A Novel Adaptive Match Scheme for Parallel Mesh-decomposition in OpenFOAM

Miao Wang, Yuhua Tang
State Key Laboratory of High Performance Computing,
National University of Defense Technology,
Changsha, 410073, China
E-mail: mercury.miao@gmail.com

Hongyun Zhang
School of Computer,
National University of Defense Technology,
Changsha, 410073, China
E-mail: grandcloud88@gmail.com

***Abstract*—A parallel multilevel k-way partitioning algorithm is used in OpenFOAM to perform parallel mesh-decomposition. Before the parallel decomposition procedure, mesh has to be pre-decomposed by the Simple method in OpenFOAM. Match-computation for the algorithm's parallel coarsening phase is based on a kind of global match scheme which introduces large amount of communication overhead. However, the domains of the mesh generated by Simple maintain good locality and continuity, which makes it unnecessary to adopt global match scheme in the parallel coarsening phase. In this paper, a novel adaptive match scheme AMS is brought forward for the parallel multilevel k-way partitioning algorithm. An adaptive critical x is calculated firstly according to the scale of the mesh and the parallel degree. For the first x stages of the coarsening phase, a local match scheme is adopted in which vertexes are only allowed to match with their adjacent unmatched vertexes with heaviest edge-weight on their local processors. For the rest stages, the traditional global match scheme is introduced and match of two vertexes on different processors is allowed. AMS can efficiently reduce the communication overhead introduced by simply adopting global match scheme. The experiment is performed on mesh of LinearPTT application on Tianhe-1A. The results show that the parallel multilevel k-way paritioning algorithm based on AMS has a better performance than that based on traditional global match scheme.**

Keywords-parallel graph-partitioning; mesh-decomposition; OpenFOAM; adaptive match scheme

I. INTRODUCTION

OpenFOAM is an open source CFD software commonly used to analyze, compute and predict the behavior of hydrofield [1]. The hydrofield is described by a series of values in corresponding discrete locations and a mesh generated from the field for the following computation of some algebraic equations. CFD applications running in OpenFOAM are generally of large scale and needed to be executed in parallel to achieve high execution efficiency. For parallel execution, it is necessary to decompose the generated mesh into several domains and assign each of them onto a single processor. This mesh-decomposition procedure in OpenFOAM is accomplished by some graph-partitioning algorithms in which graphs are used to describe the mesh.

A parallel multilevel k-way graph-partitioning algorithm implemented in parMetis [2] is used for parallel mesh-decomposition in OpenFOAM. The algorithm is adopted in this way: the mesh is firstly decomposed by a fast serial decomposition method called Simple; then the decomposed mesh is described as a distributed partitioned graph and the algorithm adopted to compute a re-partition of the graph in parallel based on the existed partition. The core procedure of the algorithm includes three phases described as follows [3]:

- Parallel coarsening: This phase contains s series of stages in which the original graph $G_0=(V_0,E_0)$ distributed on different processors is coarsened into a series of smaller graphs by computing vertex matches. G_{i+1} is constructed from G_i by collasping the matched incident vertexes into a larger vertex in G_{i+1}. This match-computation course is executed on various processors in parallel.
- Parallel initial partitioning: When a distributed graph small enough G_m has been generated after the first phase, a re-partition is computed in the second phase. Since G_m usually contains only a few hundreds of vertexes, computing a re-partition can be accomplished in a short time.
- Parallel uncoarsening: During the last phase, the new re-partitioned graph is projected back into the original graph by disassembling all the vertexes into a vertex set separately and the partition will be refined along with each projecting stage.

