**Lab 2: Histogram (Due on 10/23/2020)**

Objective

Implement a histogram routine using atomic operations and shared memory in CUDA. Your code should be able handle arbitrary input vector sizes (vector values should be randomly generated using integers between 0~1024).

Instructions

Make a new folder for your project. Copy and modify the code of histogram in CUDA sample code to include the following key functions:

1. allocate device memory
2. copy host memory to device
3. initialize thread block and kernel grid dimensions
4. invoke CUDA kernel
5. copy results from device to host
6. deallocate device memory
7. implement the routine using atomic operations and shared memory
8. handle thread divergence when dealing with arbitrary input sizes

Your final executable can be run using the following command:
./histogram_atomic -i <BinNum> <VecDim>

Input parameters:
<VecDim> is the dimension of the input vector.
<BinNum> is the number of bins (any integer that can be written as 2^k where k can be any integer from 2 to 8).

Questions

Consider an N-dimensional input vector for the histogram computation with M bins:

(1) Describe all optimizations you tried regardless of whether you committed to them or abandoned them and whether they improved or hurt performance. Which optimizations gave the most benefit?

(2) How many global memory reads (write) per input element are being performed by your kernel? Explain.

(3) How many atomic operations are being performed? Explain.

(4) How many contentions would occur if every element in the array has the same value? What if every element has a random value? Explain.

(5) How would the performance (GLFOPS) change when sweeping <BinNum> from 4 to 256 (k=2 to 8)? Compare your predicted results with the realistic measurements when using different thread block sizes.

(6) Propose a scheme for handling extremely large data set that can not be fully stored in a single GPU's device memory. (Hint: how to design an implementation for efficiently leveraging multiple GPUs for parallel histogram computation?)

Report Submission

Your Lab report should be submitted via Canvas with the source code folder (zipped) no later than the required due date. In the report, you have to:

1. Answer all the above questions.
2. Include important implementation details for developing your CUDA program.
3. Demonstrate extensive experimental results, such as compute throughputs (GFLOPS), using different combinations of input parameters and thread block sizes.
4. Discuss your results in your report using the knowledge learnt from our classes.