Application of GPUs in CFD-based Turbine Wake Simulation

Ji Qi*[1], Kenji Ono*[2]

*Kyushu University, Fukuoka, Japan

[1]qi.ji.558@s.kyushu-u.ac.jp [2]ono.kenji.693@m.kyushu-u.ac.jp

1. Introduction

Understanding wind turbine wakes is a key aspect of wind farm design. The growth of computation capacity has allowed turbine wake research to be conducted on computers using methods based on computational fluid dynamics (CFD). However, such methods involve solving the fluid equations repeatedly on numerous grid points, which makes the performance of computation crucial to both the speed and quality of simulation.

In this study, we explore the potential of GPU as a highly parallel computational device in CFD-based wind turbine wake simulation, and analyze the performance of our program.

2. Mathematical Models

The Large Eddy Simulation (LES) is used for the highly turbulent wind field, which introduces an additional viscosity term to represent the effect of small vortices.

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\nabla p + \nabla \cdot \left(\left(\frac{1}{Re} + \nu_t \right) \nabla \boldsymbol{u} \right) + \boldsymbol{f} \tag{1}$$

$$\nabla \cdot \boldsymbol{u} = 0 \tag{2}$$

Eddy viscosity ν_t is evaluated using the Smagorinsky model. The wind turbine is represented by the force term $\boldsymbol{f}$ using the Actuator Line Model (ALM), which is evaluated according to the Blade Element Theory.

3. Solver Algorithm

The velocity-pressure coupled time integral of equation (1) is evaluated using the Fractional Step Method, which split the integral over one Δt into three steps.

$$\boldsymbol{u}^* = \boldsymbol{u}^n + \Delta t \left(-\boldsymbol{u}^n \cdot \nabla \boldsymbol{u}^n + \nabla \cdot \left(\frac{1}{Re} + \nu_t \right) \nabla \boldsymbol{u}^n \right) \tag{3}$$

$$\nabla^2 p^{n+1} = \nabla \cdot \boldsymbol{u}^* / \Delta t \tag{4}$$

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - \Delta t \nabla p^{n+1} \tag{5}$$

The linear equation resulted from the spatial discretization of equation (4) is then solved by the PBiCGStab algorithm with Jacobi preconditioner.

4. Implementation

We implement the program using NVidia HPC SDK (NVHPC) with each GPU managed by a process on the host side. GPU code is written in CUDA C++ and inter-process communication is implemented with MPI.

To reduce the overhead of inter-process communications, we adopt the overlapping communication method, which overlaps communication with computation to hide the communication time. In this method, the domain of a process is divided into an inner part and a boundary part, while the boundary part needs to wait for the inter-process communication to retrieve the data in the halo region, the computation of the inner part can start without waiting.

The spatial discretization of equation (4) on a three-dimensional structured Cartesian grid produces a well-formed banded coefficient matrix. The coefficient matrix is then represented efficiently by a simple array in our program.

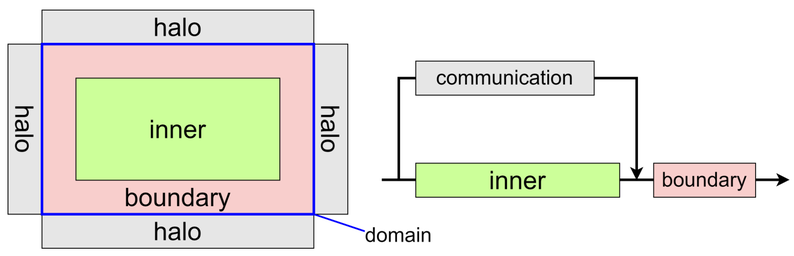

Fig 1. overlapping communication

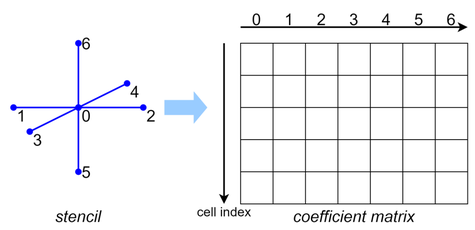

Fig 2. spatial discretization stencil and corresponding coefficient matrix

5. Numerical Result

Numerical simulation of one experimental turbine model is carried out using our program. The result shows that our program is able to produce characteristic features of the wake flow such as the spiral tip vortices.