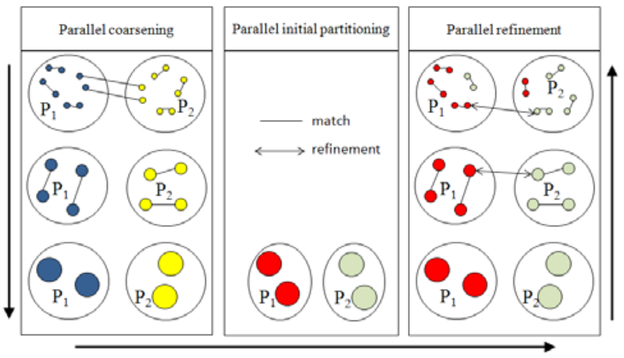

Figure 1. The core procedure of parMetis

Match-computation takes up the most time of the parallel coarsening phase [3]. The original match scheme in parMetis computes a global match for each vertex [4]. During each parallel coarsening stage, a coloring of the entire distributed graph is computed with Luby's algorithm [5] to avoid match conflict by marking adjacent vertexes with different colors. Every stage is made up of several iterations and for each iteration match-computation is performed on vertexes of one of the colors. For vertex v, it will be matched with its adjacent unmatched vertex u that has the highest edge-weight with v and u can be on v's local processor or a remote one. Since no hypothesists are made to the distribution of the original input graph, the worst case can be that all the vertexes are randomly distributed on various processors. Traditional global match scheme has a good universality independent of the distribution of the existed partition. But it is traded with large communication cost introduced not only by the match-computation, but also the coloring procedure. However, when the algorithm is used for mesh-decomposition in OpenFOAM, the structure of the generated mesh should be taken into consideration. In this circumstance, the match scheme should be modified according to the characteristics of the generated mesh.

In this paper, characteristics of mesh decomposed by Simple method in OpenFOAM are analyzed. From the analysis we can conclude that the input decompositions for the parallel graph-partitioning algorithm are usually of good locality. On the basis of that, we propose a new match scheme of the coarsening phase named AMS which combines local match with global match to adaptively compute the vertex matches. Local match scheme only allows vertexes to match with some other vertexes on their local processor, so the coloring of the graph and communication between processors are saved. When the scale of the graph has been reduced to an appropriate degree, the global match scheme is introduced. This new adaptive match scheme can efficiently speed up the coarsening phase of the parallel graph-partitioning algorithm. Experiments are performed based on a general CFD application in OpenFOAM on Tianhe-1A computer system. The result shows that the proposed match scheme in this paper can efficiently improve the execution efficiency of the parallel graph-partitioning algorithm.

II. CHARACTERISTICS ANALYSIS OF GENERATED MESH IN OPENFOAM

To run a CFD application, the hydrofield needs to be dispersed and a mesh generated. The mesh is used to describe the physical space and its overall shape is quite regular and same as the physical space itself consisting of millions of cells. Before using the parallel decomposition method, the generated mesh is decomposed by a method called Simple in OpenFOAM. It decomposes the mesh in different directions following the order of xyz into a specific number of domains according to user's configuration. Taking a 2D mesh-decomposition problem as an example and supposing the decomposition configuration is n_x parts in x direction and n_y parts in y direction, the serial number of the processors is 0-($n_x \times n_y$-1). For every vertex, there are two tags: index_x and index_y. Firstly, all the vertexes are sorted according to their x-coordinates and the mesh decomposed into n_x groups. For vertexes in group i, their index_x tags are i. Secondly the same action is done in y direction and for vertexes in group j, their index_y tags are j. Finally we can compute the owner processor number P of a vertex by the values of its two tags:

$$P = index\_y + index\_x \times n_x$$

A sketch map of the decomposition procedure with Simple method is shown in Figure 2.

