A Grid-based Flow Analysis and Investigation of Load Balance in Heterogeneous Computing Environment

Soon-Heum Ko[1*], Chongam Kim[1], Oh-Hyun Rho[1], and Sangsan Lee[2]

[1] Seoul National University, Department of Aerospace Engineering,
Seoul 151-742, Republic of Korea

[2] KISTI Supercomputer Centre,
Daejeon, 305-806, Republic of Korea

Key words: Launch Vehicle with Strap-on Boosters, Load Balance, The Grid, Heterogeneous Computers.

Introduction

According to Moore's law, computer speed doubles in every 18 months. In accordance with the development of computational environment, the problem size in CFD(Computational Fluid Dynamics) has been enormously expanded. However, even now, a lot of problems require too huge computational power to be analyzed using local computing resources. As an alternative proposal, the concept 'Grid' was planned and is on research now.

It is obvious that the Grid enables a researcher to analyze a huge-sized problem(e.g. an integral analysis and flow analysis of an airplane). However, diverse communication speed among computing resources and heterogeneity of computing resources can reduce parallel efficiency in the Grid.

Therefore, the present research focuses on the effective flow calculation in the Grid. As an analysis of a huge-sized problem, flowfield around a launch vehicle with two strap-on boosters including base region is analyzed. For an efficient load distribution, performances of all computing resources in the Grid are investigated and, on the basis of performance test results, the whole job is distributed explicitly. As an investigation of load balance in the Grid, a simple load balance algorithm is proposed and applied to the flow calculation around a tangent ogive-cylinder. The proposed algorithm distributes the whole job considering the performance of each processor and communication speed between processors. And the application shows a validity of proposed algorithm.

Governing equations and numerical schemes

The three-dimensional compressible thin-layer Navier-Stokes Equations are adopted as governing equations. As a spatial discretization, AUSMPW+(modified Advection Upstream Splitting Method Press-based Weight function)[3] has been applied and turbulent viscous components are evaluated using the Baldwin-Lomax algebraic turbulence model.

LU-SGS (Lower-Upper Symmetric Gauss-Seidel) scheme is used as the implicit time integration method. The viscous flux Jacobian is neglected in the implicit part since it does not influence solution accuracy, and local time stepping is adopted.

As a mesh generation scheme, Chimera overset mesh is applied in the configuration of a rocket with two strap-on boosters. And the mesh around the core rocket and boosters are generated as multiblock for the inclusion of base region.

*E-mail: floydfan@hanmail.net

Investigation of parallel efficiency in the grid

There have been many researches on load balance algorithm in heterogeneous computing resources. However, in the Grid where dispersed computing resources cooperate, consideration of communication speed between processors should be added. Therefore, load balance algorithm for the Grid should satisfy the minimization of machine-to-machine communication as well as the load distribution considering the capacity of each computing resource. And the algorithm has some assumptions:

1. Network speed between different parallel clusters is slower than the communication speed in the same cluster.
2. A cluster is composed of many nodes that have the same capacity.
3. Computational domain allocated to each processor is rectangular. (In 3-D, a rectangular pillar.)

Then the load balance algorithm in the Grid will be as follows.

Algorithm: Load Balance Algorithm in Multiple Different Clusters.

C : Number of Cluster Machines Used	*t_M : Calculation Time of Processors in the M^{th} cluster*
M : Cluster Number(M=1, ..., C)	*PF_{PN} : Performance of Processor*
N_M : Number of Processor Used in Each Cluster	*W_{PN} : Work Loaded in Each Processor*
TW_M : Total Work Loaded in Each Cluster	*I * J * K : Problem Domain*
PN : Processor Number (PN=1, ..., N_1+...+N_M)	*G_1, G_2 : Group Number*

0. Assumption: *PN* from 1 to N_1 belong to 1^{st} cluster and from N_1+1 to N_1+N_2 to 2^{nd} processors, so on. Calculation time of all processors in the n^{th} cluster is t_n.

1. Performance Test: Execute one iteration of a flow solver and check calculation time. Then performance of each processor is:

$$PF_{PN} = \frac{1}{t_M} \quad \text{where} \quad \begin{matrix} N_1+\ldots+N_{M-1}+1 \le PN \le N_1+\ldots+N_M, \\ M=1,\ldots,C \end{matrix}$$

And work loaded in each cluster is $TW_M = \frac{N_M}{t_M}$

Calculated TW_M are only factors now.

2. Splitting of Total Work into 2 Groups: Find the longest section among I, *J* and K(Suppose *K* is maximum.). Then total work *I*J*K* will be distributed to two parts by splitting *K* into K_1 and K_2.

And, combine all TW_{M^S} into two groups. Then there can exist $\frac{2^c-2}{2}$ different groups.

Of the whole group list, optimized grouping search is needed. Now, a test parameter is introduced :

$$Test = \left| K \times \frac{G_1}{G_1+G_2} - \left[K \times \frac{G_1}{G_1+G_2} \right] \right| * \left| K \times \frac{G_2}{G_1+G_2} - \left[K \times \frac{G_2}{G_1+G_2} \right] \right|$$

The optimized grouping is when *Test* is minimized.

At that time, K_1 and K_2 are: $K_1 = round\ (K \times \frac{G_1}{G_1+G_2})$ *and* $K2 = round\ (K \times \frac{G_2}{G_1+G_2})$, and, group 1 and 2 have total work of *I*J** K_1 and *I*J** K_2, respectively.

3. Distribution of Work to Each Cluster System : Processes in Step 2 are repeated in every group until all cluster system has their own total work.

4. Assignment of Work to Each Processor: Work loaded in each cluster is divided into all processors in the same cluster system. It is achieved simply by applying the method in steps 2 and 3 to processors.

