An adaptive load balancing method for particle-in-cell simulations

*Yosuke Matsumoto[1]

1. Graduate School of Science, Chiba University

Particle-in-cell (PIC) simulations have been used for understanding particle accelerations, particle transport and magnetic field generation in space and astrophysical phenomena. The PIC simulation bases on the Vlasov equation in which the distribution function of each particle species evolves in the phase space in response to the electric and magnetic field time evolution. While the evolution of the distribution function is represented by motions of finite numbers of macro particles, the electric and magnetic field evolution are represented by a disretized form of the Maxwell equations; the PIC simulation is composed of Eulerian and Lagrangian methods.

Using massively parallel supercomputer systems with a parallelized PIC code is a powerful way to elucidate such nonlinear phenomena. When using very large numbers of processor cores (say, greater than 1000 cores), the code needs to adopt a domain decomposition method, in which each MPI process is associated with a portion of the whole region, for an efficient parallelization. Although this parallelism introduces somewhat cumbersome implementations to the code for exchanging arbitrary numbers of particles among MPI processes every time step, resulting high scalability allows us to examine with unprecedentedly large-scale simulations. Indeed, we have elucidated important acceleration mechanisms in collision-less shocks by using the Japanese flagship supercomputer system with a hundred of thousands of processor cores (e.g., Matsumoto et al., 2017).

In this presentation, we report an upgrade with an implementation of an adaptive load balancing method to our PIC code. The load imbalance among MPI processes in PIC simulations arises if particles were in-homogeneously distributed in the simulation domain as a result of time evolution. This imbalance becomes problematic when using very large numbers of MPI processes (say greater than millions of cores), and we expect to meet this situation with the next generation Japanese flagship system called FUGAKU. A few approaches have been proposed and implemented in PIC codes. For example, in PSC and SUMILEI codes (Germaschewski et al., 2016; Derouillat et al., 2018), each domain of a MPI process is further divided into sub-domains, and these "patches" are exchanged among the processes so that the total number of particles in each MPI domain is equally distributed. The exchange of patches are determined according to the Hilbert curve passing through them. The resulting shape of each domain is rather irregular, which complicates inter-process communications. Alternatively, we adopt the recursive multi-section algorithm which have been successfully implemented into the cosmological N-body simulations (Makino, 2004; Ishiyama et al., 2009). This method requires rather simple inter-process communications because each MPI domain is a rectangular shape. We successfully implemented this method to the PIC code for the first time with benchmark tests of the Weibel instability and collision-less shock simulations.

Keywords: particle-in-cell simulation, load balancer, high performance computing