Middle East Technical University Department of Computer Engineering

CENG 478
Introduction to Parallel Computing

Spring 2018-2019

Assignment 2

Due date: 07.04.2019, 23:59

1. Sorting in Parallel (50 Pts)

   a. You will implement a parallel algorithm that sorts an array of double numbers using MPI processes and compare the results with a serial quicksort program. A sample code for serial quicksort is given.

   b. The input has the following format:

      Number_of_items_to_be_sorted
      Minimum-element Maximum-element
      Input_1
      .
      .
      .
      Input_N

      You can also use the input handling method of the given quicksort code for your implementation. The sample input has 1,000,000 numbers. Your codes will not be tested on a larger input than this.

   c. Implement a parallel algorithm to sort the array you scan. Calculate the time consumed with MPI_Wtime(). You can choose one of two methods for your sorting algorithm: **Parallel odd-even transposition** or **Parallel bucket sort**. Your output should have the following format:

      Sorting_Method: time-consumed
      Sorted_Number_1
      .
      .
      .
      Sorted_Number_N

      (The warnings given by slurm are not important)

2. Tests and Report (50 Pts)

   a. Compile and run your code, calculate the time consumed for 1,2,4,8,16 and 32 processes on the given sample input. Plot a "Time vs. Number of processors" graph. Plot a "Speed Improvement vs. Number of processors" graph. Comment on how the time and efficiency changes. What is the cause of this increase or decrease? Speculate on what can be done to improve performance further.

   b. Compile and run quicksort code with only 1 process in slurm. (It is implemented in serial, if you run it with more than one process, it will not work properly) Compare your time results with the time consumption of the serial quicksort code.

3. Notes

   a. You will submit a tar file consisting of your code file(s), your makefile, a **pdf** of your report via ODTÜClass. Note that the **comments and comparisons** on your report will **have a large effect on your grade**.

   b. Note that **zombie processes** can pile up very quickly as you are trying to learn a new way of programming. Try to **control frequently** if there are any, with *squeue* command and kill them with *scancel* command to ensure that Slurm serves all the users **properly**[1].

   c. Try to finish as early as possible. Since all of you work on the **same** HPC, the waiting times for the **queue** can be **huge** which will prevent you from testing your code **efficiently**.

   d. Implementing MPI macros **does not mean** your code works in **parallel**. You have to design your algorithm so that it really works in parallel. For those who submits code that does not work in parallel will get **no credits** from this assignment.

   e. You can still submit your work if the **deadline** is passed, however with an increasing **penalty** of **5*days*days**. (i.e. first day -5 points, second day -5*2*2=-20 points and so on). Note that even a minute late means that it is the other day.

   f. We have zero tolerance policy for cheating. People involved in cheating will be punished according to the university regulations and will get zero.