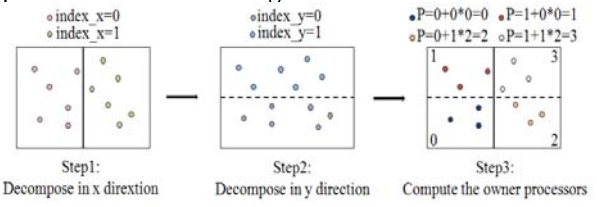

Figure 2. Illustration of Simple method

Comparing with the scale of the mesh, the parallel degree is usually much smaller. Thus after being decomposed by Simple method, each of the domains still contains a very large number of cells. Since the decomposition is based on the cells' geometrical coordinates, all the domains maintain good locality of the mesh. Some domains may contain several disjoint subdomains, but they still hold a large number of cells and these cells form several continuous fields corresponding to the subdomains. So we can conclude that after being decomposed by Simple method, domains of the generated mesh have characterastics of good locality and continuity with large numbers of cells. An example of generated mesh-decomposition by Simple method in OpenFOAM [6] is shown in Figure 3. We can see from Figure 2 and Figure 3 that the result domains of Simple decomposition generally maintain good locality and contain large numbers of cells.

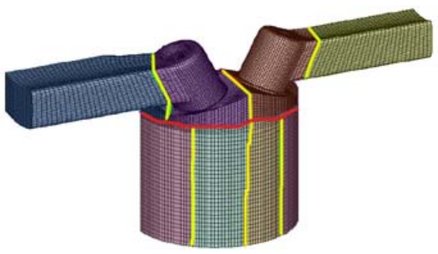

Figure 3. An example of generated mesh in OpenFOAM

III. DESIGN AND IMPLEMENT OF AMS

A. *Ideology of AMS*

From the previous analysis we can know that the input partitioned graphs in OpenFOAM are generally of good locality. Thus during the first several coarsening stages, there is no need to compute a global match for vertexes. A local match scheme is adopted in which each vertex is only allowed to be matched with one of its adjacent unmatched vertexes on the same local processor. The match result of the two vertexes for the next coarsening graph is mapped onto their local processor. Since no inter-processor matches are allowed and the computation on each processor is serial, the coloring procedure is saved. So is the communication cost introduced by match-computation between vertexes on different processors. The local match scheme maintains a good match result while reduce the communication overhead. When the graph has been coarsened with quite a small number of vertexes, the global match scheme can be introduced since the vertexes might need to be matched with some vertexes on different processors to achieve a better coarsening graph. The number of local match coarsening-stages x can be calculated before the coarsening phase according to the scale of the graph and the parallel degree. The parallel coarsening phase based on AMS is presented in Figure 4.

