A domain decomposition method based on the AD algorithm

WANG Ru-yun
College of Harbor, Coastal and Offshore Engineering, Hohai University
Nanjing, China
E-mail: wangry@hhu.edu.cn

CAO Di
College of Harbor, Coastal and Offshore Engineering, Hohai University
Nanjing , China

YU Yin-xian
College of Harbor, Coastal and Offshore Engineering, Hohai University
Nanjing, China

FANG Qing
Department of Mathematical Sciences Faculty of Science Yamagata University
Yamagata 990-8560 Japan

ZANG Zhen-tao
College of Harbor, Coastal and Offshore Engineering, Hohai University
Nanjing, China

Abstract—We first establish the relationship between the grid cell connectivity matrix bandwidth and the efficiency of parallel computing, which shows that parallel efficiency can be improved by reducing the bandwidth so that the external communication is reduced. An algorithm based on the AD algorithm is developed which can reduce the connectivity matrix bandwidth, so that parallel efficiency can be improved. The purposed algorithm uses cell ponderation to relabel cell labels. A domain decomposition method is then developed based on the new algorithm. Parts of unstructured grid of global ocean are researched based on the developed domain decomposition method. The investigation shows that our method has feasibility and effectiveness for domain decomposition of large-scale grids since the speedup and efficiency are obtained from our method.

Keywords-domain decomposition; parallel computing; AD algorithm; unstructured grid; parallel computing efficiency

I. INTRODUCTION

An important process in parallel computing is the division of the computational domain in large-scale scientific computing problems such as fluid dynamics and solid dynamics into smaller sub-regions with mapping to processors, which is known as domain decomposition. Domain Decomposition Method (DDM) is the method of how to divide computing grid and achieve computing load equilibrium, which directly affect the efficiency of parallel computing. Therefore, DDM has been an important subject of many researchers of both domestic and overseas in the high-performance scientific computing area.

Unstructured grid is widely used in the finite element method, the finite difference method, the finite volume method and others since it can accurately and effectively fit the complicated shape of the domain. At present, the commonly used DDMs based on unstructured grid are: Advancing Front Technique [1-2], Graph Partitioning Methods [3] (which contains Recursive Spectral Bisection (RSB)[4-6]), Multilevel Algorithms[7-8], Multilevel Recursive Spectral Bisection (MRSB) [9], Hypergraph Partitioning[10], etc. The advantage of Advancing Front Technique is its simplicity and easy implementation, but it has a large amount of calculation and complex processes. Graph Partitioning Methods approximates the minimum amount of segmentation so that it makes the information exchange close to the minimum between the processor, but the size of the Lagrangian matrix generated by this algorithm has a square relationship with the total grid numbers so that a large amount of memory are needed. At the same time, the algorithm needs the computation of Fiedler eigenvectors, which calls a large amount of computation time. Multilevel Algorithm can be roughly divided into three phases: coarsening, partitions and optimization, with the optimization phase generally using the KL/FM [11] method. But this algorithm has more complex implementation steps, and the KL/FM method may increase the overall computing time. MRSB method combines the multilevel algorithm and RSB method, and is widely used in the world. This method has the advantages of less processor expenses and fast calculation, but because the geometric topology information of finite element model may partly disappear in the coarsening phase, the load in processors is disequilibrium and the boundary between each part is too long. Hypergraph Partitioning Method can very accurately show the communications, and has the small amounts of external communications, but its cost is higher than other Graph Partitioning Methods.

