COMP 4901Q: High Performance Computing (HPC)

Programming Assignment 3

Due date: **16 April 2022 23:59**

Notes

1. Each problem counts 10 points. Totally 20 points contribute 10% of the overall credit of the course.
2. All submitted code will be compiled and tested on the lab 2 machines to evaluate the assignments. The code might be run on multi machines.
3. Points may be deducted if your programs consistently achieve no speedup over the serial program.
4. Write your code only in the specified area. The code will automatically compare the answer of parallel version and serial version when running. Points may be deducted if your programs output extra debug information.
5. You might need more flexible function to scatter and gather data. You can refer to the following link for the advanced usage:
   (1) scatterv: https://www.mpi-forum.org/docs/mpi-1.1/mpi-11-html/node72.html
   (2) gatherv: https://www.mpi-forum.org/docs/mpi-1.1/mpi-11-html/node70.html
   But if you find it cannot speed up your program, it's not necessary to use them in your program.

Problem 1

Description

Please use MPI to write a program to calculate the arc length of a function: $y = ax\text{^}3 + bx\text{^}2 + cx + d$,where $x \in [l, r]$.

We sample n points to calculate the arc length approximately. More specifically, let (x_i, y_i) be the coordination of point p_i where $0 \leq i < n$. We have $x_0 = l, x_{n-1} = r$, and $\forall i \in [1, n-2], x_i - x_{i-1} = x_{i+1} - x_i$.
We use following formula to calculate the arc length.

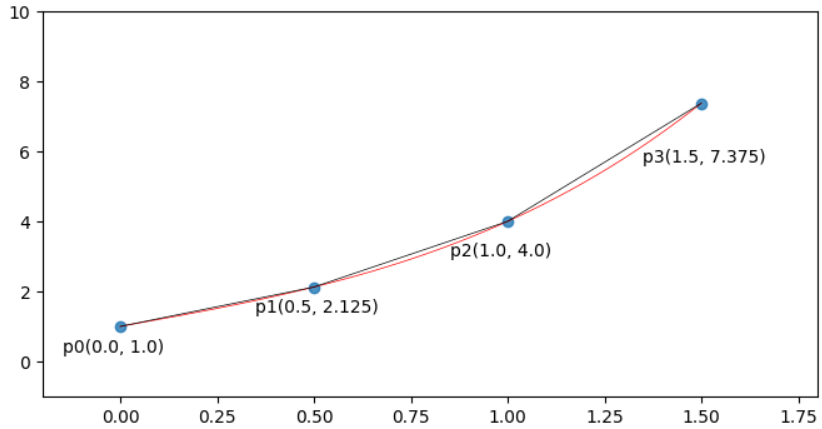
$$ArcLength = \sum_{i=1}^{n-1} d(p_{i-1}, p_i)$$

Where $d(u, v)$ is the Euler distance between u and v.

Example

For the function $y = x^3 + 2x^1 + 1, l = 0, r = 1.5, n = 4$,. As shown in following figure, the red curve is the function, and the blue points are the sample points. The black line is the segment connecting two adjacent sample.

$$ArcLength = \sum_{i=1}^{n-1} d(p_{i-1}, p_i) \approx 6.583465078$$



Sample code

The sample code is arc_length.cpp. The load data part, sequential calculation part and the test part has been implemented. Please write your code only in the specified area and do not output extra debug information.

In the sample code, the input data are in the vector named "input_data". Each element in "input_data" represent a query. You need to answer all queries and put answers into the array named "parallel_answer". The length of "parallel_answer" and "input_data" should be the same.

Some small sample input data and large sample input data are provided for debugging. When testing, we will use some large input data.

Compile and run

You can use to compile your program

mpic++ --std=c++11 -o arc_length arc_length.cpp

The running command should be this format:

mpiexec -n [process number] [--hostfile hostfile] arc_length [data_file_name]

For example:

mpiexec -n 2 arc_length small_input_data_1.txt

mpiexec -n 4 --hostfile hostfile arc_length small_input_data_1.txt

Problem 2

Description

Please use MPI to write a program to calculate match number array of two binary string st_1 and st_2. The detail of the definition is following.

We use $st[i]$ to represent the i^{th} character of a string st. The index is start from 0. We use $st[i \dots j]$ to represent the substring of st which is from i^{th} character to j^{th} character.

Let $len(st)$ be the length of the string st.

Define a function $match(a, b)$ be a function to calculate the number of equal bits of a and b, where a and b are two binary strings with same length, i.e.

$$match(a, b) = \sum_{i=0}^{len(a)-1} a[i] == b[i]$$

The match number array consists of $len(st_1) - len(st_2) + 1$ numbers. We use $q[i]$ to represent the i^{th} number of match number array. It can be calculated by following formula:

$$q[i] = match(st_1[i \dots i + len(st_2) - 1], st_2), \forall i \in [0, len(st_1) - len(st_2)]$$

Example

$st_1 =$ "10101100", $st_2 =$ "100"
$q[0] = match("101", "100") = 2$
$q[1] = match("010", "100") = 1$
$q[2] = match("101", "100") = 2$
$q[3] = match("011", "100") = 0$
$q[4] = match("110", "100") = 2$
$q[5] = match("100", "100") = 3$

Sample Code:

The sample code is match.cpp. The load data part, sequential calculation part and the test part has been implemented. Please write your code only in the specified area and do not output extra debug information.

In the sample code, the input data are in the strings named “input_st1” and “input_st2”. You need put the answer into the array named “parallel_answer”.

Some small sample input data and large sample input data are provided for debugging. When testing, we will use some large input data.

Compile and run

You can use to compile your program

mpic++ --std=c++11 -o match match.cpp

The running command should be this format:

mpiexec -n [process number] [--hostfile hostfile] match [data_file_name]

For example:

mpiexec -n 2 match small_input_data_1.txt
mpiexec -n 4 --hostfile hostfile match small_input_data_1.txt