CS 4175 Fall 2023
Shirley Moore, Instructor
Programming Assignment 4
50 points

Parallel Matrix Operations using MPI

For this assignment, we will implement parallel matrix-vector multiplication for distributed memory using MPI and using a 1-dimensional data decomposition of the matrix and vector. We will also construct an analytical model of the parallel runtime complexity for the parallel algorithm that includes both computational and communication costs.

You are provided with the following files:

- mvm-driver.c, a driver program for mvm
- mvm-skeleton.c, a skeleton program for the parallel matrix-vector multiplication that is missing the communication operation
- Makefile, a makefile for the mvm program

1. (10 pts) Copy mvm-skeleton.c to mvm.c and fill in the missing communication operation. Test your program with small input sizes (e.g., 8, 32, 1024) using different numbers of processes (e.g., 1, 2, 4, 8) on a GPU-shared partition node on Bridges-2 to verify that it works correctly. Then test with the input size 32000 that we will use for parallel timings. Show a few test results.

2. (10 pts) Give the theoretical analysis of the time complexity of sequential matrix-vector multiplication. Using just one task, time the mvm program for input sizes 1000, 2000, 4000, 8000, 16000, and 32000. Graph execution time (vertical axis) versus input size (horizontal axis). Discuss how your results compare to your analytical model. Try to explain any deviations from your model.

3. (10 pts) Develop an analytical model for parallel runtime complexity of distributed matrix-vector multiplication that includes both computation and communication time. Your model should be in terms of message startup latency t_s, word transfer time t_w, number of processes p, and input size n. Derive and explain each part of your model.

4. (10 pts) Write a batch job script that will run your parallel matrix-vector multiplication on 1, 2, 4, and 8 processes using input size 32000. Note that to run 8 processes and not oversubscribe the cores, you will need to request 2 GPUs. Turn in both your job script and the output file produced by the runs.

5. (10 pts) Graph the speedup for your runs from step 4 and compare with linear speedup on the same graph. Discuss your results with respect to your analytical model from 3.

6. (25 pts extra credit) Write a parallel matrix transpose program using MPI. Use a 1-D decomposition of the matrix. Each process should initialize its part of the matrix. Use the most efficient collective communication operation available to perform the transpose operation. Remember to do a local transpose in addition to the communication. Develop an analytical model of the parallel execution time that includes both local data movement and network communication times. Using a sufficient matrix size to get significant runtimes, time the distributed transpose for 1, 2, 4, and 8 processes. Discuss the timing results with respect to your analytical model.