A relatively simple and easy DDM is to distribute cells evenly to each processor according to cell labels. However direct use of this method applied to dividing girds, which are generated by some automatic mesh subdivision software, may lead to low computing efficiency. According to the analysis of researches, the relationship

between the grid connectivity matrix bandwidth and the amount of external communication is confirmed: when the bandwidth is reduced, it yields that the external communication is reduced and parallel efficiency is improved. There are many methods to reduce matrix bandwidth and one of them is known as the AD algorithm, which has been designed by Akhras and Dhatt in 1979. This algorithm relabels the node labels of a given mesh to reduce the matrix bandwidth of high order sparse connectivity matrices. The AD algorithm is simple and easy for parallel computing. Based on the AD algorithm, a new domain decomposition method has been designed: firstly, using the AD algorithm to relabel the cell labels of unstructured grid and to reduce the bandwidth of connectivity matrices; secondly, distributing cells evenly to each processor according to the cell labels and properties. Researches show that the proposed method can achieve load equilibrium in each processor and it is feasible and effective for large-scale grid domain decomposition.

II. THE RELATIONSHIP BETWEEN GRID CELL CONNECTIVITY MATRIX BANDWIDTH AND THE EFFICIENCY OF PARALLEL COMPUTING

In the usual numerical simulation and computation of fluid dynamics and solid dynamics, the data of peripheral cells are needed when calculating the present cell's physical quantity. When the mesh is divided into sub-regions, some of the data needed to transmit are not in the same processor, which leads to external communication between processors. Reducing the external communication can then improve the efficiency of parallel computing.

Labeling the whole mesh cells as $1,2,\mathrm{L},N$, we give the following three definitions:

1) *Correlate cell*: the data of peripheral cells are needed when calculating the present cell's physical quantity. These cells are called correlate cells. If cell i is a correlate cell of cell j and, in general, cell j is also a correlate cell of cell i, then we say that cell i and cell j is related. Let

$$a_{ij}=\begin{cases}1 & \text{cell}i \text{ and cell}j \text{ related.}\\ 0 & \text{cell}i \text{ and not cell}j \text{ not related}\end{cases} \quad (1)$$

The data of present cell is always needed during its calculation, so $a_{ii}=1$. Let $m_0=\sum_{i,j=1}^{N}a_{ij}-N$, then this value has nothing to do with the cell labels and it is an invariant quantity of connectivity matrix.

2) *cell connectivity matrix:* matrix $A=(a_{ij})_{N\times N}$ is called cell connectivity matrix. It satisfies $A=A^T$

3) *cell labels domain decomposition method (cell labels methods)*: assume that the total number of processors is r and distribute N ($N=Kr$) cells' computing tasks evenly to each processor according to the cell labels, where processor I ($I=1,2,\mathrm{L},r$) has the computing tasks of cell $(I-1)K+1,\mathrm{L},IK$.

After cell labels domain decomposition, the cell connectivity matrix is divided into r order matrixes. $A=(A_{I,J})_{r\times r}$, where $A_{I,J}=(a_{(I-1)K+i,(J-1)K+j})_{K\times K}$, $I,J=1,2,\mathrm{L},r$

Obviously, $E_{I,J}$ -the number of cells that need exchanging information between processor I and processor J, equals the number of element in matrix $A_{I,J}$;

$$E_{I,J}=\sum_{j=(J-1)K+1}^{JK}\sum_{i=(I-1)K+1}^{IK}a_{ij} \quad (2)$$

The exchanging information between processor I and processor J is in proportion to $E_{I,J}$ with the proportionality coefficient a constant. When $I=J$, $E_{I,J}$ means the number of cells that need internal communication. When $I\neq J$, $E_{I,J}$ means the number of cells that need external communication. The number of cells that need exchanging information between processor I and all the other processors ($J=1,\mathrm{L},I-1,I+1,\mathrm{L},r$) is $\hat{E}_I$.

$$\hat{E}_I=\sum_{J\in\{1,\mathrm{L},r\}\setminus\{I\}}E_{I,J} \quad (3)$$

Assume that the half bandwidth of connectivity matrix is L. Then if the element is located outside the band ($|i-j|>L$), $a_{i,j}=0$; if the element is located in the band $|i-j|\le L$, $a_{i,j}$ may equals 1. Assume that in the band, in addition to the main diagonal elements which values are 1, elements of other position have equal opportunity to get the values of 1. The computational formula of $\hat{E}=\sum_{I=1}^{r}\hat{E}_I$ is defined in two cases by using the characteristic quantity m_0.