Work assessed in each processor is calculated as follows:

$$W_{PN} = \frac{TW_M}{N_M} \quad \text{where} \quad \begin{array}{l} N_1 + \ldots + N_{M-1} + 1 \le PN \le N_1 + \ldots + N_M, \\ M = 1,\ldots,C \end{array}$$

Flow analysis around a launch vehicle

The flow field around the KSR-(Korea Sounding Rocket) is analyzed in the Grid environment. Fig. 1 shows the pressure contour around the model. The free stream conditions are the Mach number of 1.7 and the Reynolds number of 1.4×10^7 with zero angle of attack. Generated mesh has about 2.7 million mesh points and 20 processors are used as a parallel computation. Plume ejected in the nozzle has the pressure ratio of 1.084 comparing with the atmospheric pressure and is considered as a cold gas.

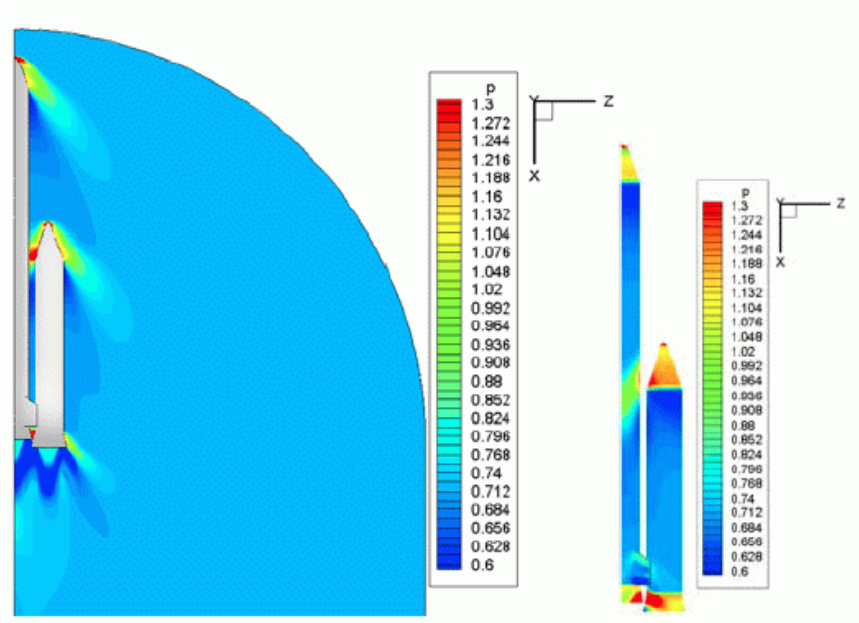

Fig. 1. Pressure contour of KSR.

Application of a load balance algorithm

As a validation problem of the proposed load balance algorithm, a tangent ogive-cylinder of 71 ×91 ×81 mesh points is analyzed by using 7 processors in a local cluster and a PC. Used local cluster has 4 nodes and each node has dual CPUs of P-III., 933MHz. Cooperated PC has a CPU of P-III., 850MHz.

The whole job is distributed to the given processors as shown in table 1. Cooperated resources form 5 groups as each node in a local cluster acts like an individual machine in the Grid. Of these groups, mesh points allocated to a PC is relatively small due to the lower performance of CPU and the most mesh points are allotted to a processor in node 2 where only one processor is used. To prove the validity of the present algorithm, computation time using present mesh distribution is compared with equally distributed case, i.e., 36 ×46 ×41 mesh points are assessed to each processor. Flow solver is the same as used in the analysis of multi-stage rocket, while domain connectivity calculation part is omitted. Flow solver converged after 2139 iterations.

Table 2 shows the comparison of computation time between two distribution cases. Firstly, in the mesh allocation part, it is unavoidable that proposed case demands much more time than the equally distributed case as the former tests the performances and network speeds of

computing resources in the first place, and then distributes the whole work, while the latter only adopts mesh points which is distributed previously.

Time consumption in mesh allocation is retrieved during the iterations of flow solver. In flow solution part, proposed case saves 119 seconds in comparison with equally distributed case. Proposed algorithm gained 145 seconds in calculation time and it is coincident with the result presented in table 1. Result of table 1 shows that a PC has the lowest performance. If it is correct, calculation time in equally distributed case will be dependent on the PC and the ratio of calculation time between two cases will be the same as the ratio of mesh points allocated to the PC. As mesh points are 64124 and 67896(=36 ×46 ×41), respectively, calculation time of proposed case will be 5.7% shorter and the result of table 2 shows 4.1% gain of calculation time. Therefore, it can be said that proposed algorithm is valid in load balance.

However, in communication part, proposed case shows 3.2% increase of communication time and it is by the increase of inter-processor boundary area in PE 6 where the most loads are allocated. Therefore, a further investigation is needed to minimize communication area.

Table 1 Allocation of Mesh

	Group No.	No. of Cells		Group No.	No. of Cells
PE 0 (node1)	1	36×46×41=67896	PE 4 (PC)	4	34×46×41=64124
PE 1 (node3)	2	36×46×41=67896	PE 5 (node4)	3	36×46×41=67896
PE 2 (node4)	3	36×46×41=67896	PE 6 (node2)	5	38×46×41=71668
PE 3 (node1)	1	36×46×41=67896	PE 7 (node3)	2	36×46×41=67896

Table 2 Variation of Computation Time

	Application of the Proposed Algorithm	Equally Distributed Mesh
Total CPU Time	**4331.06787**	**4417.14065**
Mesh Allocation	37.76025	4.50479
Flow Solution	4293.30762	4412.63586
(Pure Calculation)	(3472.85225)	(3618.37337)
(Communication)	(820.45537)	(794.26249)