ハイパフォーマンス
コンピューティング 71－4
（1998．5．15）

Pentium Pro Clusterを使った並列粒子コードのベンチマーク

陸全明，蔡東生

筑波大学電子情報工学系

Particle-In-CellもしくはParticle-In-Mesh粒子コードの並列化は多くの最新の並列計算機で研究されてきている．また，近年安価で手軽に購入できるPentium Pro, Pentium II搭載のパソコンも普及してきた．これらのパソコンの計算性能は理論的には一世代前もしくは少し古い並列計算機と単体CPUで同程度以上の性能を誇っている．本研究では手軽に手にはいるPentium Proパーソナルコンピュターをネットワークにつなぎ並列性能と単体性能を粒子コードを評価してみた．

Parallel-Particle-In-Cell code using PentiumPro Linux Cluster

QuanMing Lu and DongSheng Cai

The university of Tsukuba, Institute of Inflrmation Sciences, Tsukuba, 305, Japan
Email:{qmlu,cai}is.tsukuba.ac.jp

Particle-In-Cell or Particle-In-Mesh code has been parallelized in many advanced parallel computers. On the other hand, many new high-performance PCs have been introduced recently. In this report, using our test Skelton PIC code we have measured the performance of these personal PCs in parallel and in single.

1 Skelton PIC code

Ever since the emergence of parallel computers, particle-in-cell (PIC) or particle-in-mesh (PIM) particle simulation has been recognized as a practical tool that scientists in disciplines such as fluid dynamics and plasma sciences can use to study the complex dynamics of such particles as air molecules or sub-atomic ions and electrons. A usual PIC code maps a spatial simulation domain onto a grid. Particles are represented as moving within the grid, while both the properties that are tracked by the grid points and by the particles are updated. On a parallel computer, one can decompose either the particles or the grids onto the processors as the primary decomposed data structure. In the first case, each processor is responsible for tracing the properties associated with the assigned particles, which we will call the particle data. The properties tracked at the grid points, which we will call the grid data, are made available to the particles, which is so-called a 'gather' part of the computation. On the other hand, if the grid is chosen as the primary data structure, each processor is responsible for keeping track of a subsection of the simulation domain, and the particle data are made available to the grid points, which is so-called a 'scatter' part of the computation.

A skeleton PIC code has been proposed by Decyk [1] as a testbed where new algorithms can be developed and tested and new computer architecture can be evaluated. This code has been deliberately kept minimum, but they include all the essential pieces for which algorithms need to be developed. The code contains the critical pieces needed for depositing charge, advancing particles, and solving the field. The code moves only electrons, with periodic electrostatic forces obtained by solving Poisson's equation with the fast Fourier transforms. The code uses the electrostatic approximation and magnetic fields are neglected. The only diagnostic is particle and field energy. The basic structure of the main loop of the skeleton code is illustrated in Figure 1. The grid data are chosen as the primary data structure and decomposed into a one-dimensional processor array.

2 Two-dimensional Skelton PIC code benchmark results

The skelton PIC code has been benchmarked in many advanced parallel computers [1]. We have developed a network-clustered Pentium Pro PC system with Linux operating system to measure the performance of the two-dimensional skelton PIC code. The PCs are HP Vectra UX with two 200 MHz Pentium Pro processors SMP and 64 Mbyte EDO DIMMS memories. The networks are connected via 10 Base-T HP switching Hub. As shown in Fig. 2, the benchmarks have been done in three cases. In (1) the first one and (2) the second one, we used the GNU f77 compiler that is essentially a GNU c preprocessor and comes with Linux operating system with free. The second one is tune-uped code for the RISC processor [2]. In (3) the third case, we used pgf77 instead of Gnu f77. Here pgf77 is a commerical version of f77 compiler from Portland Group Inc. As indicated in the figure, the networked-PC-clusters are nothing so bad comparing with the theoretical peak performace of Power PC processor used in SP-2. As indicated in the Table 1, they are, respectively, 266 MFlops and 200 MFlops. In the table we also show that some new benchmarked results of RISC-optimized two–dimensional Skelton PIC code (private communication with Professor Viktor Decyk of UCLA). In due course, in Fig. 2 we expect the SP-2 bechmarked result should be 66% faster than the one shown in the figure if we use the RISC-optimized code (indicated in Table 1).

Reference

1.V. K. Decyk, Computer Physics Communciation, Vol.87, p.87-94,1995.
2.http://olympic.jpl.nasa.gov:80/Reports/SNOptPics.html.

