.

The algorithm for generating the mesh layers [5] can be represented in the form:

(i) Set the initial layer $s_1 = \left\{\omega_j^{(i)}\right\}, j = 1,2,\dots,m_1$, where m_1 — number of cells in s_1.

(ii) Layers $s_i, i = 2,3,\dots,n_s$ determined by the expression

$$s_i = \begin{cases} Adj(s_{i-1}), & i = 2; \\ Adj(s_{i-1}) \setminus s_{i-2,} & i > 2, \end{cases}$$

where $s_i = \left\{\omega_j^{(i)}\right\}$ — a plurality of cells (finite element) in the layer i; $\Omega_{\mathrm{j}}^{(\mathrm{i})}$ — cell with number j in the layer i; $Adj(s_i) = \bigcup_{j=1}^{m_i} Adj\left(\omega_j^{(i)}\right)$ — a set of cells adjacent to s_i; m_i — number of cells in s_i. This algorithm is also applied to the separation of disconnected subsets of cells within a layer and to the search of discontinuous layers.

If in the element-by-element schemes FEM, the existence of discontinuous layers is not essential for the method of Schur complement [6] is necessary to ensure not only the continuity of layers, but also the cells adjacency by faces. Search cells having one common vertex, and the definition of discontinuous sub-layers are carried out on the basis of the neighborhood along the faces of the cell, which is used in the determination of the previously built neighborhood on the mesh nodes (1). For each cell are considered neighbors by vertex, and excluded those with which the number of vertices in common is not equal to the number of vertices on the same face n_f (three — for tetrahedral cells and four — for hexahedral).

Determination of discontinuous layers on faces neighborhood viewpoint is as follows. Take an arbitrary cell in the current layer s_i and start searching across neighboring cells. After searching we check that all the cells were in the set of reachable vertices. If not, then all the cells trapped in a set of cells are placed in the sublayer s_j^* and the new search begins. Thus is formed a set of sublayers $\left\{s_j^*\right\}$ for a discontinuous layer s_i.

To exclude cells having one common vertex and to merge them with some layer s_i we check if this layer is discontinuous by faces. If layer has more than one sublayer, we look for the sublayer that has maximum number of cells. This sublayer is considered to be current layer s_i, and other sublayers merge with the previous layer s_{i-1}.

Examples of the layers for quasi-structured (fourth part of the membrane) and unstructured mesh (frame construction) are presented in the figures 2a and 2b, respectively.

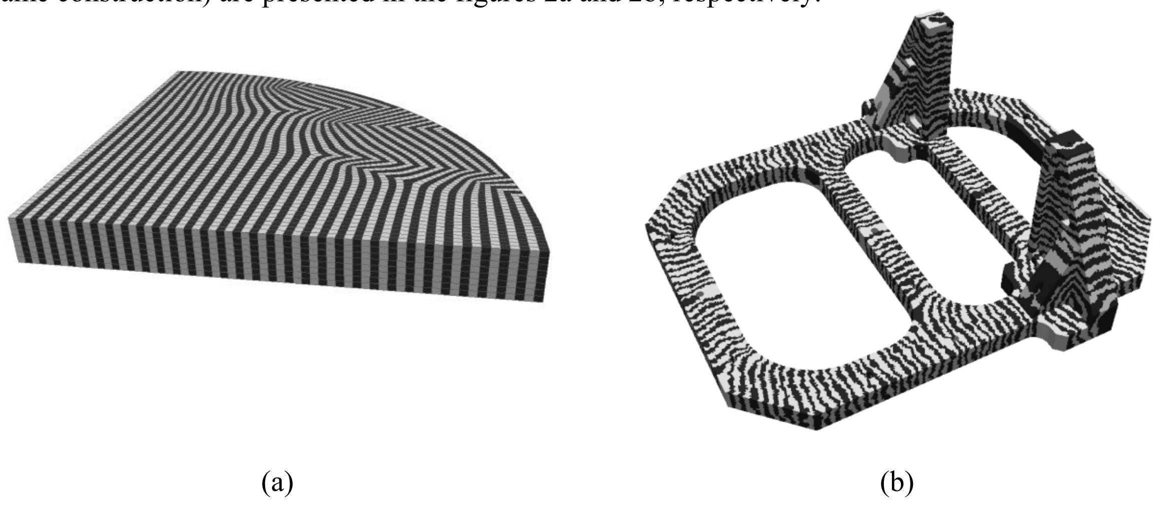


Figure 2. Layers for meshes: (a) mesh of the hexagonal cells, $m = 31744$, $n_s = 64$; (b) mesh of the tetrahedral cells, $m = 485843$, $n_s = 137$.

