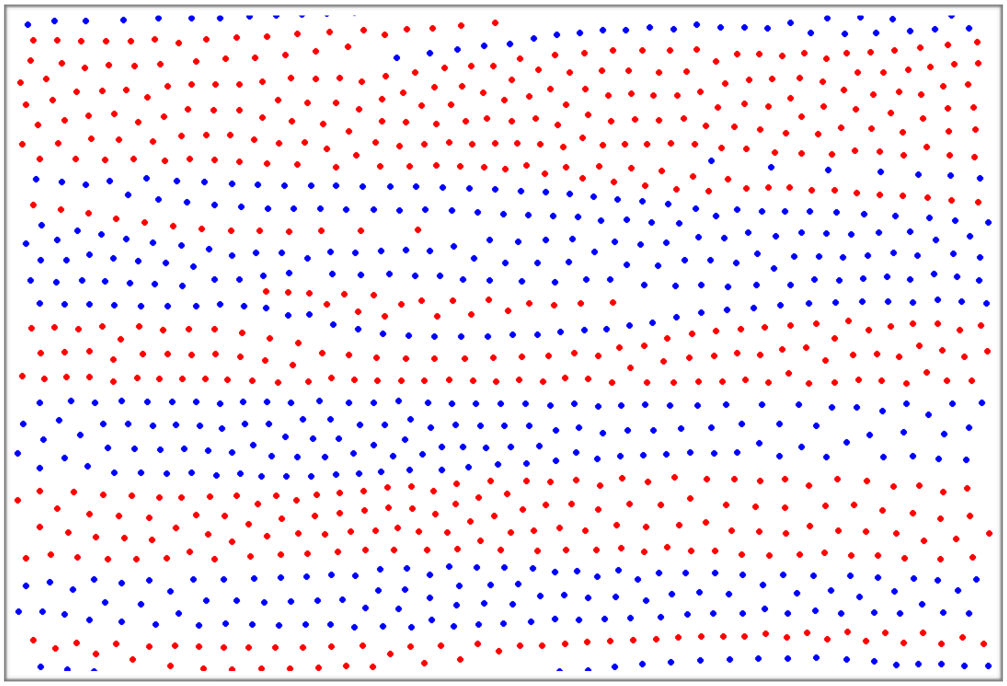
Assignment #1: Molecular Dynamics Simulation, Verlet list methods



Pedestrian crossing simulation, Self-Organizing System

Simulation Methods course

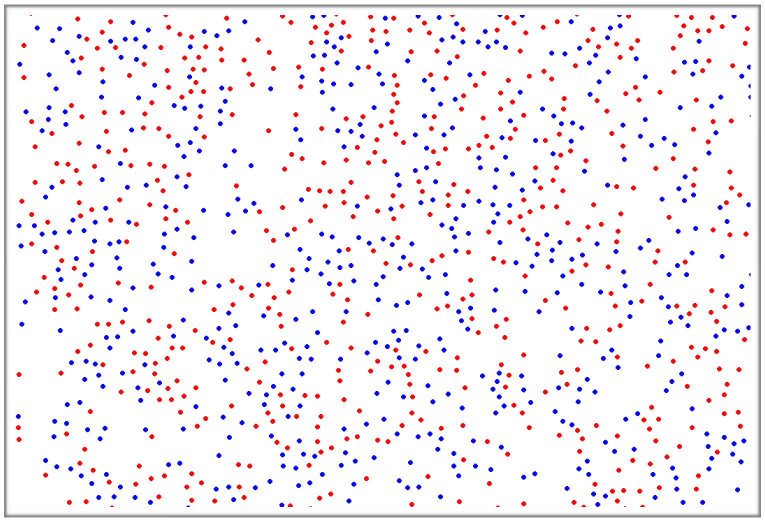
Fall 2019

Writing the Simulation

On the webpage I give you a code for the non-optimized and the Verlet optimized version. You can use these versions to generate the times for the simple + the Verlet optimized code.

If you want to rewrite the simulation, you should write the simulation in C/C++ or something low level (a language that compiles) so that you can control it better (what happens with the cpu and memory usage). High level languages are not used in simulation because of the performance is more important than the ease of coding. Do not use Java or some high level language, or Matlab to write the simulation itself. You can use these tools or whichever tool you like to plot the results of the simulation. I give you a plot tool that uses OpenGL to plot the particles (feel free to improve it).

In this molecular dynamics simulation you will simulate a system made of 2 kinds of particles, red and blue. Initially all particles are placed randomly in the box, but with care so no two particles get too close to each other (no closer than 0.2). Red particles have a force pushing them from the right to left and blue particles have a force pushing them from the left to the right. This is a constant force on them. All particles are repelling other particles with an exponentially screened $1/r^2$ type interaction, and all particles experience a very strong viscous force from the



environment in which they move (they are over-damped).

