COMP 4901Q: High Performance Computing (HPC)

Programming Assignment 2

Due date: **27 March 2022 23:59**

Notes

1. Each problem counts 10 points. Totally 30 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.
3. Points may be deducted if your programs consistently achieve no speedup over the serial program.

Problem 1: Parallel Matrix-Vector Multiplication using CUDA

Please use CUDA to parallelize matrix-vector multiplication. Your program should:
(1) Request the user to input the size of the matrix, say Row x Col; then the vector must have size Col;
(2) Create and initialize the matrix and vector with random floating-point numbers;
(3) Perform matrix-vector multiplication on CPU and measure its running time;
(4) Perform matrix-vector multiplication on GPU and measure its running time; You don't need to measure the data transfer time (i.e., memory copy between CPU and GPU).
(5) Compare the CPU results with GPU results;
(6) Output the size of the matrix, CPU running time, and GPU running time.

Problem 2: Parallel Matrix Transpose using CUDA

Please use CUDA to parallelize matrix transpose. Your program should do the followings:
(1) Request the user to input the size of the matrix, say Row x Col;
(2) Create and initialize the matrix with random floating-point numbers;
(3) Perform matrix transpose on CPU and measure its running time;
(4) Perform matrix transpose on GPU without using shared memory and measure its running time;
(5) Perform matrix transpose on GPU with shared memory and measure its running time;
(6) Verify the results of your GPU kernels by comparing with the CPU version.
(7) Output the size of the matrix, CPU running time and two GPU running times.

Problem 3: Parallel convolution using CUDA

Please use CUDA to image convolution operation. Your program should do the followings:

(1) Request the user to input the size of the image, say Row x Col;
(2) Request the user to input the size of the convolution kernel, say K x K;
(3) Create and initialize the image and the kernel with random integer numbers;
(4) Perform convolution on CPU and measure its running time;
(5) Perform convolution on GPU and measure its running time;
(6) Verify the results of your GPU kernels by comparing with the CPU version.
(7) Output the size of the image and kernel, CPU running time and the GPU running time.