ETH zürich

High Performance Computing for Science and Engineering I

P. Koumoutsakos, M. Troyer
ETH Zentrum, CLT F 12
CH-8092 Zürich

Fall semester 2014

Set 5 - Metropolis algorithm

Issued: October 24, 2014
Hand in: October 31, 2014, 8:00am

Question 1: Rigid disks

In this exercise we perform the same simulations as done by Metropolis et al. [1] when they first came up with this brilliant algorithm.

The paper studies a system of N rigid disks in a two dimensional square of size $L = 1$ having periodic boundary conditions. Rigid disks are two dimensional particles of radius $d_0/2$ interacting with perfect elastic collisions; their interaction is described by the potential energy

$$V(r = |\vec{x}_1 - \vec{x}_2|) = \begin{cases} \infty, & r < d_0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Goal of the exercise is to compute the average radial distribution of particles $N(r)$ given the particle size d_0 and the particle number N.

To simplify the calculation the distance between particles is split in $M = 512$ bins of equal surface, and we will actually compute the histogram $\langle N_m \rangle$. Given that the maximum distance between particles in the periodic box is $r_{max} = \sqrt{2}L/2$, create bins of equal area $\Delta A^2 = (r_{max}^2 - d_0^2)\pi/M$. For each configuration $\{\vec{x}_i\}$ the histogram $N_m[\{\vec{x}_i\}]$ contains the number of pair of particles $\vec{x}_i^{j_1}$ and $\vec{x}_i^{j_2}$ satisfying

$$m\Delta A^2 + \pi d_0^2 \le \pi r_{j_1 j_2}^2 < (m+1)\Delta A^2 + \pi d_0^2, \tag{2}$$

with $r_{j_1 j_2}^2 = |\vec{x}_i^{j_1} - \vec{x}_i^{j_2}|^2$ being the distance between the two particles. The bin index is easily computed as $m = \lfloor (r_{j_1 j_2}^2 - d_0^2)/\Delta A^2 \rfloor$.

The expectation value of $\langle N_m \rangle$ can be evaluated by Monte Carlo. A straight forward way to estimate the result is simple sampling, where we sample $\tilde{S}$ random configurations $\{\vec{x}_i\}$:

$$\langle N_m \rangle \approx \frac{1}{\tilde{S}} \sum_{i=1}^{\tilde{S}} N_m[\{\vec{x}_i\}] \cdot \omega(\{\vec{x}_i\})\,, \tag{3}$$

with the configuration weight $\omega(\{\vec{x}_i\}) = 0, 1$ depending whether $\{\vec{x}_i\}$ is a valid or not, i.e. there are no overlapping particles. However, this method may draw many samples which have a vanishing weight and will converge very slowly.

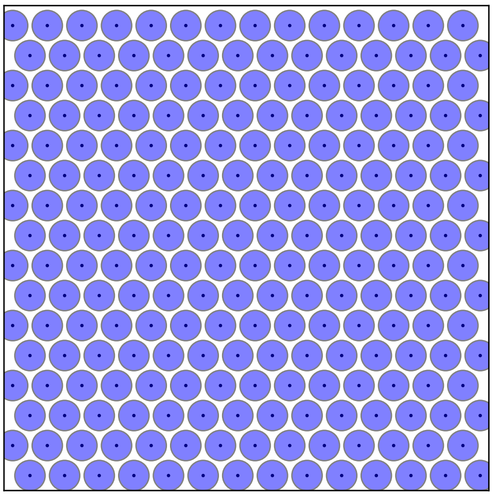

Figure 1: Initial configuration in a box of size $L \times L$.

A much more effective method is importance sampling using the Metropolis algorithm. Here, we restrict the sampling of random configurations to only S valid ones. The estimate simplifies then to

$$\langle N_m \rangle \approx \frac{1}{S} \sum_{i=1}^{S} N_m[\{\vec{x}_i\}] \,. \tag{4}$$

Starting with some initial valid configuration, the Metropolis algorithm for our problem reads:

1. Randomly pick a particle, and propose an update $\vec{x} = (x, y) \to (x + \delta x, y + \delta y)$, where δx, δy are random (uniform) displacements in $[-\alpha, \alpha[$.
2. Accept the new configuration if the new position does not overlap with the other particles.
3. Measure the quantities you want to estimate after a fixed number of updates.