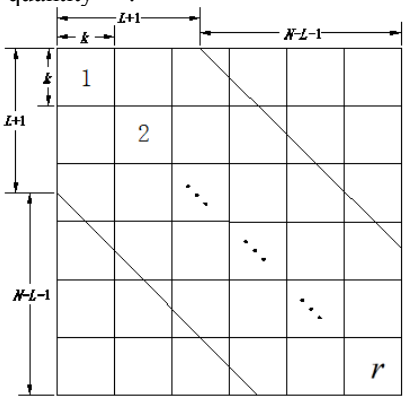

Figure 1. Figure.1 Connectivity matrix sketch map

When $K\le L+1\le N$,

$$\hat{E}(L)=\frac{m_0}{N^2-(N-L-1)^2-N}\{[N^2-(N-L-1)^2-N]-r[K^2-K]\}$$

$$=m_0[1-\frac{N(K-1)}{(L+1)(2N-L-1)-N}] \quad (4)$$

$\hat{E}(L)$ increases monotonously with respect to L, and reaches the maximum when $L=N-1$.

When $1\le L+1\le K$,

$$\hat{E}(L)=\frac{m_0}{N^2-N-L-2-N}\{[N^2-(N-L-1)^2-N]-r[K^2-(K-L-1)^2-K]\}$$

$$=m_0\frac{N-K}{NK}\frac{1}{1-\frac{1}{N}-(1-\frac{1}{L+1})^2} \quad (5)$$

$\hat{E}(L)$ increase monotonously with respect to L, and reaches the maximum when $L=K-1$.

Further conclusions from the above two cases: (i) $\hat{E}=\sum_{I=1}^{r}\hat{E}_I$ increases monotonously with respect to L, (ii) According to the structure of the band of connectivity matrix, the number of cells that need external communication in the first processor and the last processor is either $\frac{1}{2(r-1)}\hat{E}$, the number of cells that need external communication in other each processor is $\frac{1}{r-1}\hat{E}$;(iii) The computing time of external communication is larger than internal communication, thus the reduction of the cell number that need external communication (convert to internal communication) can reduce the external communication quantities, the efficiency of parallel computing improves as well. Therefore, reducing of the bandwidth of connectivity matrix can result in reducing of external communication quantities. The aim of improving the parallel computing efficiency is then achieved.

III. THE AD ALGORITHM FOR REDUCING THE BANDWIDTH OF THE CONNECTIVITY MATRIX

A. *Introduction of the AD algorithm*

The AD algorithm [12] renumbers the node labels by following three properties:

1) *Node-sum*: assuming that there are m cells having the same node as their vertex, sum up the labels of each cell's vertex, then sum up the total label number of m cells' nodes. That is the node-sum.
2) *Node-ponderation*: the result of node-sum divided by m is called node-ponderation.
3) *Span*: for the cells that have the present node as one of their vertex, the sum of the minimum vertex label and maximum vertex label is called span.

The AD algorithm is divided into two phases:

Phase 1: The nodes are relabeled based on the span criterion and the node-ponderation criterion respectively in ascending order. The relabeling process is repeated a number of times by these two criteria until no more reduction in the bandwidth are obtained.

Phase 2: After Phase 1, the nodes are relabeled such that the node-ponderation criterion and the node-sum criterion are referred simultaneously in ascending order. This process is repeated until no further reduction in bandwidth is realized. Usually, Phase 1 is re-executed for a possible bandwidth reduction.

B. *The AD algorithm for reducing the bandwidth of the connectivity matrix*

The original AD algorithm is aimed at reducing the matrix storage. It can renumber the node labels to reduce the bandwidth. From the above research, we know that the reducing of the bandwidth of connectivity matrix can reduce external communication quantities so that the aim of improving the efficiency of parallel computing is achieved. Based on the original AD algorithm, a new AD algorithm for reducing the bandwidth of the connectivity matrix is designed in this section.

