Fall 2020

CSCI 6454: HW3 (12 points)

On Distributed-memory parallel computing

Problem 1 (5 points): Basic MPI constructs

Write a parallel MPI-based distributed program that does the following:

1. **Hello! world:** each MPI process prints a text "hello from process *i*", where *i* is the process rank.
2. **Basic send/receive:** each process sends its rank information to a specific other process. You may implement your own logic. For example, process *i* might send to process *(i+1)* and the last ranked process sending to process 0. Each process prints "Received a message from process *x*", where *x* is the sending process.
3. **Broadcasting:** process 0 broadcasts a (any) number to all other processes. All processes print the received number.
4. **Scatter and Gather:** show an example of scatter and gather operations. Print what each process had before and after scatter and gather collectives.
5. **Reduce:** use MPI_Reduce to compute the sum and average of an array of numbers. Consider that the numbers are distributed evenly among processes. Make process 0 print the results.

Problem 2 (7 points): Matrix-Matrix Multiplication

Write a parallel MPI-based program to perform matrix-matrix multiplication. You may generate the matrices using random number generators. You may also use any publicly available matrix datasets from the internet. Cite your sources if you use internet datasets. Try to use matrices with dimension greater than or equal to 100(row)*100 (column). Use any approaches of your choice to distribute tasks among MPI processes. If **MatA** and **MatB** are your input matrices, then store the output in matrix ***MatC***. Print C in a file.

Additional requirements:

1. Discuss in your report how you distributed data and tasks.
2. For this problem, provide performance evaluation: serial version vs multi-process version. Show plots that indicate speedup with increasing number of processes. Discuss any insights/findings.
3. Provide at least one set of sample input that you used and the corresponding output.

Instructions:

1. For each implementation, follow good programming practices. Have plenty of comments in your source file. Even though I did not quite specify the input and output formats for some problems, you should have descriptive and

intelligible prompts and outputs. You MUST have a README file for each problem specifying your input/output and how to run and any assumptions you made.

2. Your programs MUST compile and run. Otherwise, you will lose significant points.
3. **Report:** Provide an MS Word document (or similar) with the results, plots, and other discussions.
4. To develop you code and performance experimentation, you may want to use LONI or your own computer. Most modern computers have 4 cores or so. Thus, you should be able to get at least 4 data points for performance plots. Although, the logical number of processes can be higher than the number of physical cores.
5. If you want to use LONI, create an account with LONI. Mention my name as your collaborator. I can then have your request approved and share with you my own allocation. Let me know your intention.
6. Put the solution for each problem in a separate directory and zip them together and submit through Moodle. Email me your work ONLY IF Moodle is not working.
7. Name each directory as HW3_YourFirstName_YourLastName_X, where X is the problem number. The final zipped archive name should be HW3_YourFirstName_YourLastName.
8. All work must be your own. This is not a team assignment.
9. DEADLINE: 29 October 2020 Thursday 11:59pm.