FYS4460 -
Second compulsory project

Mathilde Nygaard Kamperud

March 22, 2013

1 Parallellization

The most time consuming part of my program is in the function called findForces(). Here there is two four loops (one triple- and one quadruple- for loop), that runs through all the particles and calculates the forces. In parallellizing the code we seem to have to options; we can divide the system into cells (larger than the cells we already have), and calculate the forces in these cells in parallel (with the aid of MPI to communicate), or we can let the program run in serial except when we calculate the forces. I have chosen the latter option, where I use openMP to parallellize the loops. This is a much easier solution, because it just means adding a few lines of code in my original code. The back side to this solution is that it requires that we work on a shared memory computer, while the solution with MPI will run on a cluster. For now I am working on a shared memory computer, so it is fine, but if I want to run for a larger system I will consider using MPI.

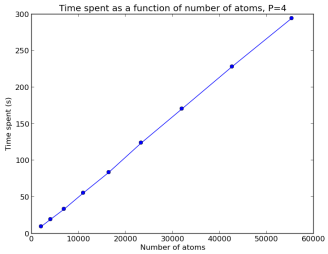
1.1 Efficiency

If it takes an amount of time, $T(N, P)$, to find the new state in a molecular dynamics code, the speed of the simulation may be characterized by

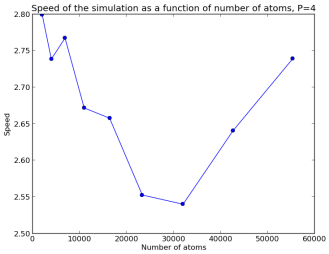
$$S_P = T(N, 1)/T(N, P)$$

where N is the number of atoms and P is the number of parallel processes.
The parallel efficiency of the method is

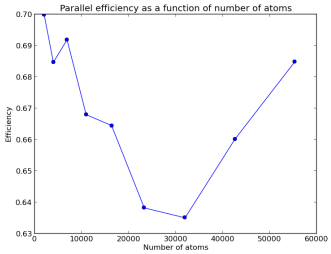
$$E_P = S_P/P. \tag{1}$$



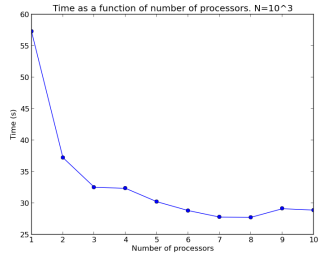
(a) Time spent with four parallel processes.



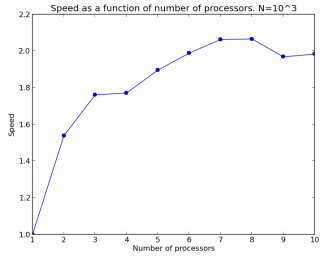
(b) The speed with four parallel processes



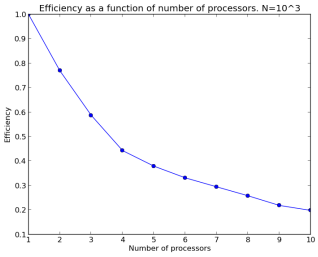
(c) The efficiency with four parallel processes



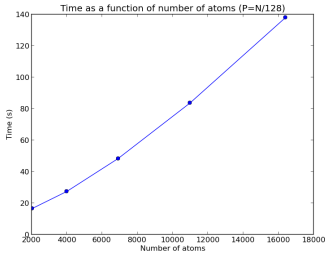
(d) Time spent with $N = 4 \cdot 10^3$, varying P.



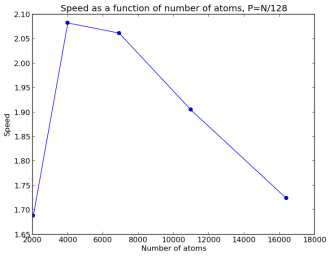
(e) Speed with $N = 4 \cdot 10^3$, vaying P.



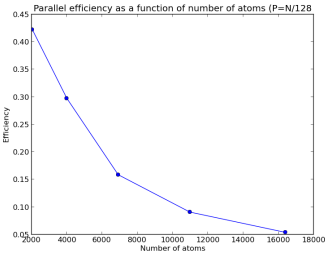
(f) Efficiency with $N = 4 \cdot 10^3$, varying P.



(g) Time spent for P=128N, varying both N and P.