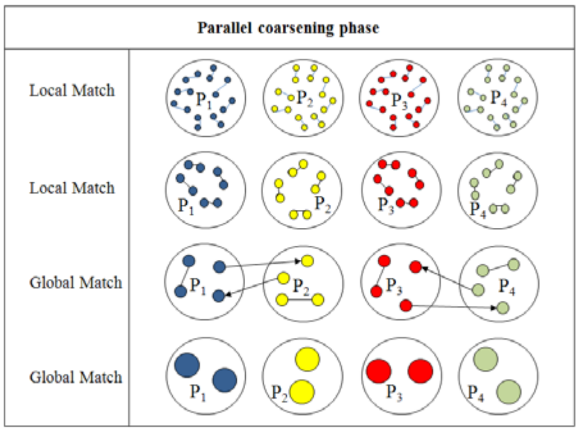

Figure 4. Paralle coarsening phase based on AMS

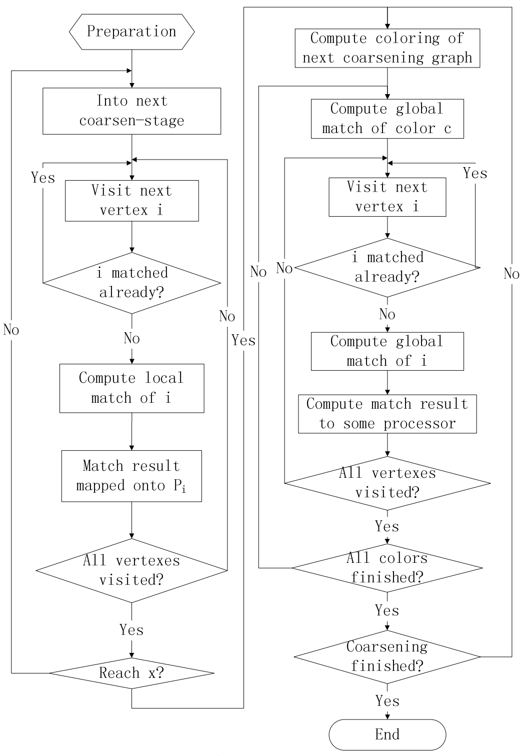

Figure 5. Flow chart of parallel coarsening based on AMS

B. Implement of AMS

The flow chart of coarsening phase based on AMS is shown as Figure 5. For the first x stages of the parallel coarsening phase, local match scheme is adopted. On each processor, the vertexes are visited in a random order. All the unmatched vertexes will be calculated a local match with some vertex on the local processor and the result vertex mapped onto the local processor respectively. For the rest stages, coloring of whole graph is computed firstly. Then global matches are computed for all the unmatched vertexes in a random order on each processor.

The algorithm of the new coarsening process based on AMS is described in TABLE I.

C. Parallel Coarsening Complexity Analysis

Assuming that the number of vertexes in the original input graph is n and the parallel degree p, thus there are n/p vertexes on each processor before the parallel coarsening phase. The maximum total number of boundary vertexes is n/p. For the worst case, every processor has to exchange information of $O(n/p^2)$ vertexes with every other processor and the communication complexity is $O(n/p)+O(p)$.

In parallel coarsening phase based on global match scheme, communication exists in all the $O(\log n)$ coarsening stages and the overall communication cost is

$$O(n/p)+O(p\log n) \ [3].$$

Based on AMS, communication only exists in the last $O(\log n/m)$ stages where m is an integer indicating the percent of the number of stages adopting AMS in the whole coarsening phase. The communication cost for the whole parallel coarsening phase is

$$O(n/mp)+O(p\log n/m).$$

Obviously, communication cost of the parallel coarsening phase based on AMS is lower than that simply based on

TABLE I ALGORITHM DESCRIPTION OF PARALLEL COARSENING PHASE BASED ON AMS

```
1.  x = log_2(nvtxs/1000)/2
      //Compute x. nvtxs: the number of vertexes of the original input graph
2.  while (coarsenstageNo<x)
3.      for (every local vertex i)
4.          if (i is not matched)
5.              compute local match of i with j
6.              mark i and j as matched
7.              match result vertex mapped onto the local processor
8.  while (coarsenstageNo>x and coarsening not finished)
9.      compute coloring of the graph
10.     for (every color c)
11.         for (every vertex i)
12.             compute global match of i with j
13.             mark i and j as matched
14.             compute mapping of match result vertex onto some processor
```

global match scheme which means that AMS can reduce the communication overhead introduced by global match scheme and speed up the execution of parallel coarsening phase.

IV. Experimental Results and Analysis

Experiment on performance of AMS is performed on Tianhe-1A computer system by decomposing generated mesh of LinearPTTwith 38000000 cells which has already been pre-decomposed by Simple method in OpenFOAM. 8 nodes each of which contains 16 processors are used. Each processor has 6 Inter(R) Xeon(R) E5450 cores of 3.00GHz. The decomposition time based on AMS is compared with that based on traditional global match scheme. The decomposition time based on AMS and global match scheme can be seen in Figure 6.

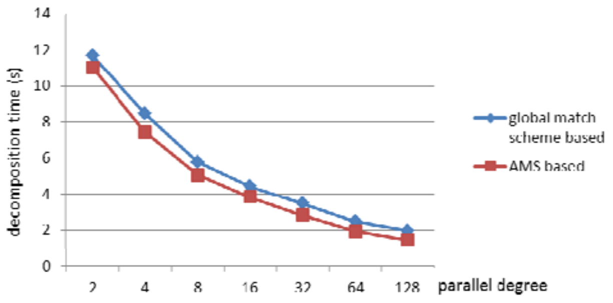

Figure 6 Decomposition time comparison

Number of faces generated by the decomposition and percent of the max number of cells of all the domains above the average number can reflect quality of decomposition in the view of communication overhead and load balance [7]. Comparisons between qualities of decompositions generated based on AMS and global match scheme are shown in Figure 7 and Figure 8.

