§4. Optimization of Particle Simulation Code for Magnetic Reconnection in Open System

Ohtani, H., Usami, S., Horiuchi, R.

Magnetic reconnection is widely considered to play an important role in energetically active phenomena in high temperature plasmas. In spite of intensive research, many basic questions about the details of mechanisms of reconnection still remain poorly understood. To clarify the relationship between particle kinetic effects and anomalous resistivity due to plasma instabilities in the reconnection phenomena, we develop a three-dimensional particle simulation code for an open system, called "PASMO" [1-3]. For performing the simulation code on a distributed memory and multi-processor computer system, PASMO code at first adopted the particle distribution algorithm, in which the space was not distributed but information of particles was. However, since all field data were diffusely duplicated on each parallel process, the memory was fruitlessly used and the execution performance was saturated when the number of parallel processes increased. To make efficient use of a scalar-type supercomputer "Plasma Simulator", the domain decomposition should be adopted in PASMO code. The algorithm decomposes the domain. For example, the field variable is defined by three coordinates (x, y and z), and we distribute it along z-direction. The processor performs "Field solver" in the mapped domain, and carries out "Particle pusher" for particles which exist in the domain. In this case, the performance is expected to grow as the number of processes is increased.

In this paper, we investigate the performance of PASMO which adopts the domain decomposition algorithm with the periodic boundary condition along three dimensions. For the distributed parallel algorithm, we use Massive Parallel Interface (MPI).

In order to check the performance of PASMO, we test-run under two parallel calculation conditions.

In the first case, we perform the simulations, in which the simulation box has $16 \times 16 \times 1024$ grid points and the number of particle per cell is 100. The numbers of domain decompositions (N_{mpi}) along z-direction are set to be 2, 4, 8, 16, 32, 64 and 128. Figure 1 shows the dependency of calculation time per step on N_{mpi}. The time is efficiently decreased until $N_{mpi} = 32$, but when $N_{mpi} > 32$ the time is increased. It is considered that increase of the cost of data transfer causes the growth of the simulation time when N_{mpi} increases.

In the second case, when N_{mpi} is increased such as 2, 4, 8, 16, 32, 64 and 128, the grid number of z direction is also increased as 32, 64, 128, 256, 512, 1024 and 2048. The number of particles per cell is 100. We compare the simulation times per step in Fig. 2. The time grows at mostly in proportion to N_{mpi}. This reason is considered that the calculation cost of Poisson solver is increased as the simulation domain is expanded. The solver is not distributed under the domain decomposition algorithm.

In order to perform larger-scale simulation using a more massive parallel computer system, we need to improve the PASMO, moreover. For example, we now solve the Poisson solver globally. In stead of the global Poisson solver, it may be a good way to use the Exact Charge Conservation method, in which the charge is locally conserved [4].

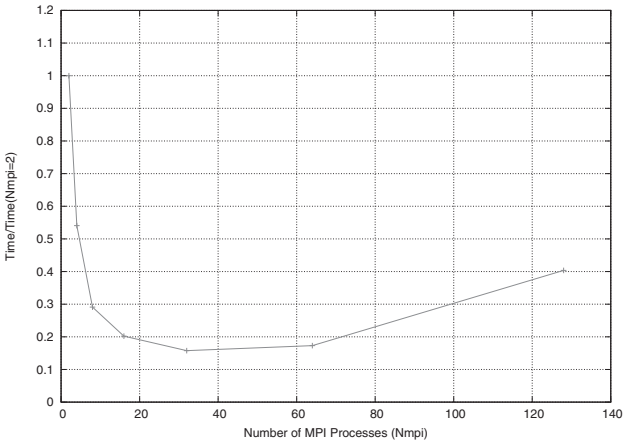

Fig. 1. The dependency of the simulation time per step on the number of parallel process N_{mpi} in the first case.

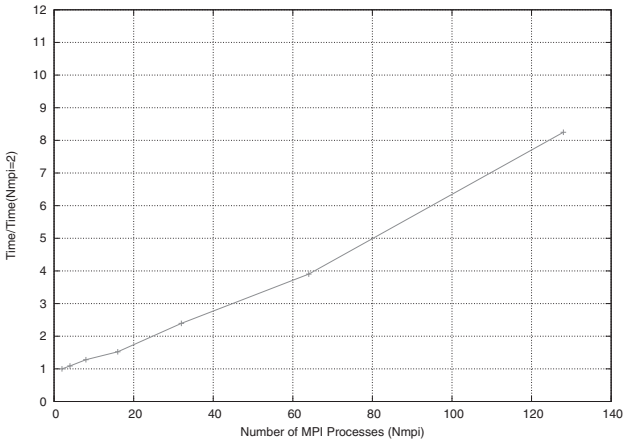

Fig. 2. The dependency of the simulation time per step on the number of parallel process N_{mpi} in the second case.

1) Horiuchi, R. *et al*: Phys. Plasmas **6** (1999) 4565.
2) Ohtani, H. *et al*: LNCL **4759** (2008) 329.
3) Ohtani, H. and R. Horiuchi: PFR **4** (2009) 024.
4) Esirkepov, T. Zh.: CPC **135** (2001) 144.