Justin Liu | 504-487-373
COM SCI 133 | Dis 1B
2017-05-18

Lab 3

Parallelization

This program is parallelized by taking advantage of the low overhead of thread creation in GPUs to create large numbers of threads that each operate on independent pieces of data at a time. In my final submission, I used 256 threads because the bias array had 256 elements.

My solution uses only global and private memory. Each work-item loads the input arrays from global to private memory, does the necessary computations, and then loads the result back into global memory.

Scalability

# of Work-Items	Performance (GFlop/s)
32	35
64	28
128	25
256	33

Note_0: server was under variable load at the time of testing, so results not guaranteed to be accurate.
Note_1: this is using the kernel that accepts variable work-items. The submitted kernel expects 256 work-items.

The serial version runs at only 2-3 GFlop/s, so any parallelized version is vastly better. While it is hard to deduce any scalability trend from the data above, from testing I know it is most efficient when there is a 1:1 ratio between elements and work-items.

Technical Difficulties

I was very lost at the beginning, but after reading the Lecture 12 slides again I had a pretty good understanding of the project. Implementing the host was easy, but the kernel was tricky as I wasn't exactly sure what convolution was, or how it was supposed to be parallelized. Luckily from Piazza I learned it was supposed to be parallelized along i, which made everything make sense.