

COMP 633: High Performance Computing

Programming Assignment 1: All-pairs n-body simulation

Fall 2007

Assigned: Thu Sep 13, 2007

Due: Tue Oct 2, 2007

1. Problem Description

We consider a very simple simulation of the trajectories of n bodies interacting through the gravitational force in 2-space. We start with a collection of bodies b_1, \dots, b_n . Each body b_i is described by a scalar mass m_i , a vector position in 2-space r_i , and a vector velocity v_i .

For a given configuration of bodies, the force f_{ij} on b_i as a result of its interaction with b_j is

$$f_{ij} = \frac{Gm_i m_j r_{ij}}{|r_{ij}|^3}$$

with $r_{ij} = r_j - r_i$ and $|r_{ij}|$ = the length of r_{ij} . The universal constant G is $6.673E-11 \text{ m}^3/\text{kg}\cdot\text{s}^2$ in the MKS system (where the units are meters, kilograms, and seconds). The total force f_i experienced by b_i as a result of interactions with all other bodies is

$$f_i = \sum_{\substack{1 \leq j \leq n, \\ j \neq i}} f_{ij}$$

To simulate the positions of the bodies over time, we update the positions and velocities of the bodies in discrete time steps of size Δt . Using $F = ma$, the change in velocity of b_i over time step Δt is

$$\Delta v_i = \left(\frac{f_i}{m_i} \right) \cdot \Delta t$$

Using a simple integration scheme we update the position and velocities of b_i as follows

$$r_i' = r_i + v_i \cdot \Delta t$$

$$v_i' = v_i + \Delta v_i$$

The problem is to compute the final state of the system following k iterations of this scheme for a given configuration of n bodies.

2. Programming Assignment

- (a) Start by implementing the simple algorithm described above using C/C++ (or Fortran) on a single node of the Dell Linux cluster (topsail), and use OpenMP directives to obtain parallel execution. To measure the performance of your program, determine the interaction rate defined as

$$R(n, t_k) = \frac{n(n-1)}{t_k}$$

where $t_k = t/k$ is the average time per iteration and t is the total time to advance the system simulation k time steps. The time to create the initial state is not included in t . You must be able

to run with $k > 1$ to preclude solutions that work well for only a single iteration, but choose k as small as possible to get consistent values for t_k .

Once you have written and debugged your initial program, spend some time trying to improve the interaction rate. The result of this effort should be a graph of n versus the interaction rate for all values of $1 \leq p \leq 8$ (each topsail node contains memory shared by two quad-core processors). Your graph should also show the interaction rate of your best sequential implementation. Choose the maximum value of n so that your fastest run requires a few seconds.

- (b) Next, consider the following potential optimization. Newton's third law, $f_{ij} = -f_{ji}$, may be used to halve the number of force evaluations. Whenever f_{ij} is computed, its contribution may be added to f_i and subtracted from f_j , eliminating the need to evaluate f_{ji} . Construct a parallel program that implements this reduction in the total work and report its performance using the interaction rate as defined above. Therefore, if everything goes perfectly, you might expect the interaction rate to double using this optimization. Graph the interaction rate for the same values of n and p (and the best sequential implementation with the optimization) as before.

3. Computational check

Our simple n-body simulation suffers from some numerical problems, so it is very difficult to provide trajectory values that are reproducible across implementations. It will help to keep the timestep reasonably small compared to velocities, e.g. $\Delta t = 0.00001$ and velocities no more than $10\Delta t$ m/s. You can perform some simple computational checks on your program. For example, the total momentum of the system should be conserved. This means the vector quantity

$$\sum_{1 \leq i \leq n} m_i v_i$$

should remain approximately the same over all timesteps. It will not be exact because of the discretization of time and roundoff errors. You can disable these computational checks when evaluating performance.

A reasonable starting configuration is to space bodies regularly in the plane with reasonable velocities as suggested above. You should use 64-bit floating-point values for the representation of the configuration and in all calculations. The effect of gravity will be minimal at these time and length scales. If you want to see more gravitational action, increase G by 5 or 6 orders of magnitude.

4. How to proceed

You may work on your own or together with one partner in the class. Develop and test your programs on topsail.

The deliverables for this project are a description of your algorithms and performance graphs as described above. The basic assignment is not very time consuming, but you should budget additional time to investigate performance improvements, since performance will be one of the grading criteria.