

# A PARALLEL UNSTRUCTURED MESH GENERATION ALGORITHM BASED ON THE DOMAIN DECOMPOSITION METHOD

Huai Zhang<sup>1</sup>

<sup>1</sup>*Laboratory of Computational Geodynamics, School of Earth Science, Graduate School of CAS, Chinese Academy of Sciences, No. 19 Jia, Yuquanlu, Beijing, P.R.C. 100039, hzhang@gscas.ac.cn*

## ABSTRACT

In this paper, a parallel unstructured mesh generation algorithm will be presented. At present time, most of the parallel algorithm are based on the basic idea of domain decomposition method (DDM), such as the finite element method, the finite difference and the finite volume method which are used as the most important computation methods both in scientific research and engineering fields. To achieve good parallel efficiency, reducing the communications between difference processors is one of the best ways. Unstructured mesh generation is the first step for the numerical computation, and it prepares the preprocessing data for the subsequent computation processes. So if the mesh generation area has been divided according the number of the processes of distributed memory parallel machine, then the unstructured meshes can be generated and be stored in the local memory. This method possesses two main advantages, the first one is that the unstructured meshes can be generated in parallel, the second one is that the data localization is achieved as the same time. The data communication between difference processors during the runtime will be greatly reduced.

**Keywords:** unstructured mesh generation, DDM, triangle

## 1. INTRODUCTION

Domain decomposition method (DDM) now become one of the most important algorithms in the parallel high performance computation field. As a good parallel software designer, one must take the efficiency of the parallel algorithm, the load balance among the difference processors and the data localization during the parallel computation processes into consideration.

Unstructured mesh generation is a time consuming work, it needs a great deal of computation time and large memory access. Oil reservoir simulation, the numerical simulation of the blank moving theory of the earth or the computation fluid dynamics (CFD) call for up to 1,000,000 unstructured meshes. Such large-scale computation problem can not be accomplished smoothly if the parallel machine could not be used due to the CPU's speed and the limitation of the local memory of the single computer. Many people concentrated in this research field are seeking good parallel generation algorithm for the unstructured mesh to get the higher efficiency of the unstructured mesh generation. DDM method,

which has been widely used in the computation filed, may be one of the best ways to settle this problem.

In this paper, we introduced a parallel unstructured mesh generation method based on the basic idea of domain decomposition. This kind of mesh generation method can be combined with the traditional DDM algorithm in FEM, FDM, FV, FVE etc. If they are combined together, many advantages will be obtained. First, we can deal with large-scale problems in parallel and need less memory access than the ordinary sequential codes. Secondly, we can schedule the load balance among all the processors according to the number of the unstructured mesh in the difference sub-domain, then better parallel degree will be greatly improved during the subsequential computation processes. Thirdly, the data for the unstructured meshes can be stored in local memory (hard disk) after they were generated. One of the processors take charge of the boundary mesh generation to communication between difference nodes, no superfluous communications were needed. So we get better data localization to improve the parallel efficiency.

This paper was arranged as following:

- In the first part of this paper, a brief introduction for one of the finite element DDM methods named A Larange Multiplier Based Domain Decomposition Method (LMDDM) will be given.
- In the second part of this paper, the parallel unstructured mesh generation algorithm will be presented in detail.
- In the third part of the paper, a load balance algorithm for the parallel mesh generation will be depicted.
- In the fourth part, some of the results of the parallel mesh generation and their comparison to sequential codes will be displayed.

## 2. A BRIEF INTRODUCTION TO LMDDM ALGORITHM

For the sake of simplicity, take the following equation as a model to introduce the Lagrange Multiplier Based Domain Decomposition Method (LMDDM) :

$$\begin{cases} -\Delta u + u = f, \Omega \\ u = 0, \partial\Omega \end{cases} \quad (1)$$

