CITS3402/5507: Project 2

2024 Semester 2

Notes:

- The project is worth 20% of your final mark.
- The due date is 11:59 pm on October 18.
- The project can be done either individually, or in a group of two.
- The submission is through `cssubmit` and will consist of code file(s) and a report file (only in PDF format). Both files must have the names and student numbers of group members.

Description

The second project is a continuation of the first project. The problem is the same: *matrix-matrix multiplication of sparse matrices.*

In the first project, the task was to perform the multiplication using threads, using a shared memory model. In this second project, you will use both distributed memory and shared memory computation.

We have access to four nodes on Setonix, with up to 2×64 physical cores in each node. This will be the hardware platform for the second project. For an example of how you can run MPI code across multiple nodes on Setonix, please take a look at **Laboratory Sheet 4 - Introduction to MPI on Setonix**.

Your task is to determine the maximum size of two random *row-compressed*[1] square matrices ($N \times N$) that can be multiplied within 10 minutes using various configurations. This means that you should always be using `#SBATCH --time=00:10:00` in your SLURM script.

The goal of this project is to produce a report on how parallelism increased the size of matrices you could multiply within the time limit. While your code will facilitate this investigation, the written report and analysis are the most critical parts of the assignment.

Your investigation should focus on densities (i.e. non-zero probability) of 0.01, 0.02, and 0.05. You should (at least) investigate the effect of different numbers of threads, processes, and nodes.

Your submission should be able to output all *sparse matrices* (XB, XC, YB, YC, XYB, XYC) to text files so the correctness of your algorithms can be easily verified (by both you and the marker). Whether your program outputs the matrix should be a configurable option. For your final experiments, we are only interested in the time taken to generate and multiply the random matrices. This means that you should **not** write the matrices to files during your experimentation.

Tasks

- Implement *row compression* with support for both distributed memory (MPI) and shared memory (OpenMP) computation.
- Determine the largest random matrices you can multiply within a 10-minute job allocation using four Setonix nodes. Investigate how this compares to the sequential performance.
  - You should (at least) investigate sequential, OpenMP, MPI, and OpenMP+MPI performance.
- Write a report that describes your findings. Your report should (at least) include:
  - An explanation of how to execute your program, with examples.

[1] For an explanation of *row compression*, please see the Project 1 description.

  - A clear explanation of how you have parallelised your algorithm. This explanation should reference both distributed memory and shared memory computation.
  - A detailed report of your experimentation using 10 minutes of run-time on up to four Setonix nodes.

Allocation of marks

- **5 marks** Correct parallelisation of the sequential algorithm using OpenMP+MPI
- **5 marks** Quality of the report detailing your algorithm and methodology
- **10 marks** Quality of experimentation for evaluating the performance and speed-up of your algorithm