**CSC 4760/5760/6803, Fall 2022**

**Programming Assignment 5**

**Assign date: April 4, 2022, Due: April 15, 2022**

For this assignment you will write parallel program for simulating heat diffusion on a metal plate using MPI. Following is the description of the assignment.

We have a two-dimensional plate. At various points on the plate, heat is applied and we must simulate how this affects the temperature of the plate. We will use a 2 dimensional grid of cells to represent the temperature of the plate as equally spaced point estimates at a specific time (represented by a 2d array of double). The simulation progresses in discrete time steps starting with time zero. During the simulation we update each cell temperature to the next time step by looking at its neighboring cells using the following equation:

|  |  |  |
| --- | --- | --- |
| Ttl | Tt | Ttr |
| Tl | Told | Tr |
| Tbl | Tb | Tbr |

All T’s except for Tnew are the values at the previous time step.

The k determines the rate of heat diffusion and value of k varies between 0 and 1.

If Told is on the edge of the plate then the out of bounds values should be replaced with the boundary temperature.

Some cells will have a fixed temperature as they represent where heat is being applied to the plate. Thus, for each time step:

1. Set all fixed cells to their fixed value.
2. Calculate the new temperature for each cell from the previous cell values.
3. Swap the old and new grids in preparation for the next time step. **DO NOT** copy element by element

When initializing the grid before simulation starts, set the edge cells to the boundary temperature.

Your program should take seven (7) command line arguments. Following are the arguments and their order:

./2d\_heat <rows> <cols> <k> <boundary\_temp> <time\_steps> <fixed\_temps\_file> [outfile]

2d\_heat – name of the executable

rows – the number of rows in the grid.

cols – the number of columns in the grid.

k – a double, the heat transfer rate.

boundary\_temp – the temperature to use when calculating the edges of the grid, as well as the initialization temperature

time\_steps – the number of iterations to run the simulation

fixed\_temps\_file – name of a file containing the cells with fixed temperatures

outfile – optional, if supplied the file to write the temperatures of the final iteration to

The format of the fixed\_temps\_file is as follows:

2 number of fixed cells

4 10 85.3 cell\_row cell\_col cell\_temp

15 2 100.0 cell\_row cell\_col cell\_temp

The format of the outfile is the same as in assignment 3:

1.2, 1.5, 1.4 1st row temperatures

1.1, 1.2, 1.3 2nd row temperatures

A set of test files will be provided.

**Report:**

Measure the compute time with 1, 2, 4, 8, 16, and 28 processes twice. In case of 16 and 28 processes run the code on nodes in such a way that half the processes run on one node and the rest in the other. Set your plate size to 3000 by 3000. Compute the speed up and efficiency of your code for the different problem sizes and number of processes. Write a report presenting and explaining your (provide graph) results.

**Note: Do not use MPI collectives.**

**Submission:**

Submit a copy of your code, a readme file (how to compile and run your code), the report.

**Suggestions:**

1. **Pay attention to the array distribution and getting the result back (stitching the array back together).**
2. **Pay attention to the halo exchange between neighboring processes (ask the TA or instructor, we will explain it in the class)**
3. **Run interactively on the cluster on 3/4 processes with small grid to make sure that that halo exchange is working properly.**

**Grading Rubric:**

Program Working Correctly 80 Points

Correct I/O 10 points.

Correct distribution of the array 10 points

(including ghost row)

Correct stitching of the array back 10 points

Correct cell update (computation) 10 points

Correct halo exchange 40 points

Overall program structure/correctness 10 points

Report 20 Points

Total 100 Points