Sergi Aliaga

Northeastern University

Homework 5

Question 1

Let us revisit the histogramming problem assigned in Homework 4. Your input to this program will be integers in the range 1-10,000,000 this time (use a random number generator that generates the numbers on the host). Your host-based data set should contain N integers, where N can be varied.

a. *This time you will implement a histogramming kernel in CUDA and run on a GPU. You can choose how to compute each class of the data set on the GPU. Attempt to adjust the grid size to get the best performance as you vary N. Experiment with N = 2^10, 2^15, 2^20 and 2^25. When the GPU finishes, print one element from each class in class ascending order on the host (do not include the printing time in the timing measurements, though do include the device-to-host communication in the timing measurement). Plot your results in a graph.*

The code for this section of the assignment is included in the file "*Q1a.cu*". The code was inspired in from the one shown in https://www.youtube.com/watch?v=DaEmuL0PYxc&t=1251s. The version of CUDA used is 10.2 and the version for gcc is 6.4.0. The output of the code is displayed in Figure 1, where one element of each class is displayed.

```
[aliaga.s@c2177 Question1]$ ./Q1a
Total computing time: 3.058549
Elements of each class of the 100 bins:
80852, 123013, 285004, 364889, 407560, 565759, 688136, 761632, 899137, 988576, 1090870, 1109147, 1297374, 1384498, 1406841, 1554768, 1676485, 1791384, 1894527, 1918714, 2081755, 2
126451, 2227294, 2365311, 2483951, 2577886, 2675674, 2706720, 2843090, 2947492, 3073665, 3188004, 3293417, 3348544, 3412153, 3549574, 3650358, 3705517, 3802760, 3917915, 4026382,
4198897, 4219746, 4397597, 4426649, 4514151, 4620334, 4733447, 4879758, 4953546, 5016463, 5190478, 5208356, 5330680, 5481800, 5526917, 5631611, 5733117, 5855508, 5932396, 6071224,
 6155507, 6278630, 6391985, 6470588, 6553339, 6615181, 6757504, 6840821, 6984423, 7067504, 7140912, 7209770, 7355234, 7400046, 7584909, 7605358, 7706113, 7825974, 7996780, 8010806
, 8166674, 8244436, 8313389, 8461297, 8526426, 8689240, 8767698, 8848765, 8946510, 9091462, 9102615, 9221320, 9313379, 9453429, 9587750, 9674781, 9707347, 9894828, 9950660,
Total number of elements counted: 33554432
```

Figure 1: Output of the histogramming program.

Figure 2 shows the computing time results for the suggested values of N. The program seems to escalate exponentially with the input size growing exponentially too. However, the relative increase in computational time is very reduced (compared to OpenMP, as shown in the next section), with an **15%** increase between N=2^10 and 2^25.

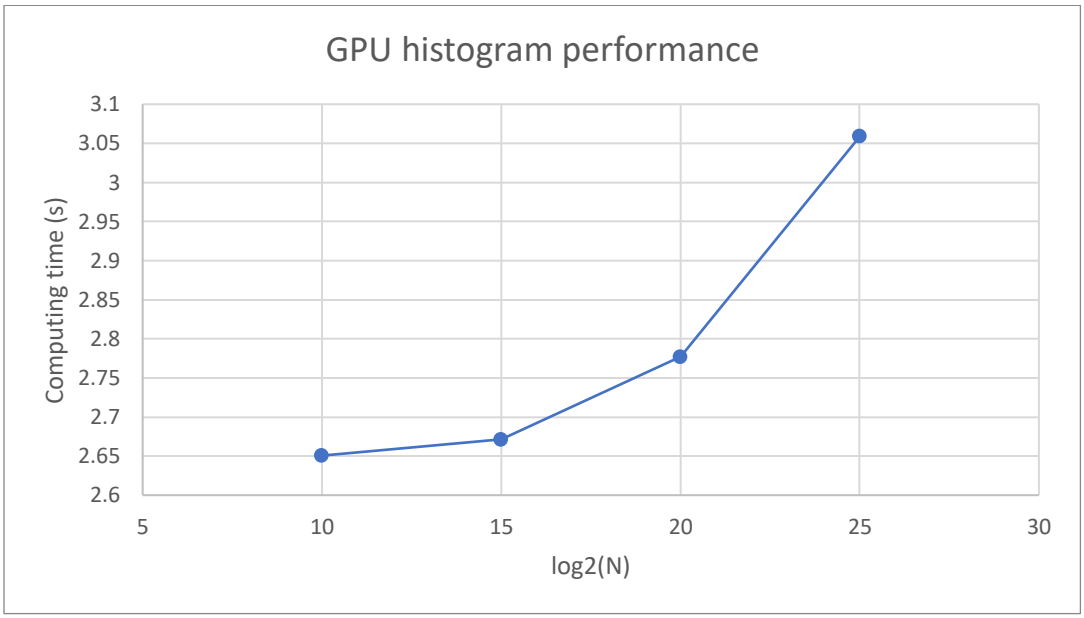

Figure 2: Performance of the histogramming kernel in CUDA

b. Compare your GPU performance with running this same code on a CPU using OpenMP.

The code used to test the histogramming problem with OpenMP is included in the file *"Q1b.c"*. Figure 3 shows the results of this implementation with the same inputs as the previous section. On the one hand, the absolute computing time values ar much lower, probably because of getting rid of the device-to-device communication between CPU and GPU. However, on the other hand, we observe poor scalability compared to the GPU version, with an **9e5%** relative increase between N=2^10 and 2^25, showing a clear scalability advantage in using GPUs.

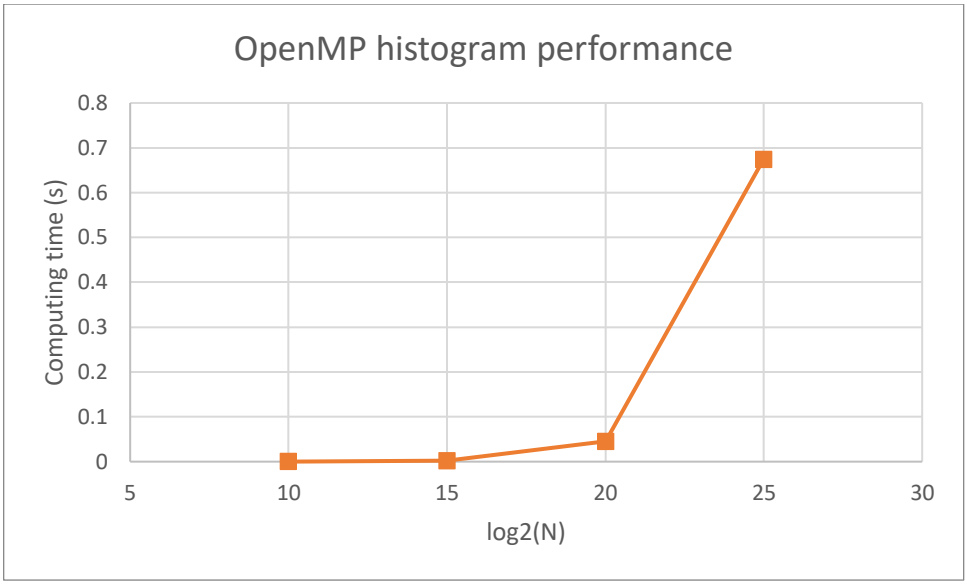

Figure 3: Performance of the histogramming kernel in OpenMP