

Algorithms to use accelerators effectively with FDPS and performance of applications

*Daisuke Namekata¹, Masaki Iwasawa¹, Keigo Nitadori¹, Kentaro Nomura¹, Long Wang^{3,1}, Miyuki Tsubouchi¹, Junichiro Makino^{2,1,4}

1. RIKEN Center for Computational Science, 2. Department of Planetology, Graduate School of Science, Kobe University, 3. Helmholtz Institut für Strahlen und Kernphysik, 4. Earth-Life Science Institute, Tokyo Institute of Technology

Numerical simulations based on particle methods have played an important role in understanding the formation and evolution of planets and planetary systems. Along with the improvement of the computing capacity of computers, more large-scale simulations and simulations taking into consideration various physical phenomena have come to be performed, in order to understand the reality in more detail.

However, it is becoming increasingly difficult to develop a simulation code that runs efficiently on modern HPC (high-performance computing) systems. The main reason for this difficulty is that modern HPC platforms have become very complex, and thus it requires lots of efforts necessary to develop complex programs to make efficient use of such platforms. Typical modern HPC systems are actually a cluster of computing nodes connected a network with a limited communication bandwidth. The number of nodes can reach up to 100,000 in the largest systems at present. Thus, we need to develop simulation code so that it shows a good load balance and requires minimal communication. In recent years, the architecture of large-scale HPC systems have been shifting from homogeneous multi-core processors to accelerator-based systems and heterogeneous multi-core processors for economical reasons. This makes code development more difficult for the following reasons. One is that for many applications, the communication bandwidth between CPUs and accelerators becomes the bottleneck. The second one is that because CPUs and accelerators have separate memory spaces, the programming is complicated and we cannot use existing programs. Thus, the time required for code development has been increasing more and more, which causes stagnation of research.

To improve the situation, we have developed a framework called FDPS (Framework for Developing Particle Simulators), which enables researchers to develop their own high-performance parallel particle-based simulation programs easily. The basic idea of FDPS is to provide a high-performance implementation of parallel algorithms for particle-based simulations in a "generic" form, so that researchers can define his/her own particle data structure and interparticle interaction functions and supply them to FDPS framework. FDPS framework compiled with user-supplied data type and interaction function provides all necessary functions for parallelization, and using those functions researchers can write their programs as though they are writing simple non-parallel program which runs on their laptop computers. FDPS offers very good performance on large-scale parallel systems consisting of "homogeneous" multi-core processors, such as K computer and Cray systems based on x86 processors. In FDPS version 2, we have extended FDPS so that it can use accelerators by writing the interaction function that uses the accelerator hardware. However, the efficiency was limited by the latency and bandwidth of communication between the CPU and the accelerator and also by the mismatch between the available degree of parallelism of the interaction function and that of the hardware parallelism, and thus would be rather poor.

In order to make it possible for FDPS to use more efficiently accelerator hardware such as GPGPUs

(General-purpose computing on graphics processing units), we introduced a new interface of user-provided interaction function. We also implemented new techniques which reduce the amount of work on the side of CPU per timestep and amount of communication between CPU and accelerators. We have measured the performance of N-body simulations on a systems with NVIDIA Volta GPGPU using FDPS and the achieved performance is around 27 % of the theoretical peak limit.

Keywords: Computational Science, Software development, Accelerators