CX 4010 – Fall 2015

Problem Set 4

[This will be the last problem set of the semester.] Large software is typically developed in teams. In this problem set, you will work in teams of 3 to write a simple particle simulator. You must form your own teams. Each team member must write approximately the same number of lines of code, as described below. Since all team members write code, this forces teams members to partition the code in a way that minimizes the couplings/interactions between the parts, which makes code development and testing simpler.

Due dates

Sunday, Nov. 1, 11:59 pm. For each team, one member of the team submits a proposed list of functions to be written. For each function, provide (1) full function prototype including all parameters and return values and their types, (2) a description of what the function does, including a description of what are the inputs and what are the outputs, (3) the proposed author of the code.

Sunday, Nov. 8, 11:59 pm. For each team, one member of the team submits a testing strategy. The strategy consists of a list of test programs. For each test program, provide (1) a description of what the test does, (2) what is the input to the test program, (3) what is the output of the test program if the test passes, (4) which functions are tested by the test program, (5) the proposed author of the test program. All functions must be tested adequately.

Sunday, Nov. 15, 11:59 pm. For each team, one team member submits:

(a) Code for the particle simulator

(b) Code for all test programs

(c) Each function in the particle simulator and test program must have a comment that describes the function (see above), which should be updated if the earlier proposed description is no longer valid

(d) Makefile for the simulator and all test programs; make sure your program compiles and runs on joker or gotham

(e) Resultant traj.xyz file of running the simulator with 100 particles, box width 12.8, and 1000 time steps (see below)

(f) For each function (including test programs), identify the author of the code. Also provide a summary of the number of lines of code written by each team member. These numbers must be approximately equal.

(g) A short report with a graph, as described below, is also required.

Description

1. Write a program called `bd` that takes 3 parameters:

```
bd npos L nsteps
```

The parameter `npos` is the number of particles; `L` is a real, possibly nonintegral number that defines the length of one side of the $L \times L \times L$ simulation box; `nsteps` is the number of time steps to be simulated.

Your program should generate random initial positions for the particles uniformly distributed inside the simulation box. Use time steps of $\Delta t = 0.002$. Use periodic boundary conditions when particles move outside the simulation box (if a particle exits through one face of the box, it reenters the box through the opposite face).

At each time step, particles experience two types of effects:

- Repulsive force. If a pair of particles is separated by a distance less than 2, then they each experience a repulsive force f of magnitude $125(2 - s)$, where s is the distance between the particles. The direction of the force is along the line joining the two particles. Be careful in computing f to make sure it is a repulsive force!
- Brownian displacement. At each time step, a particle is displaced by $g = \sqrt{2\Delta t} z$, where z is a 3×1 random vector. Each entry of the random vector has a standard normal distribution and can be calculated as $(\sum_{i=1}^{12} u_i) - 6$, where the u_i are 12 random numbers in $[0, 1]$ and can each be computed in C as `rand()/(double)RAND_MAX` (see the rand function in stdlib.h).

In summary, we can update the position of a particle i as:

$$r_i(t + \Delta t) = r_i(t) + \xi f_i \Delta t + g_i$$

where r_i is a 3×1 vector representing the coordinates of particle i. The vector f_i is the sum of the repulsive forces experienced by particle i. We will use $\xi = 1$.

2. Output. Your program should output a trajectory file called `traj.xyz`, consisting of a number of "frames", with one frame per time step simulated. One "frame" of the trajectory file records the positions of all the particles after a given time step. A frame has the format:

```
100
time_step_number
0 x1 y1 z1
0 x2 y2 z2
...
0 x100 y100 z100
```

The above example is for 100 particles, but your code should work for any number of particles.

If you wish, you can download the VMD visualization code to visualize the trajectory that you generated: `<http://www.ks.uiuc.edu/Research/vmd>`. In the above, `0` represents the name of the particle. We will just use `0` for all particles (it is needed for VMD).

3. For your report, at least plot the execution time for your simulator code for different numbers of particles. What can you conclude from your timings?