In the case of quasi-structured mesh (variant (x, y, z_{min})) received thirty two perfectly balanced layers: $m_{max} = m_{min} = 15872$ cells. For unstructured mesh (variant (x, y_{min}, z)) the maximum number of cells in a layer — 46635 for $m = 485843$, and minimum — 17 cells, the number of layers $n_s = 47$. These variants are marked by a choice of the initial layer s_1. For example, variant (x_{min}, y, z) means that at least one vertex in a cell of the layer s_1 has the abscissa equal to x_{min}.

Further, the layers and their order are used in the formation of the mesh subdomains.

4. Combining layers of mesh in subdomains

Several variants for the formation of the subdomain $\Omega_{\mathrm{i}}, \mathrm{i} = 1,2,\dots,\mathrm{n}_\omega$ layers from $s_j, j = 1,2,\dots,n_s$ unstructured mesh Ω was considered: block; by parity numbers of layers and their modifications.

Block variant (“block”) is as follows: Combining of layers $\Omega = \bigcup_{i=1}^{n_s} s_i$; partitioning of Ω on a subdomains Ω_i, such that $|\Omega_i| \approx m / n_\omega, i = 1,2,\dots,n_\omega$, where n_ω — the number of subsets of cells or mesh subdomains (figure 3).

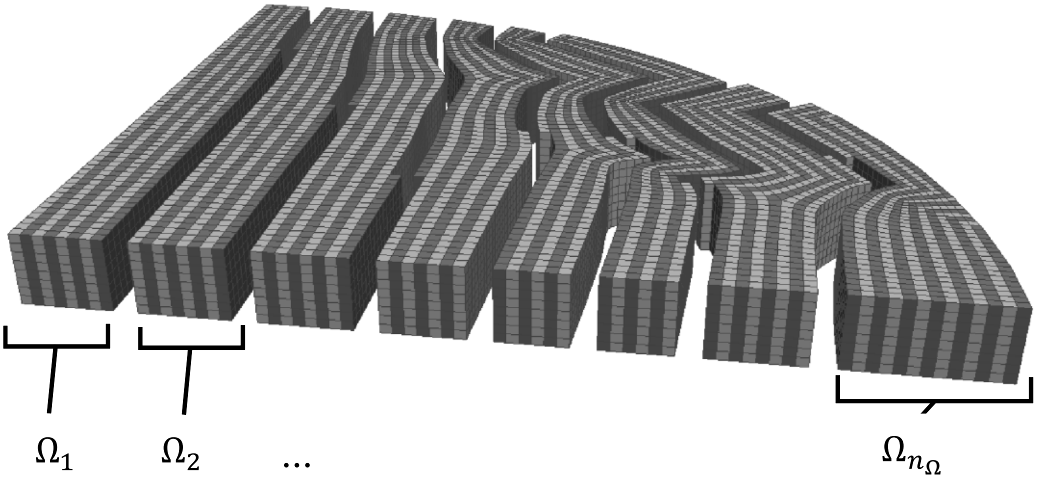
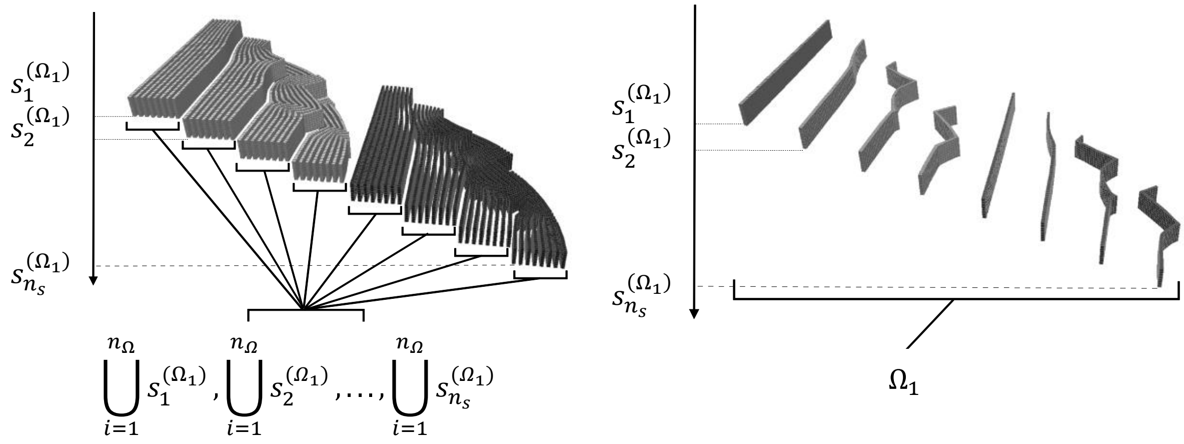


