LEBANESE AMERICAN UNIVERSITY
Department of Computer Science and Mathematics
CSC 447: Parallel Programing
Spring 2021

Lab III (MPI)

Problem 1: Parallel Merge Sort

Write an MPI C program that takes as command line argument the name of a text file containing first an integer n followed by an array of n integers. Your program should first read from the text file the array elements then apply a parallel implementation of merge sort algorithm to sort the array in parallel. The final sorted array elements should be printed out into an output text file. Save your program with the name *YourName*_Prob1.c.

Test your program on the provided text file *Input1.txt*

Compile and run this MPI program and verify its output by following these steps:

- Use an IDE of your choice to create and save your program with *YourName*_Prob1.c.
- Compile your program within the IDE or from terminal window by using the command: mpicc –o Problem1 YourName_Prob1.c
- Then, run your MPI program by typing the command in a terminal widow: mpirun -n <NumCores> <compiledProgram>
- Verify that your program gives the correct output for 1, 2, and 4 cores

Measure the execution time in your program using the function $MPI\_Wtime()$. This can be done as follows:

```
Double start_time = MPI_Wtime();
//Insert your parallel code here
if(rank==0) {
double end_time=MPI_Wtime();
printf("Wallclock time elapsed: %.8lf seconds\n",end_time-
start_time);
}
```

- Report in a tabular form the execution time of your program when using 1, 2, and 4 processors.
- Draw your conclusions on the speedups obtained.

Problem 2: Primality Testing

Write a program that takes an integer as a command line argument and checks if it's a prime number using MPI.

Your program should print out the result of the primality test of each process then the final result.
Sample input/output:
- If the input number is 1299829, the output if you run your program using two processes should be:

```
Local Result of process 0 is not Prime because it is divisible
by 59. Local Result of process 1 is Prime.
The Number 1299829 is not Prime
```

- If the input number is 1299827, the output if you run your program using two processes should be:

```
Local Result of process 0 is Prime. Local Result of process 1 is
Prime. The Number 1299827 is Prime
```

- Save your program with the name *YourName*_Prob2.c.
- Report in a tabular form the execution time of your program when using 1, 2, 4, 8, 16, and 32 processors.
- Draw your conclusions on the speedups obtained.
- Save your program with the name *YourName*_Prob2.c.

Note: A report should be submitted where you describe your parallel implementation and report in, tabular form, the execution times and speedups obtained.

Submission Instructions:

You are required to submit your solutions **by Tuesday March 8 at 8:00 am**. Place your code and report in one zipped folder and submit on the provided submission link on Blackboard.