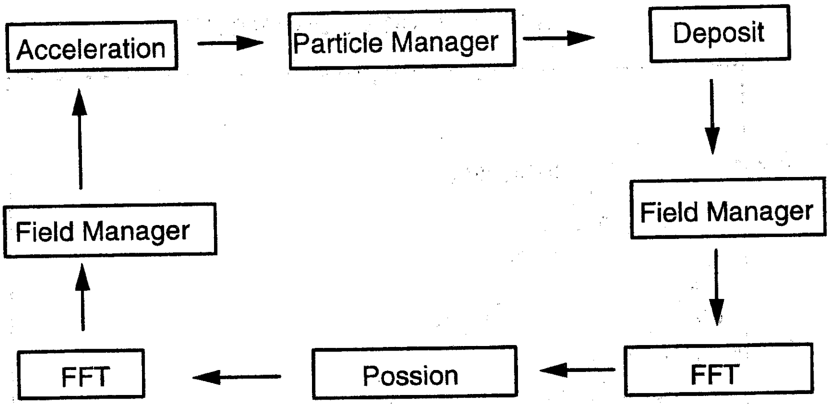

Fig.1. Structure of the main loop of skeleton codes.

Single RISC Processor Benchmark

Computer name	Compiler & option	"dusty deck" version(MFlops)	RISC-optimized version(MFlops)	Theoretical (MFlops)
Cray T3E-900	cf 77 ?	63	188	?
IBM SP2	xlf ?	59	98	266
Cray T3D	cf 77 ?	17	48	150
Intel Pentium Pro/200	unknown	36	43	200
	Gnu f77	10	13	
	Gnu f77 –O3	26	37	
	pgf77	31	45	
	pgf77 –O2 –Munroll –tp p6 -Mnofrmae	36	47	

Table 1: Benchmarks run with a 2D electrostatic PIC code, comparing Intel Pentium Pro/200 with some other machines used in high-performance computing.

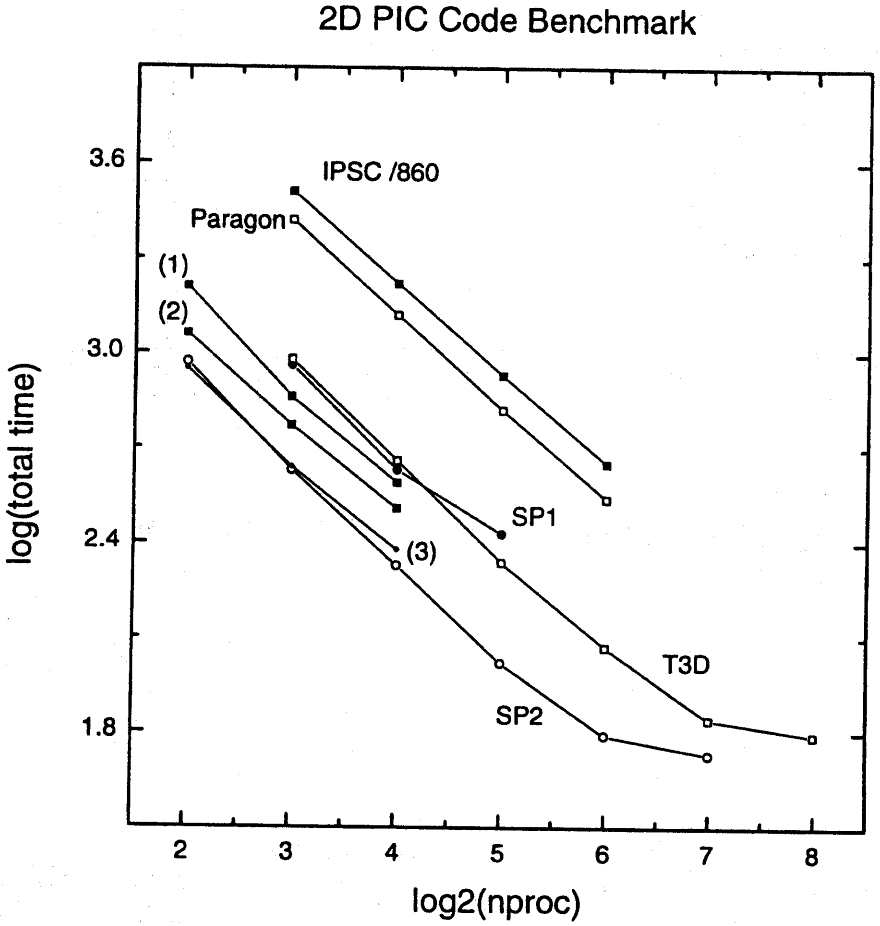

Fig. 2: Total time versus number of processors on log-log sclale for various machines for the 2D benchmark. (1), (2) ,(3) are benchmarks with PentiumPro Linux Cluster. (1) is benchmark with "Gnu f77 -O3" with no RISC-optimization, (2) is benchmark with "Gnu f77 -O3" with RISC-optimization, (3) is benchmark with "pgf77 -O2 -Munroll -tp p6 -Mnoframe" with RISC-optimization.