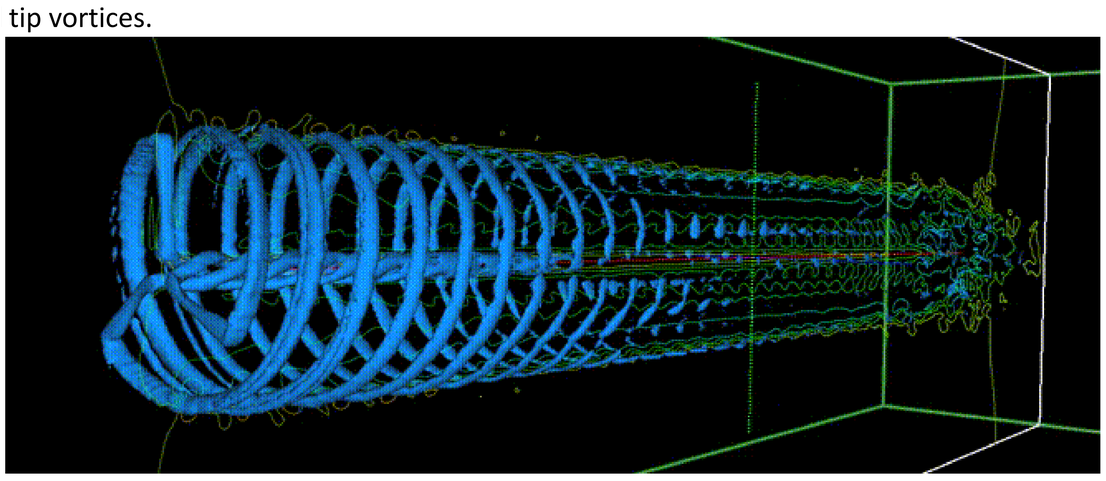

Fig 3. vorticity contour showing the turbine's wake structure

6. Performance Result

Performance evaluation is conducted on Kyushu University's supercomputer ITO subsystem B. Each node in ITO subsystem B has 36 cores in 2 sockets and 4 Tesla P100 GPUs.

Breakdown of execution time on 1 node (4 GPUs) with a grid size of 900x300x300 for 10000 timesteps is taken, which shows that the linear solver for equation (4) is the most time-consuming routine in the program.

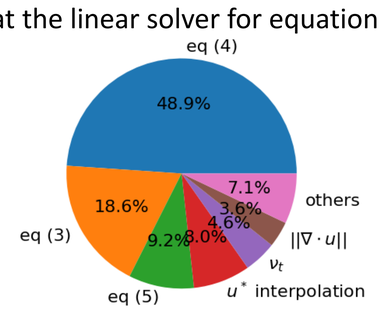

Fig 4. breakdown of execution time

Strong scaling experiment is conducted with a total grid size of 900x300x300 on 4, 8, 12, 16 GPUs for 10000 timesteps, speedup and efficiency of parallelization are calculated based on the results. It is shown that routines with more communication in relation to computation suffer from more performance degradation when the number of GPUs increases. The overall performance is largely determined by the performance of linear solver, which can be explained by the overwhelming importance of the linear solver in the execution time breakdown.

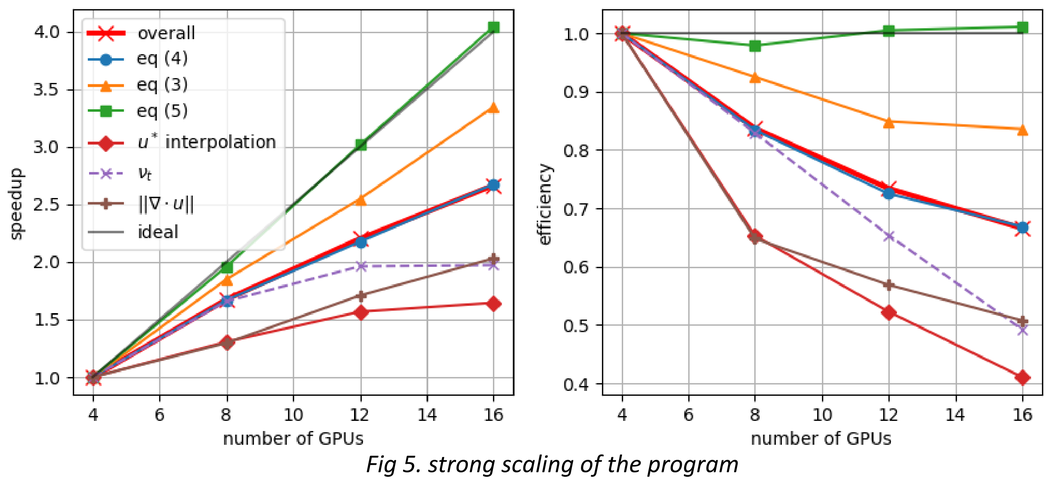

Fig 5. strong scaling of the program

To study to performance of the linear solver, strong scaling results of the PBiCGStab solver are also taken. Pure GPU routines without communication show super-linear scalability, while routines with more global reduction (e.g. inner product and norm calculation) in relation to other operations suffer from the worst scalability.

