Parallel Particle-In-Cell Plasma Simulations on GPU, Vector, and Scalar System

Tooru Sugiyama[1]

[1]Earth Simulator Center, Japan Agency for Marine-Earth Science and Technology (JAMSTEC), Japan

1 Introduction

Particle simulation model is quite useful tool to investigate particle kinetics in plasma physics. The motions of the charged particles are self-consistently calculated in the electromagnetic field [1]. The simulation step in an explicit Particle-in-cell (PIC) model consists of three parts; particle acceleration and motion where the Lorentz force is evolved on the particle's position by interpolating the electromagnetic field defined on grids; field solver where Maxwell's equations are solved on grids; and velocity moment calculation where the particle flux and density are calculated from the particle velocity and position. The number of the super-particles is huge, for example, over hundred particles are allocated in each simulation cell. Resultantly, the calculation cost of the PIC simulation is quite higher than that of the Magneto Hydro Dynamics (MHD) model. The highest cost in the particle simulation is the velocity moment calculation. This is because the particles are randomly distributed in the simulation system and they are related with tidily located grids. Therefore, this procedure also requires random access to the physical memory on the computational system. Then, the cache on the CPU chip does not work well in this procedure. However, the slow data transfer band width between CPU and DDR3 memory (36 GB/sec.) doesn't become the bottleneck of the calculation speed. Because the number of the moment calculation procedure is quite larger than that of the data transport. This coarse-grained model is appropriate to the calculation on GPU and Vector-type system.

The GPU chip on nVIDIA C2070 board has 448 CUDA cores, and then we need to use a parallelized algorithm to make a good performance on the chip. The vector type system also has a large vector register on the chip, where we need also to use a parallelized algorithm to make a good performance. Here we use the particle and domain decomposition methods for parallel calculation. Since the particles are randomly distributed in the simulation system as noted above, the moment calculation is essentially the atomic calculation. To avoid this memory conflict, some algorithms are introduced [3] [4] [5]. Here we have applied these algorithms and implement them on GPU, Vector-type, and Scalar-type systems. We have measured the calculation times.

2 Simulation Model

We treat an ion beam instability using a three-dimensional hybrid simulation model where the ions are treated as macro-particles while the electrons are treated as a charge-neutralizing massless fluid [2].

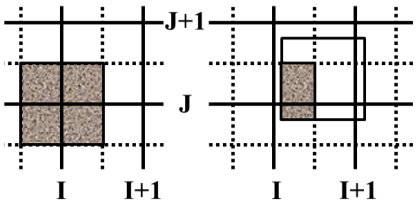

Fig. 1. Poster of JSST 2012.

In the moment calculation, we count how many particles are located in a cell with the particle shape function (Fig. 1), which is a similar procedure as the counting histogram. Note that the figure shows the first order accuracy model in two-dimensional system for convenience. When we count the number of particles using multi-processors, cores, or on the vector registers, (hereafter thread as GPU language) with the particle decomposition method, the memory conflict takes place, because each thread treats each particle. Suppose that some particles located in a certain cell and these particles are counted by different threads. These threads try to modify the memory for counting in the cell if these particles locate in the same cell. For the safety counting, the read-modify-write procedures must be operated without interruption by other threads. If not, the counting results are not correct. That is, while one processor counts a particle moment, other processors should wait until the counting finishes. This waiting reduces the parallel performance on multi-thread system, especially on the highly parallelized system. To avoid the conflict, we apply some algorithms, those are, SORT, RETRY and LIST algorithm.

3 Performance Measurements

The detailed results for the performance measurements are reported both on the followed full paper and the conference presentation.

4 Discussion

We have introduced SORT, RETRY and LIST algorithms in order to perform plasma PIC simulation on multi-core and vector-type systems. The most complicated procedure in PIC simulation on highly parallelized platform is the moment calculation. Here, we apply these methods for the moment calculation.

The most useful point using the SORT method is that the cache works well on both calculation of the moment and velocity advances. This is because the same field and moment values are used for the sorted particles. On the other hands, the most useful point using RETRY and LIST methods is that the particle sorting procedures are excluded. One of the reasons to perform the PIC simulation is that particle trajectories are followed in the self-consistent electromagnetic-fields in order to investigate the particle origin of the attracted particles. By sorting the particle ID number with the particle position in the simulation system, it makes complicated to follow the particle trajectories.

Here, we have not arranged the simulation programs to deal with the general number of system size. Indeed, the system size is strongly related with the number of threads in one thread-block on GPU as 1024, and also the number of the vector-register as 256. Therefore, there would exist the optimized number for the system size (e.g. the number of grids, the number of particles per cell) in order to maximize the performance.

In addition, we have realized one challenging problem on multi-core systems. The sorted particle ID-list created by the multi-core system is different in each run, even though the same program are used. It certainly leads the round-off error in calculating the particle moments in a different order of the particle ID number. If the results are sensitive to the round-off error, we never have the same results. Now we don't have any solutions about the error, but the ion beam instability tested here is acceptable for the error.

References

[1] C. K. Birdsall, & A. B. Langdon, 1991, Institute of Physics Publishing, Bristol and Philadelphia.

[2] Fujimoto, M., 1992, PhD thesis, Institute of Space and Astronautical Science, Japan.

[3] Imamura, T., 2005, IPSJ journal, 46, SIG 7 (ACS 10), 52-62.

[4] Sugiyama, T., et al. 2004, IPSJ journal, 45, SIG 6 (ACS 6), 171-175.

[5] Sugiyama, T., 2011, JAMSTEC Rep. Res. Dev., 12, 13-25.