Here  $\Omega$  is the domain of the problem,  $\partial\Omega$  is the boundary of domain  $\Omega$ . First, divide the domain  $\Omega$  into some sub-domain  $\Omega_i (i=1,2,\dots,n)$ ,  $\Omega_i$  is quasi-uniform sub-domain with the size of  $d$ , then divide each sub-domain into finite element mesh, i.e. divide the sub-domain  $\Omega_i$  into elements of  $h_i$ , let  $\Omega^h = \cup\Omega_i^h$ ,  $\Gamma = \cup\partial\Omega_i$ . Then construct the finite element space:

$$S_{h \times H} \subset S_h^0(\Omega) \times S_H(\Gamma) \quad (2)$$

which:

$$S_H(\Gamma) = \left\{ \lambda \mid \lambda|_{\Gamma_{ij}} \in S_H(\Gamma_{ij}) \right\} \quad (3)$$

for  $h_i$  and  $d$ , take the assumption  $0 < h_i < d$ ,  $i=1,2,3,\dots,n$  and  $\lim_{h \rightarrow 0} \frac{n^2 h}{d} = 0$ , in which  $h = \max\{h_i\}$ . the weak formation of the non-compatible domain decomposition method based on the Lagrange multiplier is to get  $(u, \lambda) \in H(\Omega) \times H(\Gamma)$ , satisfy:

$$\begin{cases} \sum_i \left\{ \frac{1}{2} (\nabla u, \nabla v)_{\Omega_i} - \langle v, \lambda \rangle_{\Omega_i} \right\} = \sum_i (f, v)_{\Omega_i} \\ \sum_i \langle u, \mu \rangle_{\Omega_i} = 0 \quad \forall (v, \mu) \in H(\Omega) \times H(\Gamma) \end{cases} \quad (4)$$

where

$$\begin{aligned} H(\Omega) &= H^1(\Omega_1) \oplus H^1(\Omega_2) \oplus H^1(\Omega_3) \oplus \dots \oplus H^1(\Omega_n) \\ H(\Gamma) &= H^1(\partial\Omega_1) \oplus H^1(\partial\Omega_2) \oplus H^1(\partial\Omega_3) \oplus \dots \oplus H^1(\partial\Omega_n) \end{aligned} \quad (5)$$

To solve this problem, there are two kinds of algorithms. The first one is PCGM or other iterative methods. It is to

construct the stiffness matrix and load vector on each sub-domain of the domain  $\Omega$ , then repeatedly add the sub-domain's stiffness matrixes and load vectors together to get the global stiffness matrix and load vector of the domain  $\Omega$ , solve this problem using the Pre-conditioner Conjugate Grade Method (PCGM) or other iterative methods. This kind of method of data parallelism is the parallel category, which needs tremendous communication between computer nodes and is not suitable to speed up the parallel computation.

The second algorithm is to get the boundary variables firstly, and then get the inner variable through the values of the boundary variables. Firstly, to eliminate the inner variables through finding the minimal value on the sub-domain's boundary in order to get the capacitance matrix and boundary load vector. Secondly, to add the capacitance matrixes and boundary load vectors together. Thirdly, to get the boundary variables, back substitute the boundary variables to the sub-domain to get the inner variables. This is a kind of task parallelism algorithm. Change equation (5) into matrix format:

$$\frac{1}{2} \sum_i (U_i^T, X_i^T) \begin{pmatrix} A_i & B_i \\ B_i^T & C_i \end{pmatrix} \begin{pmatrix} U_i \\ X_i \end{pmatrix} = \sum_i (U_i^T, X_i^T) \begin{pmatrix} f_i \\ g_i \end{pmatrix} \quad (6)$$

to get the minimal value of equation (6) with regard to  $U_i$ ,

$$\sum_i (A_i U_i + B_i X_i - f_i) = 0, \quad (7)$$

$$U_i = A_i^{-1} (f_i - B_i X_i) \quad i=1,2,3,\dots,n \quad (8)$$

take equation (7) into equation (6), get:

$$\sum_i (D_i - B_i^T A_i^{-1} B_i) X_i = \sum_i (g_i - B_i^T A_i^{-1} f_i) \quad (9)$$

then we get the boundary stiffness matrix (or capacitance matrix)  $D_i - B_i^T A_i^{-1} B_i$  and boundary load vector  $g_i - B_i^T A_i^{-1} f_i$  on boundary  $\partial\Omega_i$ . After get the boundary variable  $X_i$ , back substitute them each sub-domain to get the inner variable  $U_i$ .