(h) Speed for P=128N, varying both N and P.



(i) Efficiency for P=128N, varying both N and P.

Figure 1: Testing the parallellized program by varying P and N.

2 Generation of a porous matrix

Our plan to generate a porous material is to first generate a thermalized Lennard-Jones system, and then select a given part of the sytem to be the solid matrix. The simplest version is to freeze all the atoms in the matrix, meaning they are not allowed to move, but still let them interact with the particles that are allowed to move. One way to make pores is to generate randomly large spheres placed at random in the Lennard-Jones liquid, and choose to let the particles inside the spheres be frozen, or the other way around.

2.1 Cylindrical pore

To practice our pore-making-skills, the first task was to make a cylindrical pore with radius $R = 2nm$ at the center of a system with $N_x = N_y = N_z = 20$ unit cells of size $b = 5.72$A. First, the system is thermalized at $T = 0.851T_0 = 101.9K$. I made the matrix by letting the atoms know if they are able to move, or not. They all have an attribute called canMove, which is boolean true if the atom in fact is able to move. After thermalizing the system, I marked the particles outside the cyllinder by setting “canMove“ to ”false“. Then, only the particles that are unable to move are written to a VMD-file (my program can also do it the other way around). In figure 2 we have a visualization of the matrix.

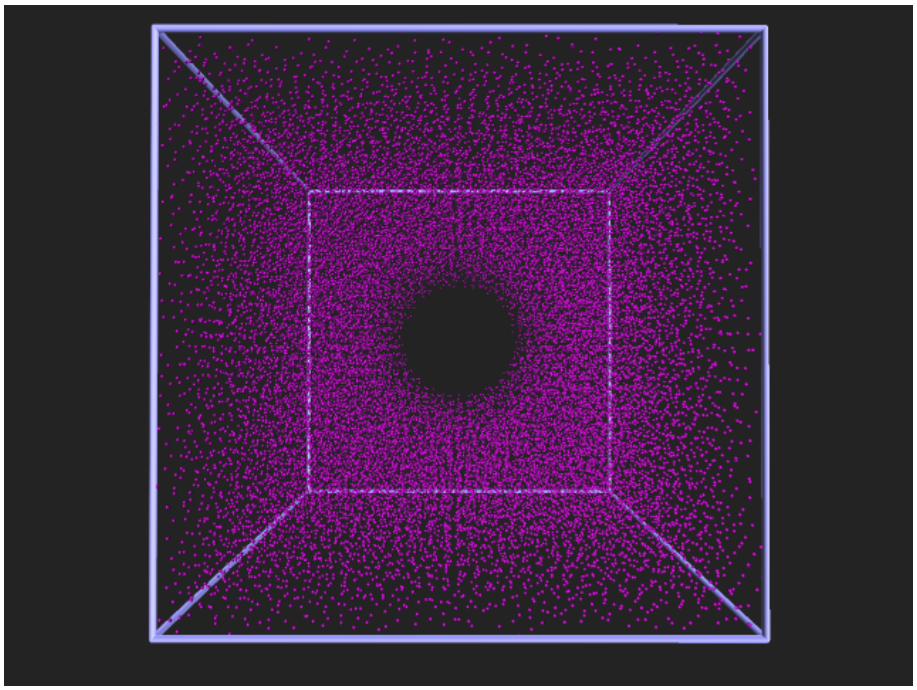


Figure 2: A cylindrical pore of radius $R = 2$ nm. The length of the system in each space direction is $20 \cdot 5.72\text{A} = 11.44\text{nm}$

2.2 First nano-porous system

In figure 3 you see the argon liquid with 20 pores at random positions. The white particles are atoms that are not able to move. I have not yet time evolved the system.