By following the approach of original AD algorithm, two variables are defined:

1) *Cell-sum*: assume that there are m cells having communication with the current cell, sum up the labels of these cells. That is cell-sum.
2) *Cell-ponderation*: the result of cell-sum divided by m is called cell-ponderation.

The original AD algorithm has two criterions in each phase, which may lead to node labels oscillation so that it reduces efficiency to some extent. Besides, the two phases have the same problem: the local order of the node labels that relabeled by the first phase may be disrupted due to the change of criterions when it comes to the second phase. This may also reduce efficiency. To avoid oscillation caused by this algorithm, cell-ponderation is used as the only criterion of our algorithm based on the AD algorithm for reducing the bandwidth of the connectivity matrix.

We introduce three variables to avoid making the program getting into an infinite loop:

1) *Reverse ratio*: check the cell-ponderation of cells according to the labels of cells such that if the cell-ponderation of the later cell is less than the former, we call it a reverse order. The ratio of the total reverse orders and the number of cells is called reverse ratio.
2) *Half bandwidth*: let the difference between the farthest correlate cell label and the current cell label be L_0, the maximal L_0 is called half bandwidth.
3) *Maximal cell-ponderation difference* ΔA: $\Delta A=\max(|A(i)-A_0(i)|),i=1,2,\mathrm{L},n$, where $A(i)$ and $A_0(i)$ represent the cell-ponderation of current iteration and the last iteration respectively.

The process is repeated until no further reduction in these three variables for successive two times of iterations.

The proposed algorithm based on the AD algorithm for reducing the bandwidth of the connectivity matrix has the following processes:

1) Construct the cell connectivity matrix by using the data of other cells if needed according to the

algorithm;

2) Calculate the cell- ponderation of each cell according to the cell connectivity matrix;
3) The cell label is relabeled based on the cell-ponderation criterion and the relationship between the original cell labels and the present cell labels is recorded;
4) The new connectivity matrix is constructed according to the relationship between the original cell labels and the present cell labels and the information of original matrix;
5) Calculate the reverse ratio, half bandwidth and maximal cell-ponderation difference ΔA. If no further reduction in these three variables for successive two times of iterations takes place, then go to the next step, otherwise go to the (2) step;
6) End the process and output the results.

IV. DOMAIN DECOMPOSITION METHOD BASED ON THE AD ALGORITHM

A. *Domain decomposition method according to the cell labels and properties*

Cell labels method can achieve the equilibrium of cell number in each processor, but the method ignores the fact that cells of different properties have different computing time. To achieve the equilibrium of load in each processor, both cell labels and cell properties should be considered. Assuming that the number of cell is N, the number of cell is r, the number of cell properties is m, the cell number in different properties is $n_1, n_2, \mathrm{L}, n_m$, an improved method is developed based on the cell label method: for any cell which property is j ($j=1,2,\mathrm{L},m$), the cells holding this property are evenly distributed to each processor, and the number in each processor is approximately determined by $k_j=[n_j/r]$. The total number of cells in each processor is approximately equal to N/r. This is called the labels and properties method for short.

B. *The new AD domain decomposition method*

The labels and properties method can perform well in domain decomposition with the computing time in each processor being approximately equal. But it ignores the problem of external communication so that this method may cause large amount of external communication, and may bring bad influence to the efficiency of parallel computing. Therefore, we should use the AD algorithm which reduces the bandwidth of the connectivity matrix before the labels and properties method. This process is aimed at reducing the external communication and improving the efficiency of parallel computing. We call this process as the AD domain decomposition method.

The AD domain decomposition method is composed of following two parts:

1) Relabeling the cell labels based on the AD algorithm which reduces the bandwidth of the connectivity matrix.
2) Taking domain decompose by the labels and properties method.

V. Numerical examples

The tests are carried out on the platform of Pentium E5400 with Windows 7 OS and 2GB memory.