This kind of method has many advantages that the first one does not possess. Firstly, the globe stiffness matrix can be divided into many sub-stiffness matrixes of the sub-domains, so the computation scale of each computer node is reduced sharply. If the parallel program is designed properly, the computation rate can be speeded up largely. Secondly, this algorithm needs very limited information exchange between master node and slave nodes and needs no information exchange between slave nodes. The communication between master node and slave nodes is limited to the boundary information of the sub-domain. Thirdly, this algorithm do not need the matched boundary meshes of two neighbor sub-domain, so the finite element meshes of different sub-domains can be divided respectively, in other words, they can be generated in a parallel fashion.

### 3. THE PARALLEL UNSTRUCTURED MESH GENERATION ALGORITHM ACCORDINACE WITH THE DDM METHOD

The parallel domain decomposition algorithm for the unstructured mesh is means that the mesh area will be divided into several sub-domains according the real problem or it can be evenly divided into many sub-domains. And then the unstructured meshes can be generated in the sub-domains separately. In this process, each node of the parallel machine (CPU) will take in charge of one or more than one sub-domain's mesh generation task. All the nodes (CPUs) generate meshes in their sub-domains according the boundary information of the sub-domains and the key-nodes or the mesh-size which were given by the master node. In this proceeding, many aspects must be taken into consideration.

#### 3.1 The Outline of the Algorithm

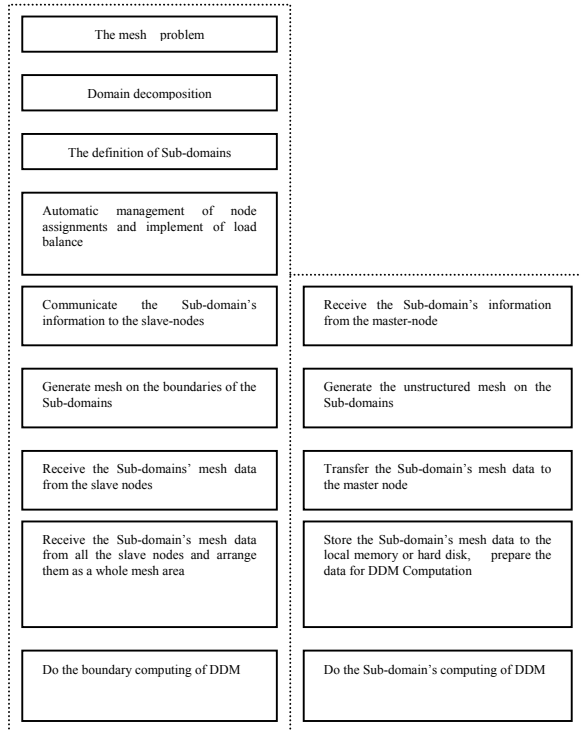


Figure 1. The outline of the parallel algorithm

We design this parallel unstructured mesh generation algorithm as a master-slave model, in which the master node will do the jobs such as domain decomposition, the node load balance assignment, the boundaries' unstructured mesh generation etc. And the slave nodes take in charge of the mesh generation on each sub-domain of the whole mesh area.

From figure 1 we can see that there are only two times of communication between the master and slave nodes. The first one is the master node communicate the information of the boundaries of the sub-domains to all the slave nodes. And the second time is that the all the slave nodes transfer the mesh

data of the sub-domains to the master node to form the mesh data for the whole mesh area. No communications were need between all the slave nodes. Fewer communications between all the nodes of the parallel machine means that this algorithm will achieve a higher parallel efficiency only if we design a good method for the load balance for all the nodes of the parallel machine.

#### 3.2 The Data Structure of Mesh Generation Subroutines

Data structure is very important in the processes of the mesh generation. And it becomes more intensive for the parallel algorithm. Considering the complexity of the data structure for the communication between the master node and the slave nodes, object-oriented programming style is very helpful. We enclose all the data to form a derived data type for all the communications. So the structure of the program for the mesh generation became very clear and simple. This can be easily realized by using the Fortran 90 compiler or C++ language.

#### 3.3 The Description of the Mesh Generation Problem for Each Node of the Parallel Machine

In this parallel algorithm, all the slave nodes would not know what work they will do if the master node would not tell them what to do or how to do. So an uniform definition for the sub-domains' mesh what will be generated must be given to the slave nodes first. Then the slave nodes will do their jobs what the master node assigned to them separately. We design all the information for the sub-domains as following:

- The boundary information of the sub-domains.
- Key-nodes, key-lines or key-surfaces with special mesh size, isotropic mesh etc.
- Material factor distribution on the sub-domain what finally be assigned to each mesh nodes .
- Sub-division information of the sub-domain.
- Match information with other sub-domains adjacent to it.
- Other information.