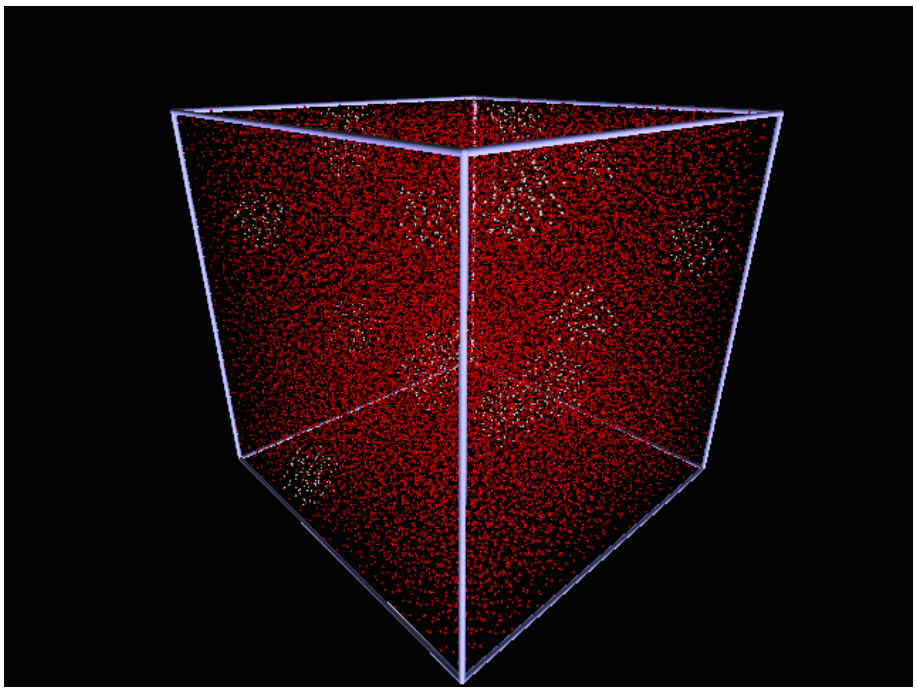


Figure 3: Argon liquid with 20 randomly sized and placed pores.

We can see here that the matrix constitute a small amount of the total volume, which is a bit counter intuitive to me. I decided to increase the number of spheres to 100, and found that the matrix looks more reasonable, see figure 4.

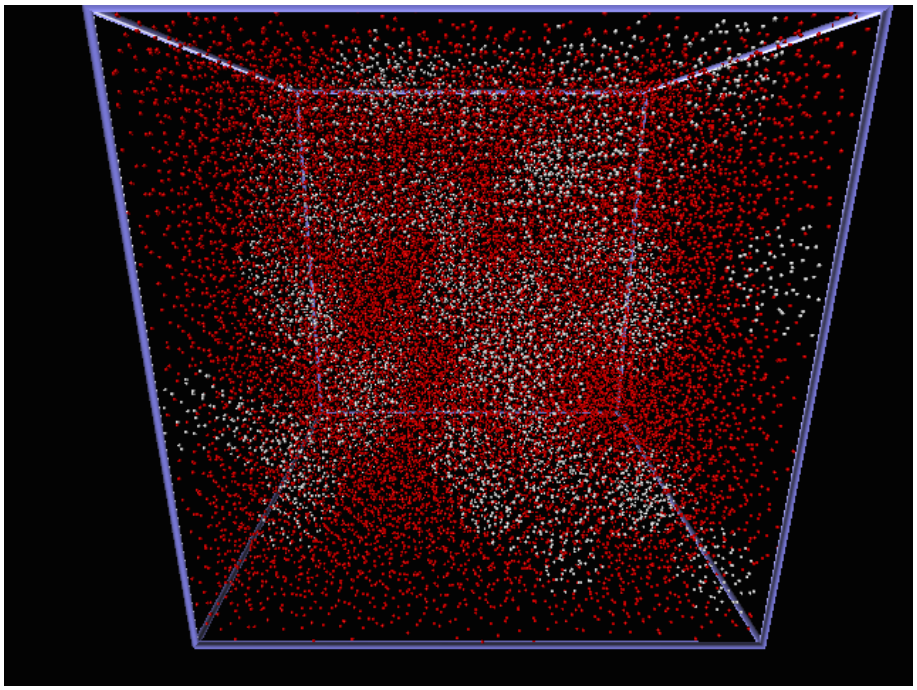


Figure 4: Argon liquid with 20 randomly sized and placed pores.

2.3 Porosity

The porosity, ϕ, is the relative amount of pore space in the volume. To calculate this I first found the average volume of each atom by dividing the total volume by the number of particles, before cutting out the pores. Then I just counted all the particles set as unmovable, and multiplied this number by the average volume to find the volume of the matrix.

$$V_a = V/N_{all}$$
$$V_{matrix} = V_a N_{matrix}$$

We can now set the porosity as

$$\phi = 1 - V_{matrix}/V \tag{2}$$