Parallel Processing - 2023

Assignment 2 - MPI

Deadline & Submission:

- 1. **Teams:** Form a team of two students from the same group or the same TA.
- 2. Upload it on Classroom with file named: A2_student1ID_student2ID_GroupName.zip e.g., A2 20130001 20130002 S1 S2.zip
- 3. Code must be in C language, and MPI & you must run it before sending.
- 4. Attach a screen shot from the console output for each problem.
- 5. Cheating could lead to serious consequences.

Problem Statement: "Counting Primes"

Write a parallel C program to count the prime numbers within an input range using the following two methods, then <u>compare the execution times</u> of both programs:

- a) MPI_Bcast and MPI_Reduce ONLY
- b) MPI Send and MPI Recv ONLY

Given

- Lower bound number x
- Upper bound number y

Output

- Count of prime numbers occurring between x and y.

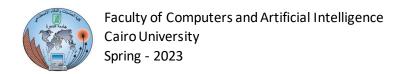
Parallelization Scenario

Master Process:

- Calculate the subrange size $\mathbf{r} = (\mathbf{y} \mathbf{x}) / \mathbf{p}$ (if including master) or $(\mathbf{y} \mathbf{x}) / (\mathbf{p} \mathbf{1})$ processes (without master).
- Broadcast \mathbf{x} and \mathbf{r} to each slave process using MPI_Bcast (or loop of MPI_Send).
- Receive sub-count from each slave process using MPI_Reduce (or loop of MPI_Recv).
- Print total count of primes between x and y.

Slave Process:

- Receive **x** and **r** through the MPI_Bcast call (or MPI_Recv).
- Calculate the lower bound **a**, and upper bound **b** according to its rank.
- Count primes in its subrange (between **a** and **b**).
- Send this partial count to the master process using the MPI_Reduce call (or MPI_Send).



Example:

```
n = 4, x = 1, y = 16 \rightarrow r = (16 - 1) / (4 - 1) = 5
p1: calculate partial count of prime numbers from 1 to 5 \rightarrow Count = 3 (2, 3, 5)
p2: calculate partial count of prime numbers from 6 to 10 \rightarrow Count = 1 (7)
p3: calculate partial count of prime numbers from 11 to 15 \rightarrow Count = 2 (11, 13)
After reduction, P0 will have Count = 6 (2, 3, 5, 7, 11, 13)
```

Note: The length of the range may not be divisible by the number of processes. So, you should handle this case.