In our application we start the simulation with a triangular lattice as shown in Figure 1 with horizontal spacing $\Delta x = L/N_x$ and vertical spacing $\Delta y = \frac{\sqrt{3}}{2}\Delta x$. The initial lattice will then contain $N = N_x \times N_y$ particles.

Since the initial state is very artificial, before measuring any quantity we have to *equilibrate* the system by running S_{equi} sweeps without measurements. After that we perform S sweeps and we measure at the end of each. Note that by *sweep* one means performing N updates.

a) What is the a-priori proposal probability $A_{a,b}$ for an update from configuration a to configuration b of the proposed Metropolis algorithm? What is the probability to propose the inverse move $A_{b,a}$?

We have to pick 1 random particle over N, and we displace it in two dimensions picking one infinitesimally small interval of size δ over the full range 2α. To restore configuration a from configuration b, we need exactly the same steps:

$$A_{a,b} = A_{b,a} = \frac{1}{N} \left(\frac{\delta}{2\Delta} \right)^2$$

b) Prove that the Metropolis acceptance probability $P_{a,b}$ satisfies the detailed balance condition.

We need to show

$$W_{a,b}\,\omega(\{\vec{x}_i\}_a) = W_{b,a}\,\omega(\{\vec{x}_i\}_b)\,. \tag{5}$$

The transition probability is decomposed into a-priory proposal probability $A_{a,b}$ and acceptance probability $P_{a,b}$ as

$$W_{a,b} = A_{a,b}\,P_{a,b}\,.$$

From the previous question we can simplify eq. 5:

$$\begin{aligned} A_{a,b}\,P_{a,b}\,\omega(\{\vec{x}_i\}_a) &= A_{b,a}\,P_{b,a}\,\omega(\{\vec{x}_i\}_b) \\ P_{a,b}\,\omega(\{\vec{x}_i\}_a) &= P_{b,a}\,\omega(\{\vec{x}_i\}_b) \end{aligned}$$

Proof:

Suppose a and b are both a valid configurations, then $P_{a,b} = \omega(\{\vec{x}_i\}_b) = 1$ and $P_{b,a} = \omega(\{\vec{x}_i\}_a) = 1$, therefore the left-hand side and the right-hand side are both equal to 1.

Another case is when one configuration is not valid, e.g. b is not valid, then $P_{a,b} = \omega(\{\vec{x}_i\}_b) = 0$ and and $P_{b,a} = \omega(\{\vec{x}_i\}_a) = 1$, therefore both sides of the equation are zero.

The last possibility is when both configurations are not valid, then all probabilities are zero and the details balance equation still holds.

c) Write a (serial) code that uses the Metropolis algorithm to calculate the expectation value of the binned radial distribution $\langle N_m \rangle$. Run the simulation with the parameter set 1 listed in Table 1.

Table 1: Example parameters.

	N_x	N_y	d_0	α	S_{equi}	S
Set 1	14	16	$0.875\,L/N_x$	$(L/N_x) - d_0$	64	64
Set 2	100	115	$0.5\,L/N_x$	$(L/N_x) - d_0$	100	500

A solution code is available in `disks_serial.cpp`, Figure 2 shows the measures distance histogram.

Important points are:

- random number distributions: `uniform_int` contains last element, `uniform_real` does not.
- periodic boundary condition for the proposed new position.
- minimal distance between particles.
- measurements only after a full sweep.
- error calculation.

d) Parallelize the S steps with multithreading. You can use OpenMP or C++11 manual threading as you prefer.

A solution code is available in `disks_openmp.cpp`.

