Multi-GPU computing of moving boundary flow using lattice Bolzmann method

Akira Hatakeyama
Graduate School of Engineering
The University of Tokyo
Tokyo, Japan
a.hatakeyama@cspp.cc.u-tokyo.ac.jp

Takashi Shimokawabe
Information Technology Center
The University of Tokyo
Tokyo, Japan
shimokawabe@cc.u-tokyo.ac.jp

Computational fluid dynamics (CFD) is recently used in a broad field, such as industrial, medical and video. In CFD, the simulation of flow around objects moving freely is called as moving boundary flow. From a practical point of view, it is regarded as one of important themes. As examples of moving boundary flow, in recent year, the simulations of propeller flying or fish swimming are studied.

It is important to simulate accurately and fast in CFD. It depends on numerical scheme and the size of computation mesh. The finer the mesh of CFD is, the more accurate the simulation is. However, when the mesh is made finer, the computing time becomes longer.

We solve moving boundary flow, using immersed boundary-lattice Boltzmann method (IB-LBM) as a numerical method. IB-LBM combines lattice Boltzmann method (LBM) with immersed boundary method (IBM). LBM does not solve Poisson's equation in each time and is easy to carry out a parallel computation because of relatively easy scheme. Moreover, LBM is better at conservation of mass and momentum than other numerical methods, such as finite difference method and finite element method. IBM deals with boundary conditions without changing mesh shape, regardless of the position of objects. Therefore, IBM is compatible with LBM that use orthogonal grid.

In order to compute quickly, parallel computing using GPU is often utilized. In applications using a lattice such as LBM, GPU is better at parallel computing than CPU, and multiple GPU computing leads to fast computation in large scale problem. However, if we use n GPUs in simulation, the calculation time does not become n times faster due to data communication between each GPU. Hence, the communication time needs to be shorten in order to increase the parallel efficiency.

Recent supercomputers installed NVLink that enable to communicate at high speed between GPUs in the same compute node. Nevertheless, when multiple compute nodes are used, data transfer from GPU of one node to GPU of another node goes through CPU and inter-node network. This inter-node communication takes a lot of time. Thus, the parallel efficiency decreases. In addition, computation time of multiple GPU depends on how to split computation domain and how to assign GPUs to each subdomain. In our implementation of IB-LBM, we store data in order of x, y and z direction. In this case, data of z direction communication is continuous, and its access is better. On the other hand, data of x direction communication is not continuous, and its access worse. Thus, the time of z direction communication is shorter, and that of x is longer. The communication time depends on split direction. Thus, It is important to decide which direction to split and how to allocate the process when two dimension split.

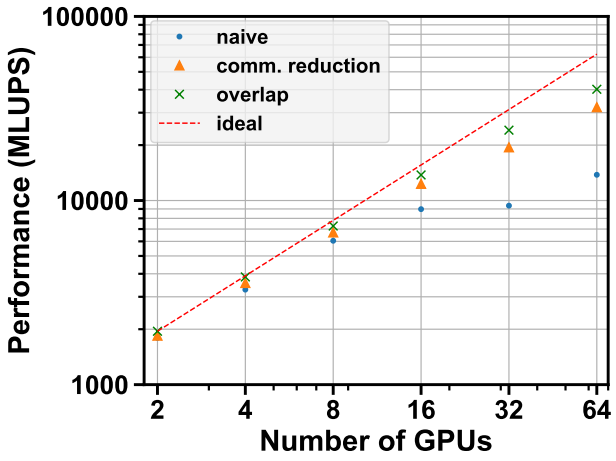

Figure 1: Comparisons of performance of three implementation in weak scaling. The "ideal" in graph represents ideal value of each number of GPUs when a standard is set as value of 2-GPUs.

We implement IB-LBM in three ways and compare the computing time. One implementation is naive. Another implementation is communication reduction. This communicates bare minimum of data to calculate. Its implementation considers the velocity direction of particle, for example. The other implementation is overlap. The amount of data to communicate is same to communication reduction implementation. Besides, this implementation communicate data while computing on GPU simultaneously. Figure 1 shows weak scaling about three implementations on the Wisteria-Aquarius supercomputer at The University of Tokyo, which is equipped with 45 compute nodes. Each node has eight GPUs of NVIDIA A100. As can be seen from Figure 1, overlap implementation improves performance as expected.

In this poster, we provide the implementation of IB-LBM, deal with benchmark problem[1], and show the results of each implementation and how to assign GPUs.

REFERENCES

[1] Kosuke Suzuki and Takaji Inamuro. Effect of internal mass in the simulation of a moving body by the immersed boundary method. Computers & Fluids 49 (2011), 173-187.