The tests are based on a higher-order weighted essential non-oscillatory (WENO) method on sphere 2D unstructured triangular meshes and on the background of global tide wave movement. In our numerical simulation, there is no open boundary but only continent boundary. The computation time of internal grids is $t_1=3.1875\times10^{-6}$s and the computation time of continent boundary grids is $t_2=3.203125\times10^{-4}$s. The computation time of internal communication (the data exchanges happen between adjacent cells in the same processor) is $t_3=3.125\times10^{-8}$s and the computation time of external communication (the data exchanges happen between adjacent cells in the different processor) is $t_4=3.90625\times10^{-6}$s. Two levels of connective cells are needed based on the WENO method (Fig .2). For the connectivity matrix, when the present cell is o, the values of a_{oi}, a_{oi_a}, a_{oi_b}, a_{oj}, a_{oj_a}, a_{oj_b}, a_{ok}, a_{ok_a}, a_{ok_b}, a_{oo} are 1, and the values of a_{io}, a_{i_ao}, a_{i_bo}, a_{jo}, a_{j_ao}, a_{j_bo}, a_{ko}, a_{k_ao}, a_{k_bo} are also 1 according to the symmetry of this matrix.

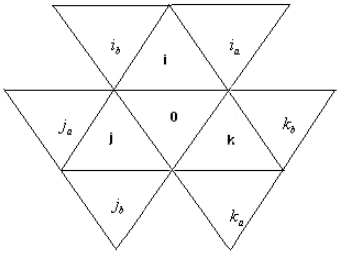

Figure 2. Classic cell computing template

The mesh data from the following numerical examples is a part of global non-open boundary unstructured triangular meshes which are created by SMS software.

Example 1

The number of cells is 54311 and the Sea island distribution map is shown in Fig .3. The results of different numbers of processors and different methods are shown in Table.1. When the number of processors is 4, the performance of different methods are shown in Fig .4-6, where the 4 different colors represent cells in different processors.

TABLE1. DOMAIN DECOMPOSITION RESULT OF DIFFERENT PROCESSORS IN EXAMPLE 1

processors		4	8	16	32	64	128	256
Cells label method	Speed up	2.241	3.488	5.373	10.486	17.271	28.88	46.225
	efficiency	56.03%	43.60%	33.58%	32.77%	26.98%	22.56%	18.05%
	external communication	6371	13899	28157	56127	95021	127257	161754
labels and properties method	Speed up	3.752	7.079	12.917	22.061	39.067	72.628	127.784
	efficiency	93.80%	88.49%	80.73%	68.94%	61.04%	56.74%	49.91%
	external communication	10103	18848	34036	63302	102555	134893	169441
AD domain decomposition method	Speed up	3.907	7.6304	14.897	28.288	50.560	83.878	137.805
	efficiency	97.68%	95.38%	93.11%	88.40%	79.00%	65.53%	53.83%
	external communication	3463	7598	13704	23649	42445	79040	143296

