Università della Svizzera italiana
Institute of Computing CI

HPC Software Atelier, Fall Semester 2022
Team: Prof. O. Schenk, Dr. J. Kardoš,
T. Holt, M. Lechekhab, Z. Panthakkalakath

Project 5
Parallel Nonlinear PDE using MPI
Due date: 30 November 2022, 23:59 (midnight)

We introduced OpenMP to the reaction-diffusion PDE algorithm in Project 3 (”Parallel Nonlinear PDE with OpenMP”) so that we could utilise all cores on a single node. The purpose of Project 5 is now to incorporate MPI into the OpenMP version, allowing us to employ multiple nodes in a distributed-memory version as well. The source code is nearly identical to the OpenMP version you’ve already implemented. You should make the same changes as before. There are some comments below that will assist you.

Before you start

Please download the source codes iCorsi or from git

```
[user@icslogin01]$ cd hpc2022
[user@icslogin01]$ git pull
```

Allocate as many nodes as you need. Let’s say 4.

```
$ salloc -N 4 --ntasks-per-node=1
```

For debugging you can use 1 node and execute multiple processes there. But when your code works well and you are running benchmarks, use 1 process per node. Please load openmpi and python modules using the following commands:

```
$ module load openmpi python/2.7.12
```

Domain decomposition

There is only one process that has all of the data in the serial and OpenMP versions. There are numerous processes in the MPI version (ranks). The grid on which we compute the stencil is divided into equal pieces, each of which is assigned to a different process. Each process has its own section of the grid and cannot access data belonging to other processes. But in order to compute the new value of some point, we need to know values of all of its neighbors. If the point is on the boundary, we need to get value from the process that has this data. So before every iteration, processes first exchange ”halo cells” and save values from neighbors to boundary buffers (bndN, bndS, bndE, bndW). After the exchange every process has all the data it needs to compute the next iteration.

Hints

Note that at the beginning the code does not run. It is because you first have to initialize MPI and there are also some errors because of missing parts you should fill in. Secondly, when you work on ghost cells exchange (`operators.cpp`), look at the resulting image. You will see that only part of the image is correct, or parts of the image are flipped in N-S or E-W direction. Think why this happens. It will help you to find what is wrong in your code.
For testing you can use the same parameters as for OpenMP version, e.g. `./main 128 128 100 0.01` Don’t forget that you cannot start MPI application the same way as serial or OpenMP version. You have to use `srun` or `mpirun`.

```
$ export OMP_NUM_THREADS=10
$ mpirun -np 4 ./main 128 128 100 0.01
```

If you want to run all processes on one node. It is useful for debugging, because you don't block many nodes. But use it **only for debugging**.

```
$ salloc -N 1 --ntasks-per-node=4
```

For measurements to your report use **one process per node**.

```
$ salloc -N 4 --ntasks-per-node=1
```

1. Task 1 - Initialize and finalize MPI [5 Points]

In file `main.cpp`

- Initialize MPI
- Get current rank and number of ranks
- Finalize MPI

2. Task 2 - Create a Cartesian topology [10 Points]

Mini-app uses a 2D grid. Each rank will work on a part of the grid. Make a 2D domain decomposition of the grid depending on the number of ranks.
In file `data.cpp`

- Create the dimension of the decomposition depending on the number of ranks
- Create a non-periodic Cartesian topology for the grid of domains
- Identify coordinates of the current rank in the domain grid
- Identify neighbours of the current rank: east, west, north and south directions

3. Task 3 - Change linear algebra functions [5 Points]

Make the dot product and the computation of the norm over all ranks.
Why only these two functions and not the others?
In file `linalg.cpp`

- Add a collective operation to compute the dot product
- Add a collective operation to compute the norm

4. Task 4 - Exchange ghost cells [45 Points]

Each rank has only it's own part of the grid. Before every iteration it is necessary to exchange ghost cells between the neighbors, so every process has all the data it needs for the stencil computation.
Use point to point communication to exchange ghost cells among neighbours.
In file `operators.cpp`

- Add point-to-point communication for all neighbours in all directions
- Use Non-blocking communication
- Try to overlap computation and communication

Be careful to send first / last row / column. Before you send the data, copy it to the send buffers (buffN, buffS, buffE, buffW) and receive to ghost cells (bndN, nbdS, bndE, bndW). Because you copy the data to the 1D array fist, you don't need any custom data types for send and receive. Look at Figure 1 to see an example.

