CSE 625 Parallel Programming Project 3

October 20, 2022 (100 points)

Due: Novenmber 4 (Friday) midnight (Submit your project and report and notebook to the Blackboard)

Readings:

[1] 04\_C++\_11\_MultiThreading\_1.pdf

[2] 04\_C++\_11\_MultiThreading\_2.pdf  
[3] MNIST\_2 All\_pair\_distance.pdf

CodeBlocks Project:

All\_Pair\_Distance

Python Notebook and data:   
  
 MNIST\_Dataset.zip

Download link: <https://louisville.box.com/s/g5ssgz9jgttxtd1u09o5xe2077gowfkj>

It contains MINIST dataset   
  
 train-images.bin (60,000 28x28 float32 images)

train-labels.bin (60,000 ubyte labels)

test-images.bin (10,000 28x28 float32 images)

test-labels.bin (10,000 ubyte labels)

and notebook, MNIST\_Dataset.ipyn.

Assignment Description

1(40 points)

In the posted notebook, Mis\_match.ipynb, it defines a list called mis\_match,

which contains a list of 309 ordered pairs (test\_idx, train\_idx). For example, the first   
 pair is (115, 8111) and it will be used to pair and compare 115-th test image with   
 8111-th trainimage given in the MNIST dataset (i.e., test-images.bin and

train-images.bin).

Based on this notebook, expand it to perform the following tasks:

1. Read the MNIST dataset into numpy arrays.
2. Plot 10 pairs in the list, mis\_match, in a single plot. For each image in the plot, include a title of index of test or train image and its test or train label.
3. Take a user input of the index (between 0 and 308) of the pair in the list, mis\_match, and plot the pair of images side-by-side with appropriate title (like that mentioned in task 2).

Show a run of your notebook in the project report and submit your notebook to the   
 Blackboard.

2 In the CodeBlocks project, All\_Pair\_distance, it implements four functions to   
 compute the pair-wise distance matrix of MNIST train images (loaded from  
 train-images.bin). These four methods are:

1 sequential\_all\_pairs (sequential computing)   
 2 block\_all\_pairs (C++ multi threads - block work distribution)

3 block\_ cyclic\_all\_pairs (C++ multi threads - block cyclic work distribution)

4 dynamic\_all\_pairs (C++ multi threads - dynamic work distribution)

2 .1 (22 points) Measure the runtimes of using these four functionss to compute the  
 pair-wise distance matrix of various sizes and put the results in the following table:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Matrix Size | 400 | 800 | 10,000 | 20,000 | 30,000 | 60,000 |
| Method 1 |  |  |  |  |  |  |
| Method 2  12 threads |  |  |  |  |  |  |
| Method 3  12 threads  Chunk size 2 |  |  |  |  |  |  |
| Method 4  12 threads  Chunk size 2 |  |  |  |  |  |  |

* 1. (8 points) In the report, explain the key ideas of the function, dynamic\_all\_pairs, of its work distribution implementation and how and why the std::mutex object is used.

3(15 points) In problem 2, suppose we use 3 threads in the function, block\_all\_pairs, to   
 compute the pair-wise distance matrix of size 60,000-by-60,000 of the MNIST train   
 images. Calculate the work amount (i.e., the number of outmost iterations) done by   
 each thread. Show your work and the results. (Ref: MNIST\_2 All\_pair\_distance.pdf)

4(15 points) In problem 2, suppose we use 3 threads and chunk size 1 in the function,   
 block\_cyclic\_all\_pairs, to compute the pair-wise distance matrix of size   
 60,000-by-60,000 of the MNIST train images. Calculate the work amount (i.e., the   
 number of outmost iterations) done by each thread. Show your work and the results.

(Ref: MNIST\_2 All\_pair\_distance.pdf)