Figure 3. Subdomains are obtained in block form of the construction

As shown in figure 3, in block form, layers of unstructured mesh partitioned into parts belonging to two adjacent subdomains. For example, the subdomain Ω_1 includes seven layers $s_1, s_2, \dots, s_7$ and most of the layer s_8. Due to the division of the layers, subdomains are obtained equal by the number of cells and respectively load balancing on parallel computing processes is achieved in the future. However, the allocation of parts of the layers requires the control of the adjacency ratio. The block partition requires at least two layers in the subdomain to avoid computational errors related to the conflict with parallel access to data associated with mesh. Otherwise it is necessary to specify the order of cells in the layer, for example, in coordinate directions.

Combining of layers by the parity of numbers is as follows. First, in a subdomain $\Omega_{\mathrm{i}}, \mathrm{i} = 1,2,\dots,\mathrm{n}_\Omega$ unite all layers with odd numbers, and then the even (figure 4a). In the subdomain ω_i is placed layers (odd-numbered or even-numbered) with the numbers $i, i + n_\Omega, i + 2n_\Omega, \dots$ for example, the subdomain Ω_1 in figure 4b.

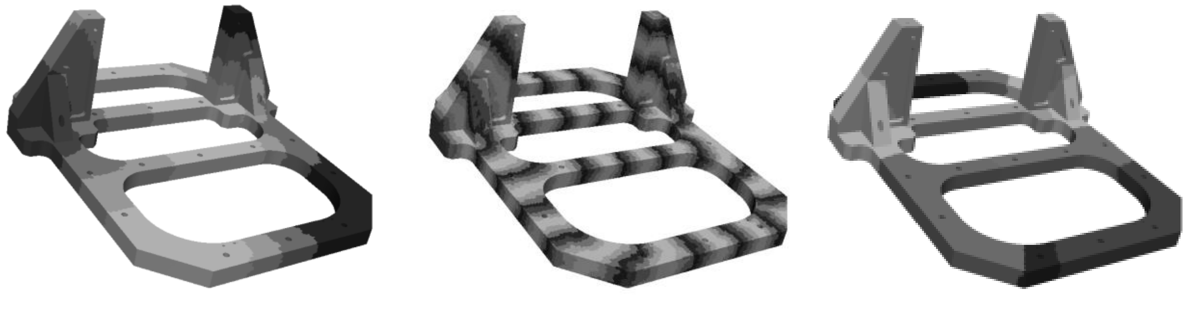

$s_1^{(\Omega_1)}$
$s_2^{(\Omega_1)}$
$s_{n_s}^{(\Omega_1)}$
$\bigcup_{i=1}^{n_\Omega} s_1^{(\Omega_1)}, \bigcup_{i=1}^{n_\Omega} s_2^{(\Omega_1)}, \dots, \bigcup_{i=1}^{n_\Omega} s_{n_s}^{(\Omega_1)}$
$s_1^{(\Omega_1)}$
$s_2^{(\Omega_1)}$
$s_{n_s}^{(\Omega_1)}$
Ω_1

(a) (b)

Figure 4. Subsets of cells are formed by parity numbers of layers and distribution of layers on subdomains Ω_i: (a) the layers belonging to the eight subdomains ($n_\Omega = 8$), which cells will be simultaneously involved in parallel computing; (b) the sequence of layers in the subdomain Ω_1.

The results for load balance are based on the even or odd layers ("even/odd") union approach was used and providing a balanced distribution mesh cells on subdomains. In this case, the layers $s_1, s_3, s_5, \dots$ are united in a subdomain $\Omega^{(even)}$("even") and layers $s_2, s_4, s_6, \dots$ — into subdomain $\Omega^{(odd)}$. Further $\Omega^{(even)}$ is partitioned on n_Ω subdomains $\Omega_i^{(even)}$, such that $\left|\Omega_i^{(even)}\right| \approx \left|\Omega^{(even)}\right| / n_\Omega$ and corresponding to $\Omega^{(odd)}$ on n_Ω subdomains $\Omega_i^{(odd)}$, for which $\left|\Omega_i^{(odd)}\right| \approx \left|\Omega^{(odd)}\right| / n_\Omega$.

On figures 5 are presented obtained mesh subdomains for $n_\Omega = 8$. As can be seen from these figures, the layer-by-layer partitionig in contrast to the k-way, it allows to order domains connections (the order of colors in figure 5). In addition, in layer-by-layer variants, nodes can belong to no more than two mesh subdomains that does not guarantee, such as multi-level algorithm from the METIS library [7] for large values of n_Ω on adaptive unstructured meshes.