So the constant force on the particles is
fx = 0.5 or fx = -0.5 depending on color

the particle-particle interaction is $f = 1/r^2 e^{-r/r0}$
where a is the inverse screening length a=0.25
(which means a screening length of r0=4)

and the equation of motion is

x = x + fx * dt
y = y + fy * dt

where dt is small (0.05 or 0.005 ish)

Step by step all forces are calculated (added from all contributions), particles are moved, forces zeroed and this is repeated again and again.

The result is a movie file that you can visualize using the provided softplot program (OpenGl) (should work on mac, linux and windows) or in any other way you would like, Matlab or anything else)

You have the code that does the simulation and the plot program. The code has the optimized and the non-optimized version. You need to show that by increasing the system size (N, number of particles and also the box size, proportionally! if you increase the box to be 2x as big in x and y direction N has to be 4x as much to have the same density).

You need to measure the time it takes to run the program as a function of N, and plot that for both the non-optimized and the optimized code, and show me the run times vs N for both cases in one plot.

What you need to present:

- you ran the non-optimized code and got the time for running the simulation for different system sizes (N=200,300,400, … try going up a bit in size - it is perfectly normal to run a simulation for hours this is not supposed to be done in minutes)

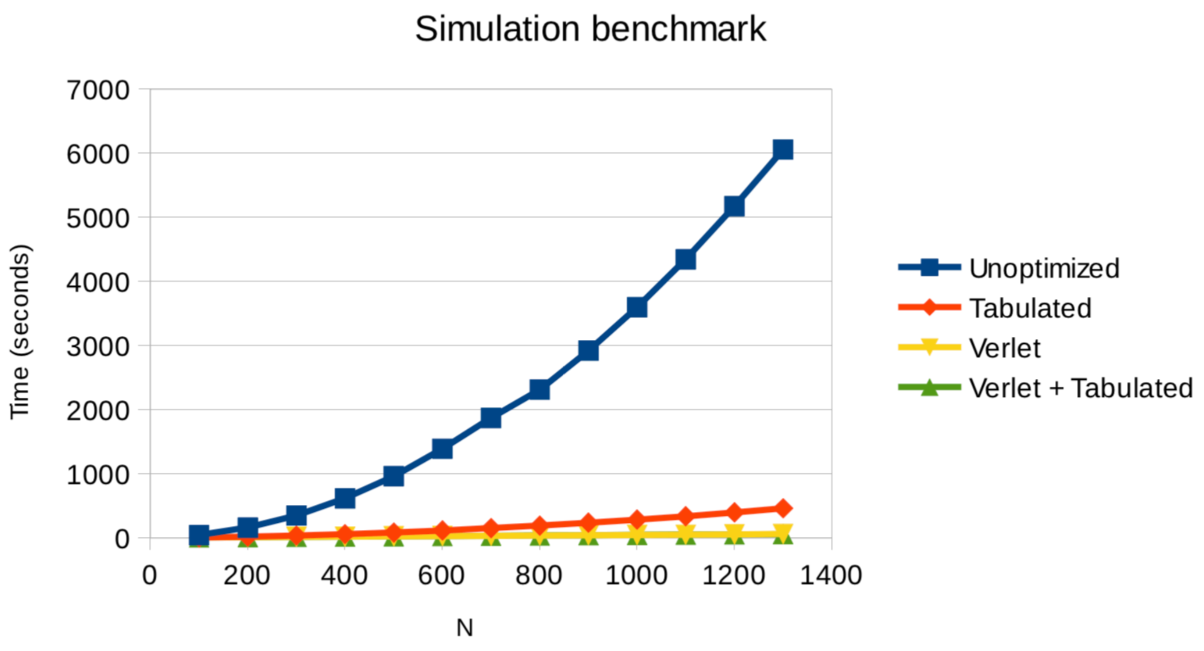
- you ran the Verlet optimized code and got the time for running the simulation for different system sizes (N=200,300,400, … try going up it is perfectly normal to run a simulation for hours this is not supposed to be done in minutes)

- show the plot time vs N_particles with the three curves on it
  - show me some videos of the system (using plot or any method you with to plot)
  - I will ask about the modifications to the code you did for the grid case or other aspects of the code to see if you understood it

  The time to run the simulation should look something like this:
- unpotimized
- optimized with Verlet
- optimized with Verlet and Tabulated forces

You need to write the tabulation part yourself by tabulating f/r as a function of r2 and then recalling the values instead of re-calculating them every time.

Note: you will only have 3 curves, unoptimized, optimized and optimized + tabulated


Simulation benchmark
Time (seconds)
N
Unoptimized
Tabulated
Verlet
Verlet + Tabulated