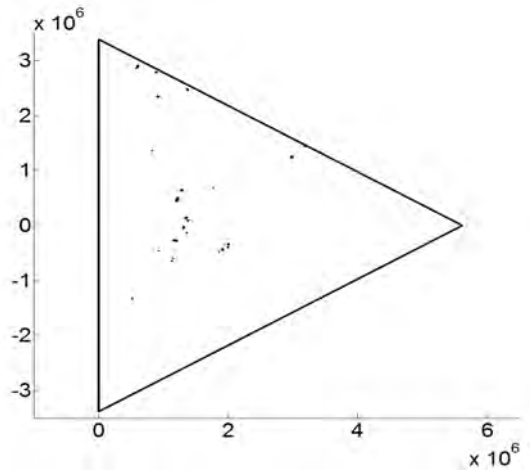

Figure 3. Sea island distribution map

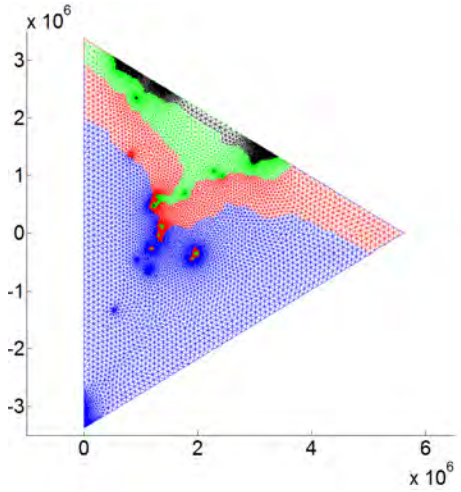

Figure 4. result of the cell labels method in 4 processors

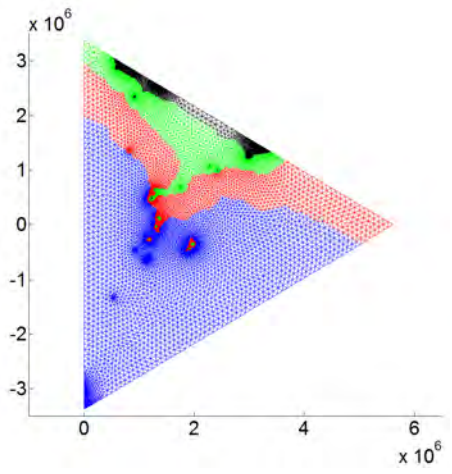

Figure 5. result of the labels and properties method 4 processors

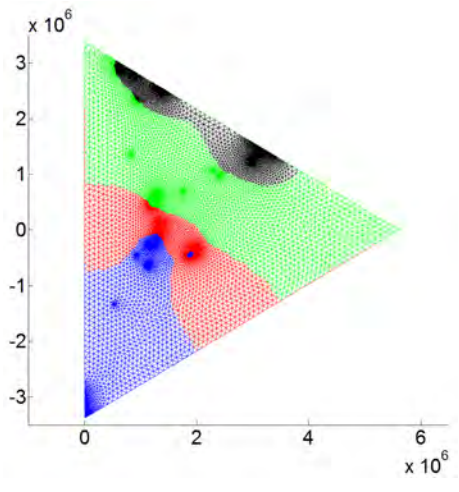

Figure 6. result of AD domain decomposition method 4 processors

Example 2

The number of cells is 588755 and the results of different numbers of processors and different methods are shown in Table.2.

TABLE II DOMAIN DECOMPOSITION RESULT OF DIFFERENT PROCESSORS IN EXAMPLE 2

processors		4	8	16	32	64	128	256
Cells label method	Speed up	3.231	5.954	11.007	18.066	24.216	37.317	48.956
	efficiency	80.78%	74.42%	68.79%	56.45%	37.83%	29.15%	19.12%
	external communication	13638	28231	57528	114813	232013	466692	897020
labels and properties method	Speed up	3.92	7.673	14.764	27.623	49.019	80.097	128.35
	efficiency	98.01%	95.91%	92.28%	86.32%	76.59%	62.57%	50.13%
	external communication	17194	37012	74255	137826	256893	493058	924123
AD domain decomposition method	Speed up	3.935	7.779	15.398	30.06	57.712	106.066	183.029
	efficiency	98.38%	97.23%	96.24%	93.94%	90.17%	82.86%	71.50%
	external communication	11932	24901	39082	62922	106469	189907	355394

The numerical examples show that although the cell labels method is easy and fast, its efficiency is low. The efficiency of labels and properties method is improved, but it has a large amount of external communication so that it is not conducive to the parallel efficiency. The AD domain decomposition method can reduce external communication, and achieve higher efficiency. In general, because the increase in processors causes the increase in the amount of external communication, the parallel efficiency will decrease with the increase in processors. By using the AD domain decomposition method, the computation time in Example 1 which has 54311 cells is 684.831s, and the computation time in Example 2 which has cells 588755 is 74978.228s. These data show that the AD domain decomposition method can achieve higher efficiency in a short time. The figures show that the AD domain decomposition method can centralize cells in each processor, and the continent boundary cells in each processor is even. The AD domain decomposition method can achieve higher efficiency in a short time when dealing with large-scale data. The feasibility and efficacy of our method are explored by the numerical results.