(a) (b) (c)

Figure 5. The partitioning of unstructured mesh into eight subdomains, $n_\Omega = 8$: (a) in block combining; (b) by combining the parity; (c) k-way algorithm from package METIS.

Discontinuous subdomains were obtained for multiply connected domain, as in the layer-by-layer so multi-level mesh partitioning. As noted earlier, in the parallel element-by-element of finite element discontinuous mesh subdomains do not affect the stability of computational algorithms, but slow memory access. In many algorithms are need to combine parts or discontinuous subdomains maintaining the total number of subdomains, or to treat these parts as independent subdomains with an increase a total number of them. The proposed ordering of layers and subdomains simplifies the procedure of combination with the neighboring subdomains, which involved a smaller number of adjacent subdomains than in [8].

Table 1 presents potential conflicts in the assembly of vector for several variants of mesh partitioning. As an estimate of possible conflicts with the parallel implementation of the operation of finite element composition is considered the number of cells to the nodes which are accessed in the adjacent subdomains of the mesh.

In variant “by generation” used the cells order obtained by mesh generation in the mesh generator. The mesh is divided into a predetermined number of subdomains n with about an equal number of cells in the subdomains $\Omega_i \approx m \mathbin{/} n_\Omega$.

Table 1. The number of possible conflicts in the assembly of vector.

Partitioning variants						
n_Ω	“by generation”	METIS	“block”	“even/odd”		“even+odd”
				“even”	“odd”	
		Quasi-structured mesh, $m = 507904$				
8	0	0	0	0	0	0
16	0	384/0	0	0	0	0
32	0	24/0	0	0	0	0
60	0	60/2	0	0	0	0
		Unstructured mesh, $m = 485843$				
8	284/88	7/2	0	0	0	0
16	623/198	13/4	0	0	0	0
32	1144/352	60/25	0	0	0	0
60	2425/815	169/71	167/40	0	0	4/3

The numerator shows the number of cells, to nodes that are accessed from different mesh subdomains (threads), in the denominator — the number of cells in which the treatment of the different mesh subdomains happens to nodes with the same local (inside the cell) numbers. The estimate is presented in the denominator, more strict, based on the fact that the parallel processes (threads) begin at the same time, and access to each data cell occurs for equal period of time from any of the parallel process. If this estimate is different from zero (see table 1), then the parallel assembly of finite element vectors occurs to computational errors.

Errors of computing related to the conflicts in the shared memory can not be observed due to the ordering of cells, resulting in the mesh generation, as in the case of quasi-structured mesh (table 1, the second column). When unstructured mesh is partitioned by parity of layers (“even+odd”) the occurrence of conflicts is associated with an imbalance in the number of layers of cells. For such meshes access to the mesh cells and related data is carried out in a sequence of two parallel regions OpenMP (“even/odd”), separately for the even (“even”) and odd (“odd”) layers. When the block partitioning conflicts are the result of discontinuity of the layers to ensure a balanced mesh partitioning and the absence of cells ordering in layers. Combining entire layers eliminates these conflicts, but leads to a much less balanced partitioning.

Acknowledgments

The reported study was partially funded by RFBR according to the research projects №14–01–00055-a, №16–01–00129-a.

References

[1] Fuhry M, Giuliani A and Krivodonova L 2014 *International Journal for Numerical Methods in Fluids* **76** 982

[2] Kopysov S P, Kuzmin I M, Nedozhogin N S, Novikov A K, Rychkov V N, Sagdeeva Y A and Tonkov L E 2014 *Computer Research and Modeling* **6** 79

[3] Komatitsch D, Micha D and Erlebacher G 2009 *J. Parallel Distr. Comp.* **69** 451

[4] Adams L and Ortega J 1982 *Int. Conf. Parallel Processing* p 53

[5] Kopysov S, Novikov A and Piminova N 2016 *Proc. of the 10th Ann. Inter. Sc. Conf. on Parallel Comp. Tech. (PCT 2016) (Chelyabinsk, Russia: South Ural State University) p 555*

[6] Kopysov S P, Kuzmin I M, Nedozhogin N S, Novikov A K and Sagdeeva Y A 2013 *Lecture Notes in Comp. Sci.* **7979** 65

[7] Karypis G and Kumar V 1998 *SIAM Journal on Scientific Computing* A **20** 359

[8] Golovchenko E and Yakobovskiy M V 2015 *Russian Supercomputing Days: Proceedings of the Int. conf. (Moscow.: Moscow State University)* p 303