Homework Assignment-4

Due by 11:59pm on 24th April 2023

(Total weightage: 6%)

Instructor: Vivek Kumar

No extensions will be provided. Any submission after the deadline will not be evaluated. If you see any ambiguity or inconsistency, please seek a clarification from the teaching staff.

Plagiarism: **This is an individual assignment.** All submitted programs are expected to be the result of your individual effort. You should never misrepresent someone else's work as your own. In case any plagiarism is detected, it will be dealt **strictly** as per the new plagiarism policy of IIITD.

Please also note that discussing the approach/solution of assignment with your batch mate, i.e. using your friend's idea is also plagiarism.

Instructions

1) Hardware requirements:
    a. You will require a Linux/Mac OS to complete this assignment.
    b. You can easily code/complete this assignment on your laptop.
2) Software requirements are:
    a. MPI library.
    b. GNU make.
    c. You **must** do version controlling of all your code using github. You should only inside a **PRIVATE** repository. If you are found to be using a **PUBLIC** access repository then it will be considered as plagiarism.

Learning Objectives

1) Hybrid parallel programming using MPI+OpenMP

Problem Description

There are two programs that you have to implement in this assignment by using MPI/MPI+OpenMP as mentioned below.

Assignment-4a: Iterative Averaging

The goal of this assignment is to solve a One-Dimensional Iterative Averaging problem by developing a MPI+OpenMP, where MPI will be used for inter-process communication and OpenMP will be used for intra-process computation. You can use point-to-point or collective communications in MPI as appropriate.

Your program should take the following three inputs on the command line:

1. Input size, **N**, of type long.
2. Total number of iterations **NI**.

3. Number of processors, **P**, of type int (to be supplied to mpirun).

The code for a sequential and HClib implementation of iterative averaging is already available in your Github course repository. Given an input size, N, a solution to this iterative averaging problem must perform the following steps:

1. Allocate a (logically) global array of doubles, A, of size N+2 partitioned across the P processors. You may also allocate a second shadow array for A, if needed.
2. Initialize A[0] := 0 and A[N+1] := N+1.
3. In each iteration, replace A[i] by the average of its neighbors, i.e., A[i] := (A[i-1] + A[i+1])/2 for all 1 <= i <= N. This computation will entail communication of boundary elements on each processor.
4. Repeat the previous step until the iteration count NI is reached.
5. Print the Sum of final values of A[1...N].
6. Your program should measure the elapsed time of the iterative part of the program (steps 3 and 4 above), which excludes initialization and final output. You can run your program in two ways: a) P=2, OMP_NUM_THREADS=2, and b) P=4, OMP_NUM_THREADS=1

Assignment-4b: Matrix Multiplication

You can use the 3-dimension for-loop style of matrix multiplication that we have discussed several times in FPP lectures. You have to implement a distributed memory parallel implementation (MPI-only) of this matrix multiplication as explained below.

1) Your program will be tested with NxN matrices where N=1024
2) You should save your program as mm_mpi.c
3) Program flow
    a. Master process in MPI_COMM_WORLD gets the size of the matrices to be multiplied.
    b. Each MPI process allocates the matrices a, b and c.
    c. Master process initializes the input matrices a and b. It then transmits the content to all other processes.
    d. Each process will work on independent row(s) of the output matrix c.
    e. After final multiplication each MPI process transmits the local content to the root process that then validates the output.
    f. Your program should measure the elapsed time of the parallel part of the program (steps d and e above), which excludes initialization and final output/validation.