LOAD BALANCE ALGORITHM RESEARCH ON PARALLEL PARTITIONED MULTI-BLOCK STRUCTURED OVERLAPPING GRID

WANG WEN*, JIANG HUABING* AND LI CHUNLI*

* Institute of Systems Engineering, China Academy of Engineering Physics
621000 Mianyang, Sichuan, CHINA
e-mail: wangwenbuaa@126.com

Key words: Computational fluid dynamics; Overlapping grid; Parallel computation; Load balance algorithm.

Abstract: Grid technologic acts as a crucial role in Computational fluid dynamics. As decreasing the demand of topology and the difficulty of grid generation efficiently, overlapping grid has been widely used in the past 30 years. The efficiency demand of parallel computation needs the extending of serial overlapping algorithm to parallel environment and well load balance. Based on the efficient, robust and fully automatic grid assembly approach on multi-block cell-centred structured grid for massively parallel computation, two load balance algorithms are designed to solve the imbalanced problem. A test case is applied to test those load balance algorithms. The comparison of speed-up and detail of grid transport quantity is presented and shows that the improved load balance algorithm achieves good speed-up. Meanwhile, there is still some optimization remaining to be done in the future work.

1 INTRODUCTION

The overlapping grid assembly method is widely investigated. Currently, there are several developed grid assembly packages such as PEGASUS5 [1], SUGGAR++/DiRTlib [2], OVERFLOW [3], BEGGAR [4], FASTRAN [5], Overture [6], and ElsA [7]. Meanwhile, some teams devoted themself to improve the overlapping grid assembly method and efficiency. Based on the grid size of decomposed domain, Kima [8] developed an overlapping grid assembly approach. Landmann [9] extended the Implicit Hole Cutting (IHC) [10,11] algorithm to the parallel environment. However, most of their parallel efficiency is less than desirable. Partitioning some of the partitioned grids again, Roget [12] proposed an adaptive load re-balance method on partitioned unstructured grids to achieve better load balance. However, this method is not suitable for partitioned structured grids because the disordered distribution of those grids on processors will lead to the discontinuity of grids in space. The grid re-partition algorithm by applying a volumetric division cannot deal with the discontinuous portioned structured grids. Fig. 1 shows a typical example of partitioned grids in a parallel environment. Different colors represent different processors. It is clear that the grids of a body distributed to a specific processor do not continue generally, especially for near body grids. It is the most significant difference with the partitioned unstructured grids. After dealing with the overlapping process on parallel enviroment, how to achiveve a high effience in the overlapping process is another curcial aspect. Based on our improved overlapping grid approach, three load balance algorithms are achieved and some detials of grid transportation among processes will be discussed in this parer.

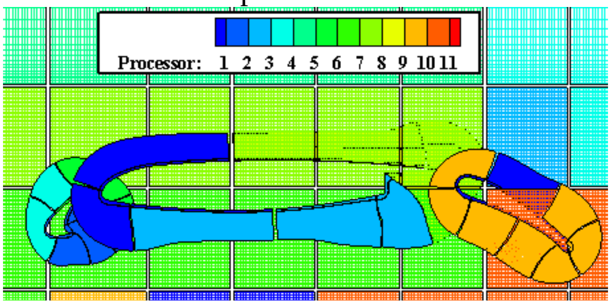


Fig. 1 A multi-block structured partitioned grids in the parallel environment

2 LOAD BALANCE ALGORITHMS

2.1 Without load balance

Donor search is a crucial step in the overlapping grid assembly method. The purpose of donor search is to determine the donor cells of query points and their interpolation weights. Through Query point identification, the target processors of all query points should be sent to for donor search are picked out. Without any load balance consideration, the query points are packed and sent to their target processors directly.

2.2 Initial load balance algorithm

The first step of initial load balance algorithm is to estimate the load for every processor. As mentioned before, the number of query points per processor is used as the criterion of load balance. With the help of the above query point identification method, the number of query points of every Cartesian cell can be estimated easily. Then the average load (*La*) among all processors is computed as follow:

$$La = \sum_{m=1}^{nobj} \cdot \sum_{(i,j,k) on\ AG_m} N_{i,j,k,m}^{query} / N \qquad (1)$$

where *nobj* is the number if the bodies, $N_{i,j,k,m}^{query}$ is the number of query point located in the (*i*,*j*,*k*) Cartesian cell of *AGm* and *N* is the number of processors.

The second step is to redistribute load for every processor. The load on every cell of AGs is clear and every AG cell is marked as target ID where its query points should be sent for donor search orderly. The algorithm is illustrate in Fig 2.

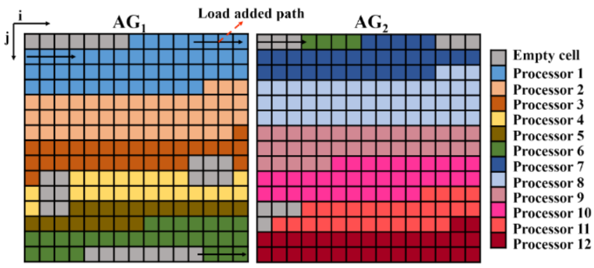


Fig. 2 Initial load balance algorithm in 2D

2.3 Improved load balance algorithm

For a query point, the computation cost also depends on the number of ADT nodes. The necessary cost to solve the geometric searching and intersection problems is found to be proportional to log($N*$) where $N*$ is the number of ADT nodes. Then the load on the (i, j, k) Cartesian cell of $AG_m(L^m_{i,j,k})$ is determined as follow:

$$La = \sum_{m=1}^{nobj} \cdot \sum_{(i,j,k) on\ P_n} N^{query}_{i,j,k,m} \cdot \log \sum_{(i,j,k) on\ P_n} N^{ADT}_{i,j,k,m} \tag{2}$$

where *nobj* is the number of bodies. Different from the above algorithm which only uses the number of query points as the load, this average load (La) cannot be estimated easily. Therefore, the distribution algorithm needs to be redesigned as follows.