#### 3.4 The Communication Between the Master node and Slave nodes and the Storage of the Unstructured Mesh Data

We use message passing interface (MPI) as the basic communication library for the parallel program. The MPI's message passing library supplies rich procedures for different types of users, which contains two communication methodologies: point to point communication and collective communication. This algorithm mainly applies the point to point communication procedures of MPI. MPI mainly supplies two kinds of methods of data transmission mechanism for the point to point communication: block communication and non-block communication. The former can guarantee the safety of resource reuse, such as the use of

buffer (include receive buffer, send buffer etc.). This methodology simplified the complicated process of communication handshake and inquiring-in-turn of resource, and non-block communication allows overlapping the communication and it's calculating. Data duplication and communication can work at the same time under the support of the proper hardware. However, the return of non-block communication can not guarantee the resource reuse. Block and non-block communication call can both realize the following communication methodologies: standard, buffer, synchronous and ready.

Since these communication methodologies have different advantages and disadvantages under different situations, rational selection of these methodologies can improve communication efficiency and the parallel degree of parallel communication programs. For example, by running test on the two parallel machines the Dawning 2000 and IBM SP2, the communication data size is 512k bytes, the efficiency of communication of the buffer mode is at the highest efficiency.

#### 4. AUTOMATIC MANAGEMENT OF NODE ASSIGNMENTS AND AUTOMATIC IMPLEMENT OF LOAD BALANCE

When designing parallel algorithms and parallel computing programs, a main pending problem is that how to implement the load balance of each node. If the load of each node is distributed reasonably, the parallel degree of parallel computing will be increased greatly. In the designing of parallel unstructured mesh generation algorithm based on the DDM method, one of the emphases is how to implement the load balance technique of computing assignments without users' direct interference. It is necessary to point out that the concept of load balance in this parallel unstructured mesh generation algorithm is different from that of other load balance systems such as Loadlever running on many parallel machines, which is taking charge of node assignments in many parallel machines. Yet the aim is the same.

The computation scale of the mesh generation will be analyzed firstly, then it will be divided into many sub-domains or sub-problems. Each sub-domain or sub-problem will be seemed as a job to the nodes of the parallel machine. All the jobs can be aligned to a queue according to the load in each sub-domain. The algorithm will optimize these jobs in this queue in term of the principle of minimizing the difference among each node's amount of computation assignments. Finally system map the assignments in each sub-domain to different nodes to obtain load balance.

The arithmetic can be described as following:

- a. Account the amount of computation work in sub-domain  $T_i \quad i = 1, 2, 3, \dots, n$ ;
- b. Map the amount of computation in each sub-domain to jobs' queue which length is  $N$ ;
- c. On the assumption that there are  $M + 1$  nodes to be concerned with this parallel process, system point out a node

as master node (Rank=0), then it could acquire the rank of the rest  $M$  nodes;

- d. Map  $N$  assignments to  $M$  nodes, make the No.  $j$  node have  $k$  jobs, then system know the amount of assignment in each node  $Mtask_j = \sum_{i=1}^k T_i$ ;

e. Optimize the task of the node of  $M$ , let  $Max(Abs(Mtask_i - Mtask_j)) = Min$ , here  $i = 1, 2, 3, \dots, m \quad j = 1, 2, 3, \dots, m$ ;

- f. Map the optimized task sequence to the  $M$  nodes  $Mtask_i \Rightarrow Rank_i \quad i = 1, 2, 3, \dots, m$ ;

#### 5. SOME NUMERICAL RESULTS

In this part, we will give some mesh generation results and numerical results using finite element method.

