Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

High-Performance Computing Lab for CSE **2024**

Student: Benedict Armstrong Discussed with: Tristan Gabl

Solution for Project 4

Due date: Monday 29 April 2024, 23:59 (midnight).

---

1. Ring sum using MPI [10 Points]

For this task we are asked to implement a ring sum using MPI. We calculate the sum of all MPI ranks in a communicator. To do this each process starts by sending its rank to the next process in the ring adds it to an internal sum and then sends the message on to the next process. This is done until each message has made a full loop of the ring. We notice that by the end the sum should be equal to the sum of all ranks in the communicator.

$$\sum_{i=0}^{n-1} i = \frac{n(n-1)}{2}$$

To avoid avoid deadlocks we can make every process send before they receive.

2. Cartesian domain decomposition and ghost cells exchange [20 Points]

The implementation of this task was relatively straightforward after doing the MPI tutorial. The main difficulty was not making any errors with the indexing into the data array. The convenient `MPI_Cart_shift` function makes it easy to get the rank of the cartesian up/down/left/right neighbors. To avoid deadlocks I used the asynchronous `MPI_Isend` and `MPI_Irecv` functions and then waited for all the communication to finish using `MPI_Waitall` at the end. For the diagonal neighbors I used the `MPI_Cart_coords` function to get the coordinates of the current rank and then added/subtracted 1 from each coordinate to get the coordinates of the diagonal neighbors. This was then converted back into a rank using `MPI_Cart_rank`. Running the program using 1 produces the output in 2. For debugging purposes I also printed the neighbors of each rank. Below that we can see that the output data from rank 9 indeed matches the expected result outlined in the assignment.

```
mpirun -np 16 ./build/ghost
```

Listing 1: Running the ghost cell exchange program

```
Neighbors of rank 9
+-----------+
|  4|  5|  6|
|-----------|
|  8|  9| 10|
|-----------|
| 12| 13| 14|
+-----------+
data of rank 9 after communication
4  5  5  5  5  5  5  6
8  9  9  9  9  9  9 10
8  9  9  9  9  9  9 10
8  9  9  9  9  9  9 10
8  9  9  9  9  9  9 10
8  9  9  9  9  9  9 10
8  9  9  9  9  9  9 10
12 13 13 13 13 13 13 14
```

Listing 2: Output of the ghost cell exchange program

3. Parallelizing the Mandelbrot set using MPI [30 Points]

To start I implemented the partitioning as outlined in the assignment handout. The MPI documentation at was very helpful for this. After each process has calculated its part of the Mandelbrot set we need to gather the data back to the root process. This is implemented by simply receiving the data from each process in the root process (rank 0) and writing the data to a png file. There is no danger of a deadlock here as there are no dependencies between the processes. Only the root process writes the data to a file. The output image is shown in figure 1.

To evaluate the performance I ran the program with $p = 1, 2, 4, 8, 16$ processes and an image size of 4096×4096. Figure 2 shows the time taken by each processes (rank) for different number of processes.

4. Parallel matrix-vector multiplication and the power method [40 Points]

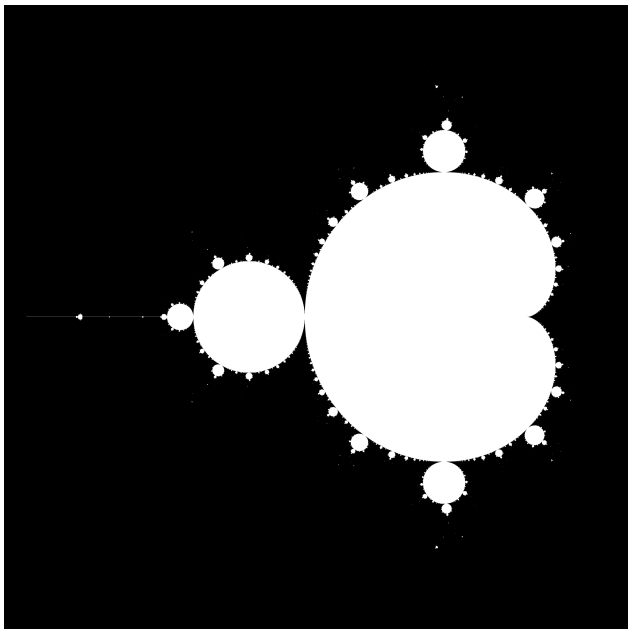

Figure 1: Mandelbrot set

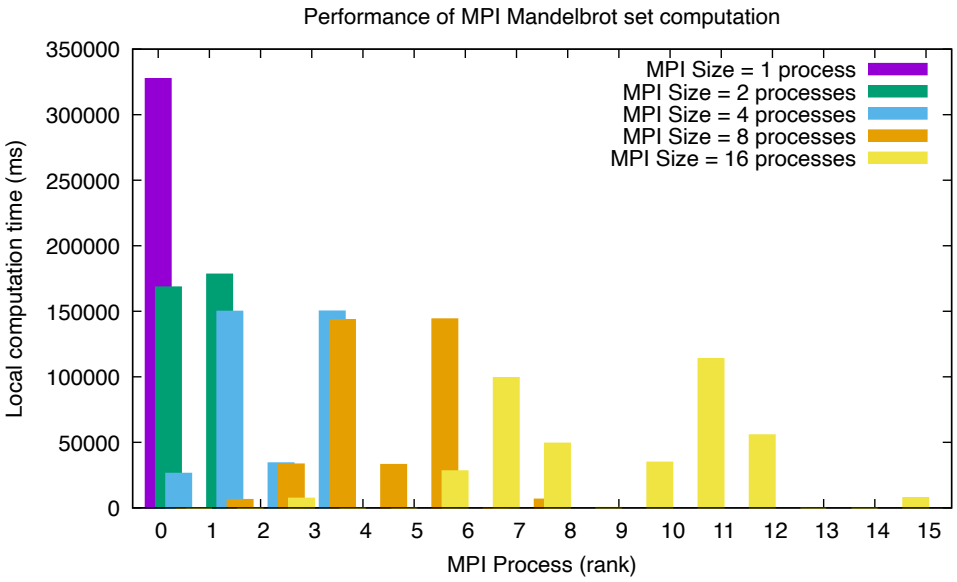

Figure 2: Performance of the Mandelbrot set program