As the similar distribution process, the illustration of improved load balance algorithm is shown in Fig3.

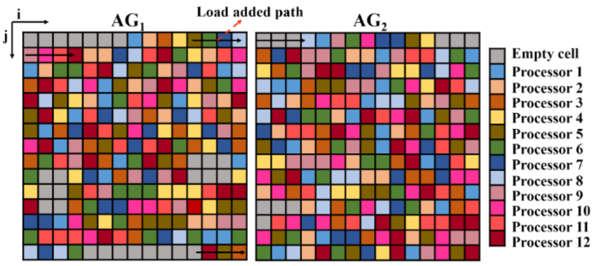
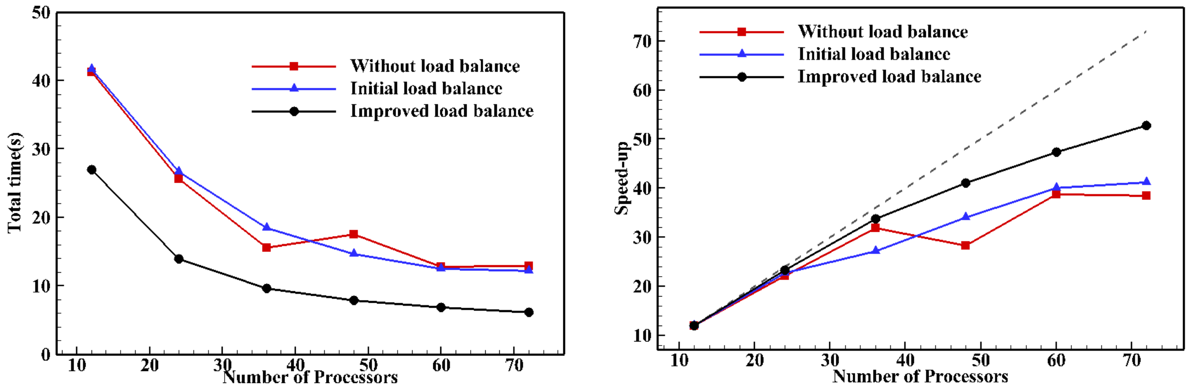


Fig.3 Improved load balance algorithm in 2D

3 TEST CASE AND CONCLUSIONS

The 30P30N airfoil is applied to test the efficiencies of those load balance algorithms. The total number of grids is 27 million. This case runs on 48 processors (2.6 GHz Intel Xeon E5-2650 Sandy Bridge processors). Fig 4 shows the variation of speed-up and total time required to perform the grid assembly of those three algorithms, for different numbers of processors (12, 24, 36, 48, 60 and 72). The improved load balance algorithm keeps the speed-up very well.



(a) Total time of overlapping grid assembly (b) Speed-up of overlapping grid assembly

Fig.4 Time to perform of three overlapping grid assembly algorithms

REFERENCES

[1] Rogers S E, Suhs N E, Dietz W E. PEGASUS 5: an au-tomated preprocessor for overset-grid computational flu-id dynamics[J]. AIAA journal, 2003, 41(6): 1037-1045.

[2] Noack R W, Boger D A, Kunz R F, etal. Carrica, Suggar++: An improved general overset grid assembly capability, in: 19th AIAA Computational Fluids Dynamics Conference, San Antonio, TX, 2009.

[3] Buning P G, Pulliam T H. Cartesian off-body adaptation for viscous time-accurate flow simulations, in: 20th AIAA Computational Fluid Dynamics Conference, Honolulu, Hawaii, 2011, AIAA Paper 2011-3693.

[4] Belk D, Maple R. Automated assembly of structured grids for moving body problems[C], in: 12th AIAA Computational Fluid Dynamics Conference, Part 1, Washington, DC, 1995, pp. 381-390.

[5] Wang Z J, Parthasarathy V. A fully automated Chimera methodology for multiple moving body problems[J]. International Journal for Numerical Methods in Fluids, 2000, 33(7):919-938.

[6] David L, Brown W D H, Quinlan D J. Overture: object-oriented tools for overset grid applications, in: 17th AIAA Conference on Applied Aerodynamics, Washington, DC, 1999.

[7] Jeanfaivre G. Juvigny X. Benoit C. Parallel Chimera computations of helicopter flows, in: 24th International Congress of the Aeronautical Sciences, 2004.

[8] Kim E, Kwon J H, Park S H. Parallel Performance Assessment of Moving Body Overset Grid Application on PC Cluster[J]. Parallel Computational Fluid Dynamics, 2007, 57(12):59-66.

[9] Landmann B, Montagnac M. A highly automated parallel Chimera method for overset grids based on the implicit hole cutting technique[J]. International Journal for Numerical Methods in Fluids, 2011, 66(6):778–804.

[10] Lee YL and Baeder JD. Implicit hole cutting-a new approach to overset grid connectivity[J], AIAA Paper 2003-4128, 2003.

[11] Lee YL and Baeder JD. High-order overset method for blade vortex interaction[J], AIAA Paper 2002-0559, 2002.

[12] Roget B, Sitaraman J. Robust and efficient overset grid assembly for partitioned unstructured meshes[J]. Journal of Computational Physics, 2013, 260(1):1-24.