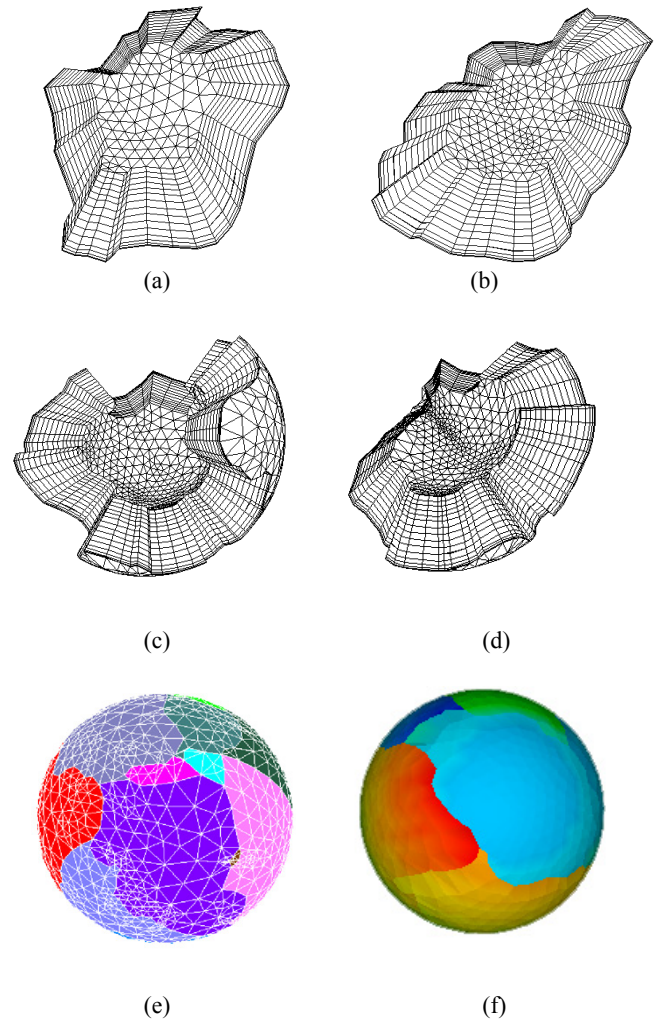


Figure 2. The difference blanks of the earth

We divide the whole earth into 16 different parts due to the blank theory of the earth. They have irregular boundaries. Each blank possesses many material factors. We use 17 computer nodes to do this mesh generation work. One of them (Rank = 0) do the master job. Each of the other nodes do gets the irregular boundaries of each blank and generate the unstructured mesh separately. We did this test on Dawning 3000 supercomputer and got speedup up to 20.01, which means that this parallel algorithm can get super-linear speedup.

Figure 2 (a), (b), (c) and (d) show the different parts of the earth. Figure 2 (e) shows that all the different parts of the earth which were generated by difference nodes of the parallel machine were combined together to form the whole earth's unstructured mesh. Figure 2 (f) shows computation result of the moving speed of the different parts of the earth's blanks. We use the finite element method based on the DDM algorithm, which we mentioned at the first part of this paper. The different colors of the blanks' show that all blanks have their own moving speed.

## ACKNOWLEDGEMENT

The authors gratefully acknowledge the inspiration of Prof. Shi Jincai of the National Research Center of Intelligent Computation (NRCIC), many helpful and constructive suggestions are essential to this project. The authors also sincerely thank NRCIC for the use of Dawning-3000 parallel computer.

## REFERENCES

- [1] The Langrage multiplier domain decomposition method of hybrid finite element, Guoping Liang, Ping Liang, *J. Computation Math.*, Vol. 3 1998
- [2] MPI: A Message-Passing Interface Standard, *Message Passing Interface Forum*, 1998,5
- [3] J. A. Talbert and A. R. Parkinson, Development of an Automatic Two-Dimensional Finite Element Mesh Generator Using Quadrilateral Elements and Bezier Curve Boundary Definition, *Int. J. Numer. Methods Engrg.*, 29, 1551-1567 (1990)
- [4] B. P. Johnston, J. M. Sullivan, Jr. and A. Kwasnik, Automatic Conversion of Triangular Finite Element Meshes to Quadrilateral Elements, *Int. J. Numer. Methods Engrg.*, 31, 67-84 (1991)
- [5] Mark A.Yerry and Mark S, Shephard, Three Dimensional Mesh Generation by Modified Octree Technique, *International Journal for Numerical Methods in Engineering*, 20, 1965-1990 (1984)
- [6] W.A.Cook, and W.R. Oakes, Mapping Methods for Generating Three Dimensional Meshes, *Computers in Mechanical Engineering*, August, 67-72 (1982)
- [7] Freitag, Lori A. and Carl Ollivier Gooch, Tetrahedral Mesh Improvement Using Swapping and Smoothing, *International Journal for Numerical Methods in Engineering*, 40, 3979-4002 (1997)