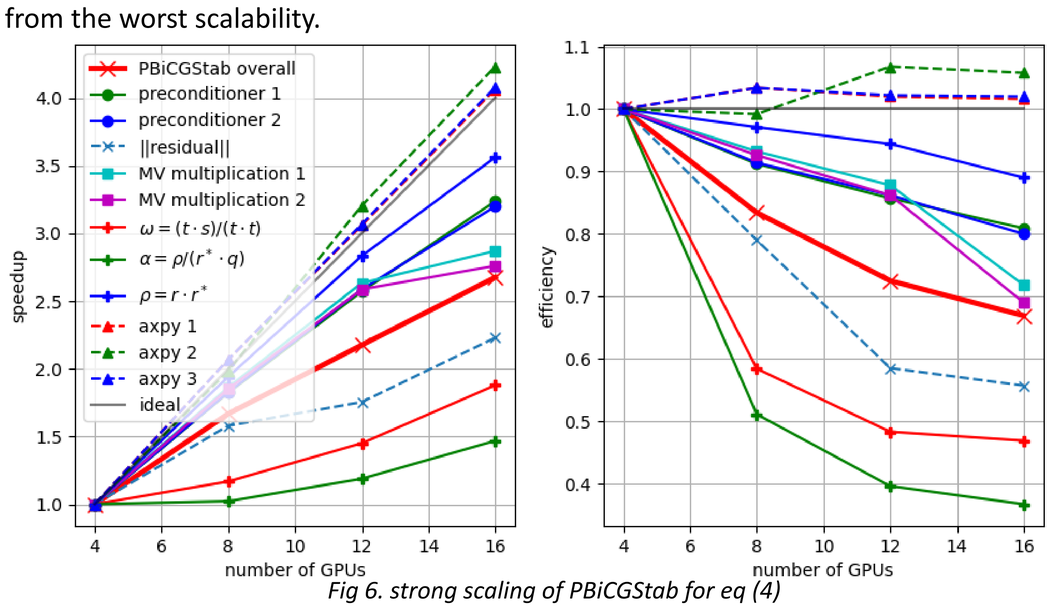

Fig 6. strong scaling of PBiCGStab for eq (4)

Weak scaling experiment is conducted on 4, 8, 12, 16 GPUs with a grid size of 900x300x300 per node (4 GPUs) for 10000 timesteps. Routines that feature heavy computation show good scalability in weak scaling while routines featuring more global reductions suffer from degradation in efficiency.

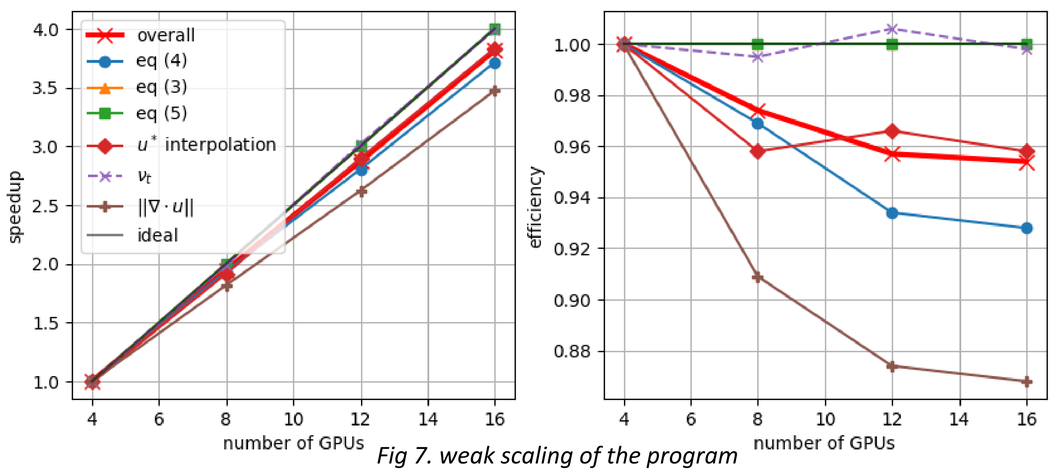

Fig 7. weak scaling of the program

7. Conclusion

In this research, we implemented a multi-GPU program using MPI-CUDA hybrid parallelization for the CFD-based simulation of turbine wake, and studied the performance of our program in detail.

The research demonstrated the effectiveness of the application of multi-GPU technique in wind turbine simulations. Through performance study, we demonstrated the dominant importance of the linear solver for equation (4) to the overall performance of the program, as well as the effect of performance degradation resulted from global reduction operations.

In future research, we aim at improving the performance by reducing the number of global reductions in the linear solver, as well as improving the packing/unpacking operations of MPI communication buffers to reduce communication overhead.

References

1) Takeo Kashijima. 1999. Numerical Simulation of Turbulent Flows. Yokendo Ltd
2) Jens Nørkær Sørensen and Wen Zhong Shen. 2002. Numerical Modeling of Wind Turbine Wakes. Journal of Fluids Engineering 124, 2 (05 2002), 393–399. https://doi.org/10.1115/1.1471361
3) Xi Zhang, Xiaohu Guo, Yue Weng, Xianwei Zhang, Yutong Lu, and Zhong Zhao. 2023. Hybrid MPI and CUDA paralleled finite volume unstructured CFD simulations on a multi-GPU system. Future Generation Computer Systems 139 (2023), 1–16. https://doi.org/10.1016/j.future.2022.09.005