Lab Course II, WS 22/23
Prof. Dr. F. Knechtli
Dr. Roman Höllwieser

Homework 9
due Dec 21, 2022

Exercise 9.1 : Lattice setup

In this exercise we start working on some parts of a lattice simulation program, that will be developed during the forthcoming weeks. The goal is the simulation of a three-dimensional scalar ϕ^4 theory for a two-component real scalar field ϕ.
All the routines have to be written keeping in mind that they are going to be reused in the next exercise. Clear coding and modularity are an essential request.
We consider a three-dimensional Euclidean lattice. The dimensions are labelled by `mu`=0,1,2. We will distribute onto a processors grid only the two "space" dimensions 0 and 1, and not the "time" dimension 2. In your implementation adhere to the following guidelines:

- The user inputs the number of processes `NPROC0` and `NPROC1`. If `NPROCmu`=1 then the dimension `mu` is not divided.
- The user inputs the **local** lattice sizes L_0 and L_1 and the **global** size L_2.
  L_0, L_1 and L_2 should be **even**.
- Write a routine to create a two-dimensional grid of processes. Define arrays `int cpr[2]` and `int npr[4]`. `cpr[mu]` containes the `mu` coordinate of the process in the cartesian grid (`mu`=0,1). `npr[0]` is the rank of the neighboring process in direction -0, `npr[1]` the rank of the neighbor in +0, `npr[2]` the rank of the neighbor in -1 and `npr[3]` the rank of the neighbor in +1.
- The local lattice is enlarged by the exterior boundary points which are copies of the corresponding points of the local lattices on the neighboring processes. The number of exterior boundary points in direction 0 is `FACE0`, in direction 1 is `FACE1` and the total number of boundary points is `BNDRY`=2×(`FACE0`+`FACE1`). We set `FACEmu`=0 if `NPROCmu`=1.
- Write a routine which determine the following arrays:
  - `int ipt[VOLUME]`:
    `ipt`[nl] is the index of the local point $nl = x_0 + x_1L_0 + x_2L_0L_1$, where (x_0, x_1, x_2) are the local coordinates of the point. Defining `VOLUME`=$L_0L_1L_2$, the indexing of the points has to be such that `ipt`=0, 1, $\cdots$ `VOLUME`/2-1 label the even points and `ipt`=`VOLUME`/2, $\cdots$ `VOLUME`-1 the odd points. A point is even if $x_0 + x_1 + x_2 \bmod 2 = 0$ and odd otherwise.

The ordering of the index for the boundary points is as follows:

  - `VOLUME` $\leq$ `ix` $<$ `VOLUME+FACE0/2` : even points on the face in direction -0
  - `VOLUME+FACE0/2` $\leq$ `ix` $<$ `VOLUME+FACE0` : even points on the face in direction +0
  - `VOLUME+FACE0` $\leq$ `ix` $<$ `VOLUME+FACE0+FACE1/2` : even points on the face in direction -1
  - `VOLUME+FACE0+FACE1/2` $\leq$ `ix` $<$ `VOLUME+BNDRY/2` : even points on the face in direction +1
  - `VOLUME+BNDRY/2` $\leq$ `ix` $<$ `VOLUME+BNDRY/2+FACE0/2` : odd points on the face in direction -0
  - etc.

  These indices will be stored in the following two arrays `iup` and `idn` and mapped to corresponding local indices on neighboring processes with another array `map`:

- `int iup[VOLUME][2]`
  `iup[ix][mu]` is the index of the nearest neighbor point in positive ("up") direction `mu`=0,1 of the point on the local lattice with index `ix`. If the nearest neighbor is on the exterior boundary, its index will have values `VOLUME` $\leq$ `iup[ix][mu]` $<$ `VOLUME+BNDRY` .
- `int idn[VOLUME][2]`
  `idn[ix][mu]` is the index of the nearest neighbor point in negative ("down") direction `mu` of the point on the local lattice with index `ix`. If the nearest neighbor is a boundary point, the same ordering of the index applies as it was discussed for `iup`.
- `int map[BNDRY]`
  if `ix` is a point on the local lattice and if `iy=iup[ix][mu]` is the index of a point on the exterior boundary of the lattice, then `map[iy-VOLUME]` is the index of the point on the local lattice of the neighboring process in direction +`mu` which corresponds to `iy`. The same applies if `iy=idn[ix][mu]` is a boundary point.

(20 points)

Exercise 9.2 : Random walk

To verify the correctness of the indeces we can use a property of the ”random walk”. For this we define the global coordinates

$$\mathbf{x} = (\mathbf{x_0}, \mathbf{x_1}, \mathbf{x_2}) = (\texttt{cpr[0]} \cdot L_0 + x_0, \texttt{cpr[1]} \cdot L_1 + x_1, x_2). \tag{1}$$

You should implement a program in which, for $1 \leq \texttt{i} \leq N_{meas} = 10^4$:

- a starting point $\mathbf{x}^{(0)}$ is chosen at random
- for $1 \leq t \leq T = 100$:
  - move one step in a random direction: $\mathbf{x}^{(t)} = \mathbf{x}^{(t-1)} \pm \hat{\mu}$
  - measure the distance

$$d_{\texttt{i}}(t) = \sqrt{\sum_{\mu} \min\left[(\mathbf{x}_\mu^{(0)} - \mathbf{x}_\mu^{(t)})^2,\ (\texttt{NPROC}\mu \cdot L_\mu - (\mathbf{x}_\mu^{(0)} - \mathbf{x}_\mu^{(t)}))^2 \right]} \tag{2}$$

- the measured value of $d_{\texttt{i}}(t), t = 1, 2, ..., T$ is written to a file.

In matlab, implement a script which reads in these $N_{meas} \times T$ measurements, and plots the function t and the average

$$\langle d^2(t) \rangle = \frac{1}{N_{meas}} \sum_{\texttt{i}=1}^{N_{meas}} d_{\texttt{i}}^2(t) \tag{3}$$

against t. If everything is working correctly, the two curves should be very similar.

Perform this test on a lattice of **global** volume $20 \times 20 \times 20$ and on a lattice of global volume $40 \times 40 \times 40$.

Remark: At any time t, only one process of your grid can be actively moving the point $\mathbf{x}^{(t)}$ within its local lattice, and measuring the distance $d_{\texttt{i}}(t)$. Make sure that the information about the location of the point (and about which process should ”work”) is available to all processes.

(10 points)

Remark: From what you hand in, how to verify the correctness of your program should be clear and simple.