An optimisation of particle information exchange using one-sided communication for the MPS method

Aoto Abe, Kazumi Kayajima, Dai Wada, and Takaaki Miyajima*
{ce235002,ce235050,ce235048,takaaki_miyajima}@meiji.ac.jp
Department of Computer Science, School of Science and Technology, Meiji University
Kawasaki-shi, Kanagawa, Japan

1 INTRODUCTION

The Moving Particle Simulation (MPS) method is one of the computational methods for simulating fluid behaviour, classified as a particle-based method. JAXA has developed an in-house MPS program called P-Flow. It can simulate phenomena such as large deformation and separation of fluids more easily than stencil-based methods. Dynamic domain decomposition is inevitable for large-scale simulations on distributed-memory systems to balance the load of each process [1]. Particles are moved during the computational process, so the particles needed for calculation may exist in another sub-region in another process. In this case, inter-process communication of particle data is required. As a pre-processing step in particle data communication, it must inform how many particles will be transferred from where to where. Collective communication is often used for this step as a naïve implementation. When collective communication is used, the communication time becomes a scaling bottleneck since the elapsed time increases rapidly in proportion to the number of processes.

2 EVALUATION

A test program modelling the particle information exchange is made since the entire program of P-Flow is too large (10,000+ lines in Fortran90). This test program compares and evaluates the time required for MPI communication concerning the proposed technique. The naive implementation uses *MPI_Alltoall*, and the proposed technique uses *MPI_Put*. The test data is extracted from the particle information exchange in *trans_mpi* at the actual run of the MPS method. The data comprises the number of particles, the destination and the source process. The test data depends on the state of domain decomposition and the number of *trans_mpi* calls, so the amount of particle data communicated and the transfer pattern are different for processes 36 and 72.

JAXA JSS3 TOKI-RURI ST node is used for evaluation. The number of nodes used varied from one to nine, while the number of processes remained fixed at 36 or 72. The number of trials was one. Figures 1 and 2 show the execution time of naive implementation and the proposed technique using 72 processes, respectively. The data exchange was completed in a shorter time with *MPI_Put* compared to *MPI_Alltoall*. Specifically, the median time was reduced to 1/3.63 for six nodes with 36 processes and 1/4.38 for nine nodes with 72 processes. The difference in execution time increased as the total number of processes increased, and this difference is considered to be proportional to the number of processes.

In naïve implementation, the computation time increases as the number of nodes increases. This is due to the use of *MPI_Alltoall*, where communication with all processes occurs. On the contrary, the proposed method shows a decrease in communication time as the number of nodes increases. In the two to four-node range,

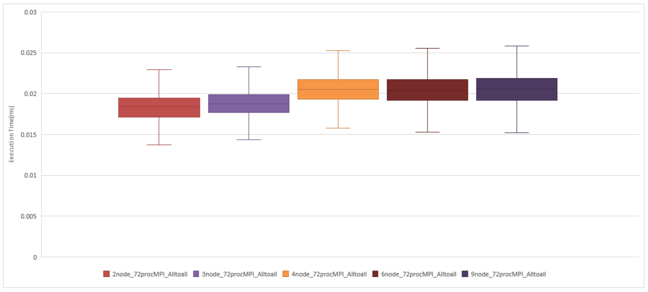

Figure 1: Data communication time for the naïve implementation (*MPI_Alltoall*) at 72 processes.

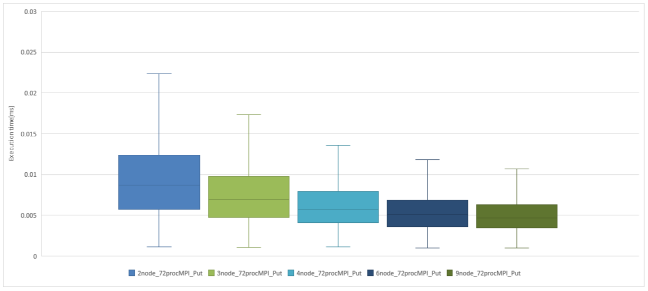

Figure 2: Data communication time for the proposed technique (*MPI_Put*) at 72 processes.

the communication time decreases inversely proportional to the number of nodes. This is possible because the size of the data to be communicated is inversely proportional to the number of nodes mentioned above.

3 CONCLUSION

We propose a technique to reduce the amount of communication using one-sided communication while considering a background of particle movement. Specifically, using one-sided communication (*MPI_Put*) to shorten the communication time of exchanging information instead of collective communication (*MPI_Alltoall*). A test program modelling the particle information exchange is made and used to evaluate the proposed technique. The results show that the proposed technique reduces communication time by 1/3.63 for six nodes in 36 processes and by 1/4.38 for nine nodes in 72 processes.

Acknowledgments This work used computational resources TSUBAME3.0 supercomputer provided by Tokyo Institute of Technology through the HPCI System Research Project (Project ID: hp200100). The experiments were also conducted using the JAXA supercomputer JSS3.

REFERENCES

[1] Kohei Murotani, Seiichi Koshizuka, Tasuku Tamai, Kazuya Shibata, Naoto Mitsume, Shinobu Yoshimura, Satoshi Tanaka, Kyoko Hasegawa, Eiichi Nagai, and Toshimitsu Fujisawa. 2014. Development of Hierarchical Domain Decomposition Explicit MPS Method and Application to Large-scale Tsunami Analysis with Floating Objects. *Journal of Advanced Simulation in Science and Engineering* 1, 1 (2014), 16–35. https://doi.org/10.15748/jasse.1.16