CMPSC 140 Summer 2022: Homework 3, Exploration of MPI

Instructor: Richard Boone

Due: July 20th, 2022 9 PM

Purpose: This assignment is meant to familiarize you with a variety of MPI programs and MPI programming styles. You will explore multiple degrees of parallelization and discover various advantages available to you.

Program 1: Estimation of Pi through random sampling In this assignment we will ask you to estimate the value of pi by random sampling. For a visual example of this, check the following picture. For each random sample, generate a random point in the area between (0,0) and (1,1). Check if the value is within a unit circle's distance from the origin. After a number of samples, you should be able to generate an estimate of pi based on a comparison between the number of samples in the unit circle and the total number of samples.

Steps:

1. Write an MPI program for estimation of Pi using the above method using only simple MPI commands (Send, Receive, etc.) for communication of the final evaluation from each process directly to process 0. This and all future versions of the program should print out the estimated pi value, accuracy ($abs(\frac{\pi - \pi_{estimated}}{\pi})$), and total time taken.

2. On a single process, increase the number of samples you perform until it takes over 100 seconds to finish the sampling.

3. With the maximum number of samples from the previous step, increase your number of processes one at a time until you reach your maximum available number of processes. Record the time taken as you increase number of processes.

4. Rewrite your program to include a tree-based communication structure as discussed in lecture 1. Repeat step 3, recording times again.

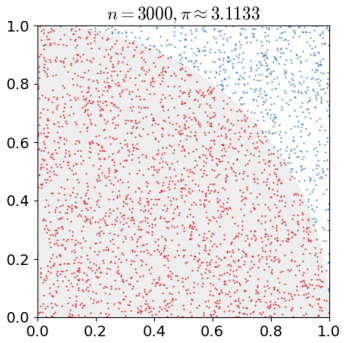
5. Repeat the previous step, instead using MPI_Reduce for communication.

6. For a large problem size (time ¿ 30 seconds), compare the three versions of your program. In your report, explain what changes in timing you do or don't see.

7. In your report, give a table of your time evaluations, and graph the three methods against each other. Write a short paragraph describing why the time estimations you see might make sense

8. Additionally, give a table of accuracy as problem size increases, and graph accuracy against number of samples as you increase number of samples.

9. Include all your code in a zipped file submitted to gradescope alongside a README that explains how to compile and run all three versions of your code.



Program 2: Vector summation For this program, you will be required to perform vector summation of a set of vectors. Each vector is of dimension k. The first vector is [0, 1, 2, ..., k-1]. The second vector is [k, k+1, ..., 2k-1]. The j-th vector is [j*k, j*k+1, ..., j*k+k-1]. Those vectors are evenly assigned to p MPI processes using a block mapping. Each process generates the initial values for its local vectors that it owns. Then you develop two version of implementation. One is to use basic MPI send and receive functions to communicate. The second version uses MPI reduce function.

Requirements and settings

1. Include a test function in your program that allows command line input of n and k. After running, the test function should print out the first 30 elements of the result vector for verification, as well as the total time taken.

2. Test both versions of your program on large problem sizes on a single core, recording timing. Repeat such tests with 2, 4, 8 (if available), and your maximum number of cores. For all numbers of cores, you should stop increasing problem size when it takes more than a minute to run your code.

3. In your report, graph your times as problem size and number of cores increases for basic communication and using MPI reduce. Calculate speedup and efficiency for your highest number of cores. In a short paragraph, describe why you see the speedup/efficiency results you do.

4. In a README attached to your code describe how to compile your code and how to run with the test function active with an example. Make sure your code is compilable on CSIL, as we will test your code.

Turn in Instructions: Please turn in the homework via gradescope by 9 PM on Monday July 25th, 2022. For this assignment, there will be two separate submissions on gradescope. Submit your code in a with an included makefile to hw3_pa and submit your report to hw3. For this programming assignment you will be allowed to submit in pairs. If you are submitting with a partner, please link your partner on your Gradescope submission.