Important points are:

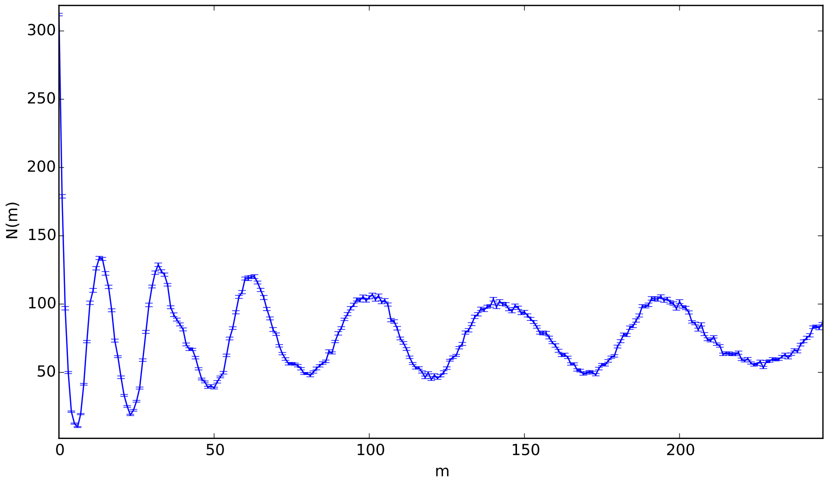

Figure 2: Distance histogram N_m for parameter set 1.

- different seeds in all threads.
- all threads have to perform the equilibration sweeps.
- final *merge* of the accumulated sum and sum of squares.

e) Use the parallel code to simulate bigger systems (see parameter set 2 in Table 1) and compute the strong scaling with the following parameters. What is the maximum speedup that you expect from your code? Do you get close to this value?

In the parallel Metropolis algorithm all threads have to perform the equilibration steps, hence we have a serial fraction

$$r = \frac{S_{\text{equi}}}{S_{\text{equi}} + S} = \frac{100}{100 + 500} = \frac{1}{6} \,. \tag{6}$$

According to Amdahl's law the maximum speedup we can obtain in this simulation is $s \lesssim 1/r = 6$.

In Figure 3 we plot the strong scaling, as expected the speedup does not scale to more than a factor $\times 6$.

References

[1] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller (1953). "Equation of State Calculations by Fast Computing Machines". *Journal of Chemical Physics*, 21 (6): 1087–1092. DOI: `http://dx.doi.org/10.1063/1.1699114`

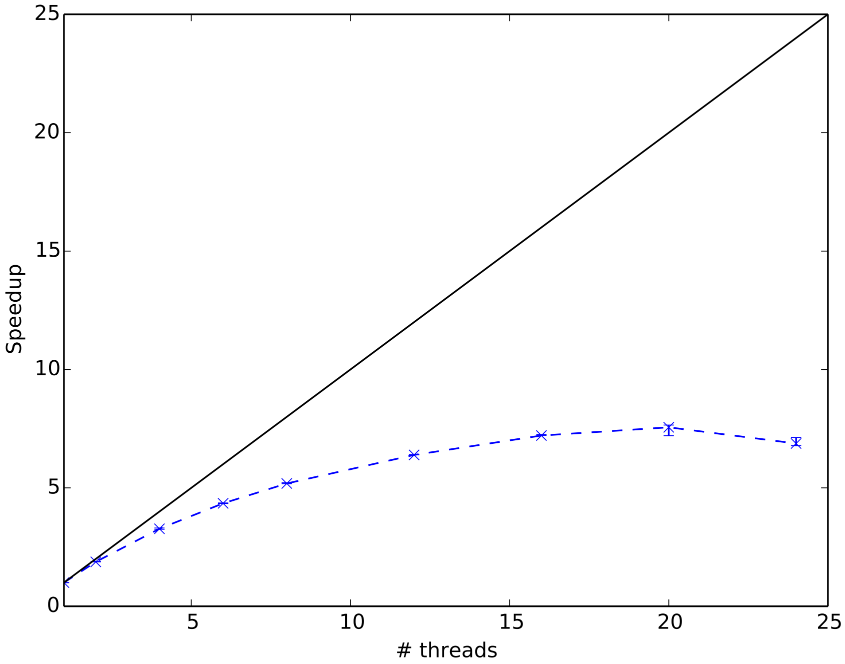

Figure 3: Strong scaling for parameter set 2. Code compiled with GCC 4.8 and `-O3` optimization flags. Runtime measured on a whole Euler's node with the option `OMP_PROC_BIND=1`.

Summary

Summarize your answers, results and plots into a PDF document. Furthermore, elucidate the main structure of the code and report possible code details that are relevant in terms of accuracy or performance. Send the PDF document and source code to your assigned teaching assistant.