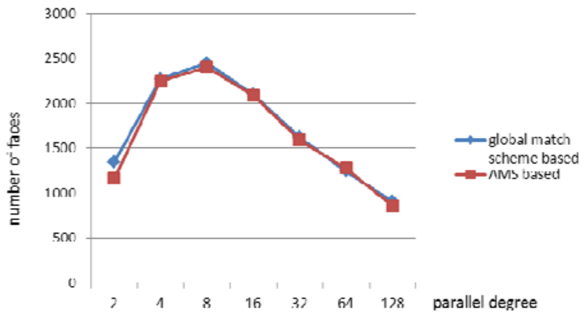

Figure 7 Decomposition quality comparison in communication overhead

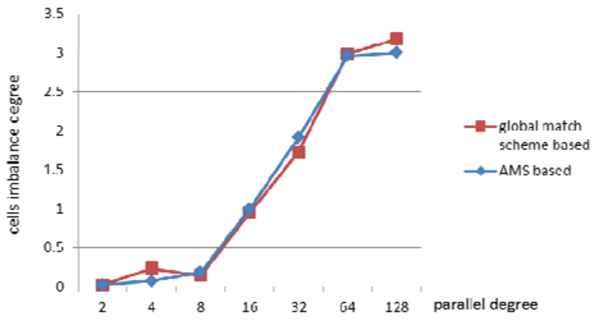

Figure 8 Decomposition quality comparison in load balance

From the experiment results we can see that the parallel multilevel k-way partitioning algorithm based on AMS can improve the decomposition efficiency while maintain good decomposition quality.

V. Conclusion

In this paper, a novel adaptive match scheme AMS is presented for the parallel multilevel k-way partitioning algorithm. An adaptive critical x is calculated firstly according to the scale of the mesh and the parallel degree. For the first x stages of the coarsening phase, a local match scheme is adopted in which vertexes are only allowed to match with local vertexes. For the rest stages, the traditional global match scheme is introduced and match of two vertexes on different processors is allowed. From the complexity analysis and experiment results we can see that AMS can efficiently reduce the communication overhead introduced by simply adopting global match scheme while maintain good decomposition quality. The parallel multilevel k-way partitioning algorithm based on AMS has a better performance than that simply based on traditional global match scheme.

References

[1] Jasak H. OpenFOAM: Programming tutorial [R]. 2007.

[2] Karypis G, Kumar V. A Coarse-Grain Parallel Formulation of Multilevel k-way Graph Partitioning Algorithm [C]. In Parallel Processing for Scientific Computing. 1997.

[3] Karypis G, Kumar V. Parallel multilevel k-way partitioning scheme for irregular graphs [C]. In Proceedings of the 1996 ACM/IEEE conference on Supercomputing (CDROM). Washington, DC, USA, 1996.

[4] Schloegel K, Karypis G, Kumar V. Parallel static and dynamic multi-constraint graph partitioning [J]. CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE. 2002, 14: 219−240.

[5] Luby M. A simple parallel algorithm for the maximal independent set problem [C]. In Proceedings of the seventeenth annual ACM symposium on Theory of computing. New York, NY, USA, 1985: 1−10.

[6] Hendrickson B, Kolda T G. Graph partitioning models for parallel computing [J]. Parallel Computing. 1999, 26 (2000): 1519−1534.

[7] Miao Wang, Yuhua Tang, Xiaowei Guo and Xiaoguang Ren. Performance Analysis of the Graph-partitioning Algorithms Used in OpenFOAM [C]. In Proceedings of International Conference on Advanced Computational Intelligence. 2012.