Porting the 3D Gyrokinetic Particle-in-Cell Code GTC to the CRAY SX6 Vector Architecture: Perspectives and Challenges

Stephane Ethier[1], Zhihong Lin[2]
[1]*Princeton Plasma Physics Laboratory, Princeton, NJ*
[2]*University of California Irvine, Irvine, CA*

Several years of optimization of the GTC code on the super-scalar architecture has made it more difficult to port the current version of the code to the SX6 vector architecture. Code structures that are beneficial and optimal for a super-scalar processor can result in poor performance on a vector processor, although small modifications can often fix the problems in a lot of cases. This paper explains the work that has been done to port the GTC code to the SX6 computer and to optimize the most time consuming parts. Early performance results are shown and compared to the same tests done on the IBM SP Power 3 and Power 4 machines. The work has been done on the 8-cpu SX6 at the Arctic Region Supercomputing Center (ARSC) so only single-node runs are presented, i.e. from 1 to 8-processor runs. GTC is a mixed-model MPI/OpenMP code that usually runs on 1024 processors (at NERSC) for large production simulations, but small problem cases can be done on a limited number of processors while keeping the amount of work per cpu the same as for larger runs. We show that small changes to the code can lead to significant improvements, but that the particle-in-cell method in itself still remains difficult to optimize.

The particle-in-cell (PIC) method has always been notoriously challenging for vector processors. The method makes use of particles to sample the distribution function of the plasma system under study, and these particles interact with each other only through a self-consistent field so that no binary forces need to be calculated. This saves a great deal of computation since it scales as N instead of N^2, where N is the number of particles. At each time step, the charge of each particle is distributed among its nearest grid points according to the current position of that particle; this is called the "scatter" operation. The Poisson equation is then solved on the grid where the charges have been deposited in order to get the electrostatic potential on the whole grid. For each particle again, we calculate the value of the field (force) at its position from the values at the nearest grid points; this is the "gather" operation. Now that the force on each particle has been updated, we just need to move the particles for one time step. These steps are repeated until the end of the simulation.

The big challenge in vectorizing this algorithm is the fact that the particles can be anywhere in the system and that 2 particles located next to each other at one time will most probably not be next to each other in the array that holds the information about these particles. This leads to memory conflicts during the charge deposition (scatter) step because two vector operations try to write to the same memory location, hence preventing the vectorization. This paper explains how to circumvent this problem by using temporary arrays to achieve the full vectorization of the scatter operation. It also discusses the cost of using these arrays.

Work Supported by DOE Contract No.DE-AC02-76CH03073 and in part by the DOE SciDAC Plasma Microturbulence Project