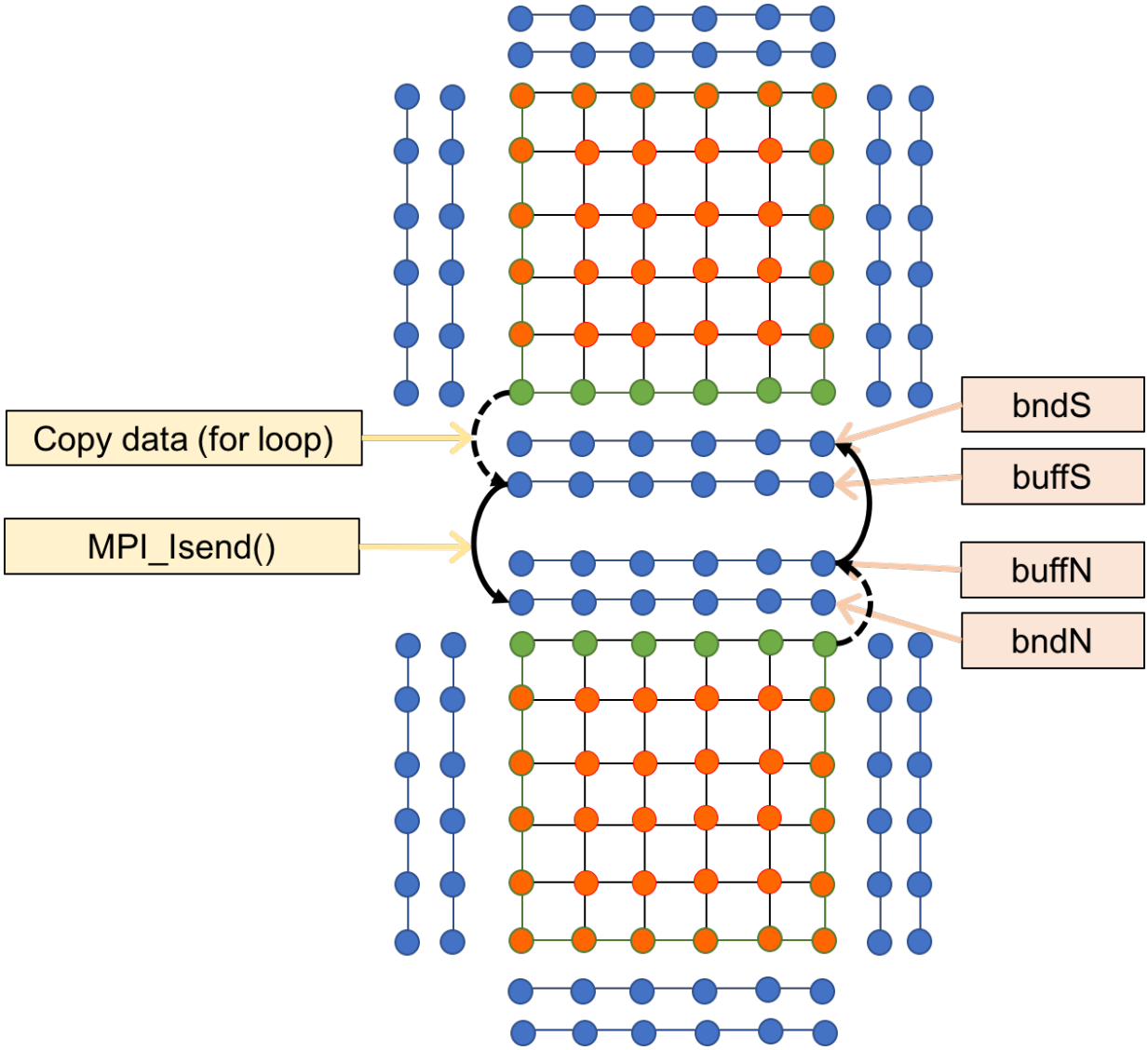

Figure 1: Ghost cell exchange. Copy the North (South) row to buffN (buffS), send buffN (buffS) to the neighbor, receive to bndS (bndN).

5. Task 5 - Testing [20 Points]

How does it scale at different resolutions? Plot time to solution for these grid sizes for 1-10 threads and 1, 2, 4 MPI ranks.

- 128x128
- 256x256
- 512x512
- 1024x1024
- You can also use larger grid sizes.

6. Task 6 - Quality of the Report [15 Points]

Each project will have 100 points (out of which 15 points will be given to the general quality of the written report).

Additional notes and submission details

We only accept submissions using our Latex template and C/C++ code. Please submit the source code files (together with your used `Makefile`) in an archive file (tar, zip, etc.), and summarize your results and observations for all exercises by writing an extended Latex report. Use the Latex template provided on the webpage and upload the Latex summary as a PDF to iCorsi .

- Your submission should be a gzipped tar archive, formatted like project_number_lastname_firstname.zip or project_number_lastname_firstname.tgz. It should contain:
  - all the source codes of your MPI solutions;
  - your write-up with your name project_number_lastname_firstname.pdf.
- Submit your .zip/.tgz through iCorsi .