VI. CONCLUSION

For the similar problems, the parallel efficient of 4 processors is 85%-97%, and the parallel efficient of 256 processors is 20%-70%. Using the labels and properties method after relabeling cells label by the AD algorithm can achieve higher speed up and efficiency. From the test of 588755 cells, the results show that the AD domain decomposition method can achieve higher efficiency in a very short time. The feasibility and effectiveness of the method for large-scale grid domain decomposition is shown.

ACKNOWLEDGMENT

The work is supported by the Fundamental Research Funds for the Central Universities, NO.2014B06314

REFERENCE

[1] Lohner Rainald. A Parallel Advancing Front Grid Generation Scheme[J]. International Journal for Numerical Methods in Engineering, 2001,51(6): 663-678.

[2] SI Hai-qing WANG Tong-guang CHENG JuanDomain decomposition and parallel algorithms to solve Euler equations on the unstructured grid [J] [in Chinese].ACTA Aerodynamica Sinica,2006,24(1):102-108.

[3] Henrickson B, KoldaTG Graph Partitioning Models For Parallel computing [J]. Parallel Computing,2000,26 (12):1519-1534.

[4] Pothen A, Simon H, Liou K P, Partitioning Sparse matrices with eigenvectors of Graphs[J] .SIAM J.Math.Anal.Appl,1990,11(3):430-452.

[5] Simon H.D. Partitioning of Unstructured Problems for Parallel Processing. Comput. Eng[J], 1991.2(2/3),35-148.

[6] ZHOU Chun-hua. A Method of Nonstructured Mesh Partition for Parallel Computation[J] [in Chinese]. ACTA AERONAUTICA ET ASTRONAUTICA SINICA,2004,25(3):229-232.

[7] Walshaw C, Cross M. Mesh partitioning: A multilevel balancingand refinement algorithm [J]. SIAM Journal on Scientific Computing, 2000,22(1):63-80.

[8] Karypis G, Kumar V. A fast and high quality multilevel scheme for partitioning irregular graphs [J].SIAM Journal on Scientific Computing,1998, 20(1): 359-392.

[9] CATALYUREK U V, BOMAN E G, HEAPHY R, et al. Hypergraph based dynamic load balancing for adaptive scientific computations[C]/ /Proc of IEEE International Conference on High Performance Computing,2007: 367-380.

[10] TRIFUNOVI A, KNOTTENBELT W J. Parallel multilevel algorithms for hypergraph partitioning[J]. Journal of Parallel and Distributed Computing, 2008, 68(4) : 563-581.

[11] Kernighan BW, Lin S. An efficient heuristic procedure for partitioning graphs[J]. The Bell System Technical Journal, 1970, 49(2): 291-307.

[12] Akhras G, Dhatt G. An automatic node relabeling scheme for minimizing a matrix or network bandwidth[J]. International Journal for Numerical Methods in Engineering,1976, 10(4):787-797.

[13] ZHAO Yujing, WU Jianjun, FU Wenzhen. algorithm for optimizing the node number of the numerical simulation in sheet forming[J] [in Chinese]. China metal forming equipment and manufacturing technology, 2009,44(06):73-75.

[14] Hu Zhiyu, Zou Qi, Chen Tao3. An Improved AD Algorithm for Finite Elements Mesh Code Optimization [J] [in Chinese]. J of China Three Gorges Univ.(Natural Sciences), 2009,31(02):63-65.Hoffmann K H, Jun Zou, Parallel efficiency of domain decomposition methods[J], Parallel Computing,1993,19(12): 1375-1391