Homework 5- More CUDA

Compile and running instructions:

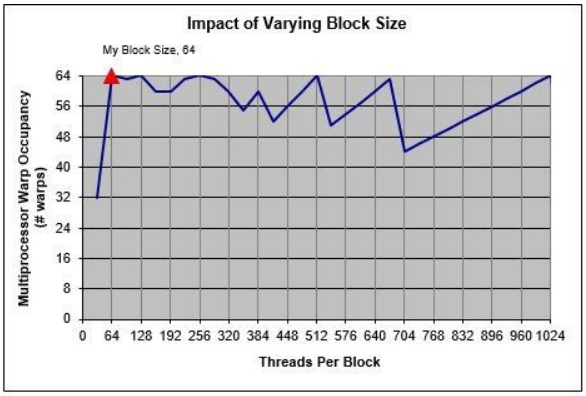
The code was compiled using the flags `--resource-usage`.
The code was executed on the peanut cluster (Titan) in interactive mode
The repo includes `CMakeLists.txt`.
Run `make hist_coal` for the coalesced version and make `hist_private` for the privatized shared memory version.
Run with **7** arguments: <filename> <numBins> <rangeStart> <rangeEnd> <gridDim> <blockDim> <numObservations>

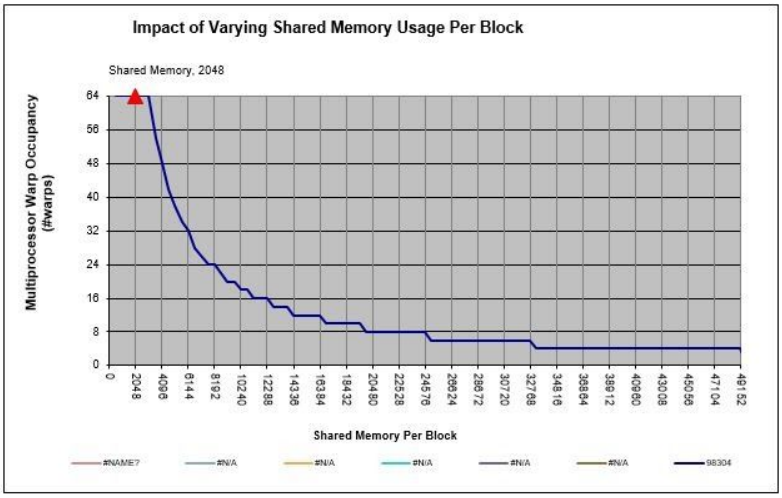
Functionality:

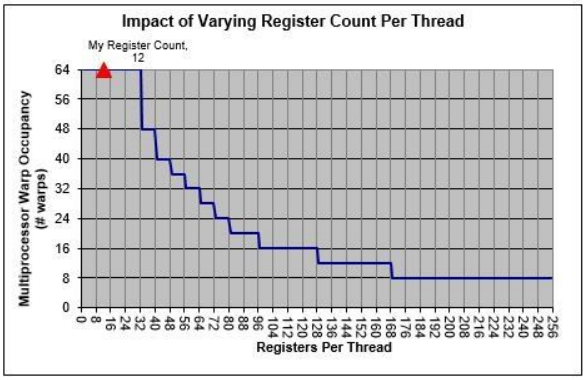
The repo contains two `.cu` files, `hist_coal.cu` and `hist_private.cu`. They are both programs that take continuous values and count them into bins of user defined ranges. The difference between the two is that the first one uses coalesced memory access between threads, and the second uses a shared memory buffer.

Optimum block size calculation:

Using the excel sheet from the Nvidia website, I calculated the optimum block size for my program, which used 12 &13 registers per thread, as reported by `--resource-usage`. Max occupancy was 100% at 64 threads per block.





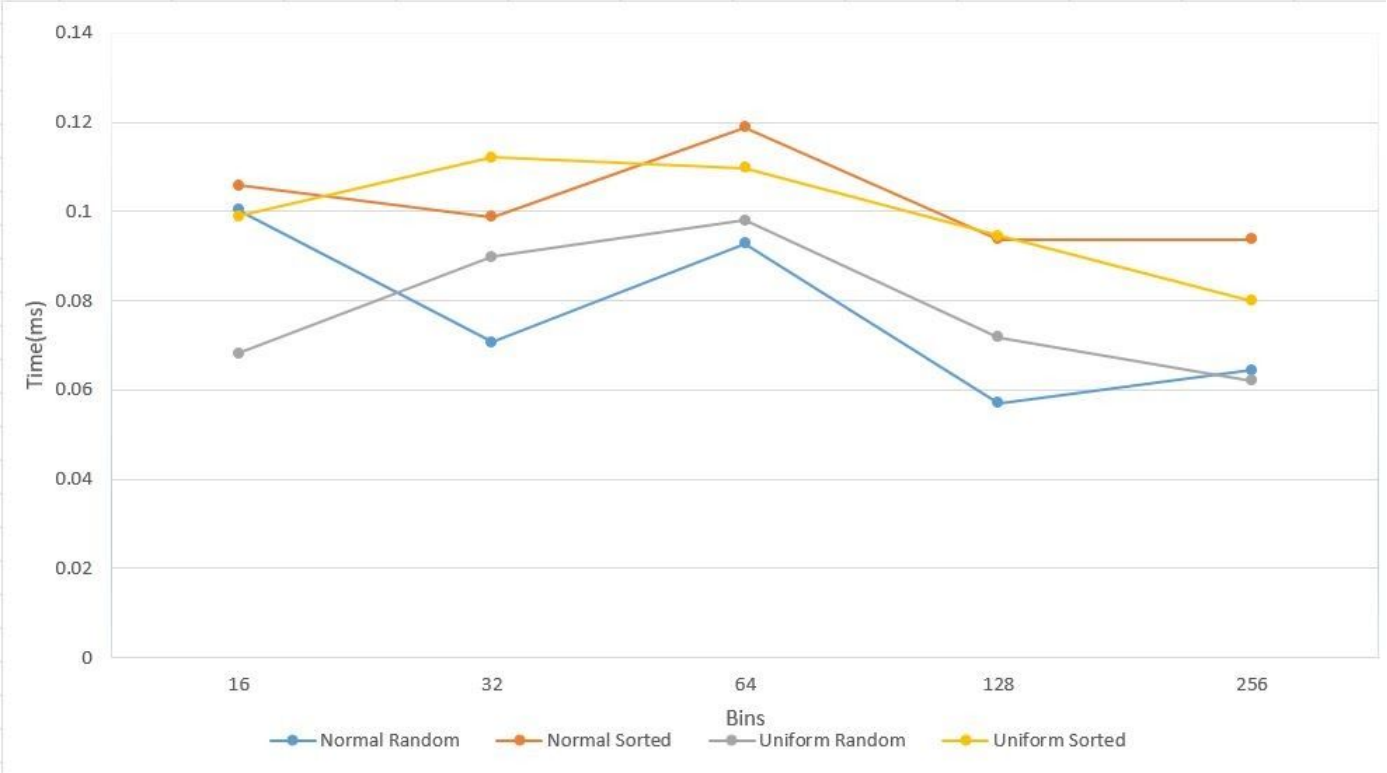


Timing tests:

The following is the graph for hist_coal run with the following command line arguments:

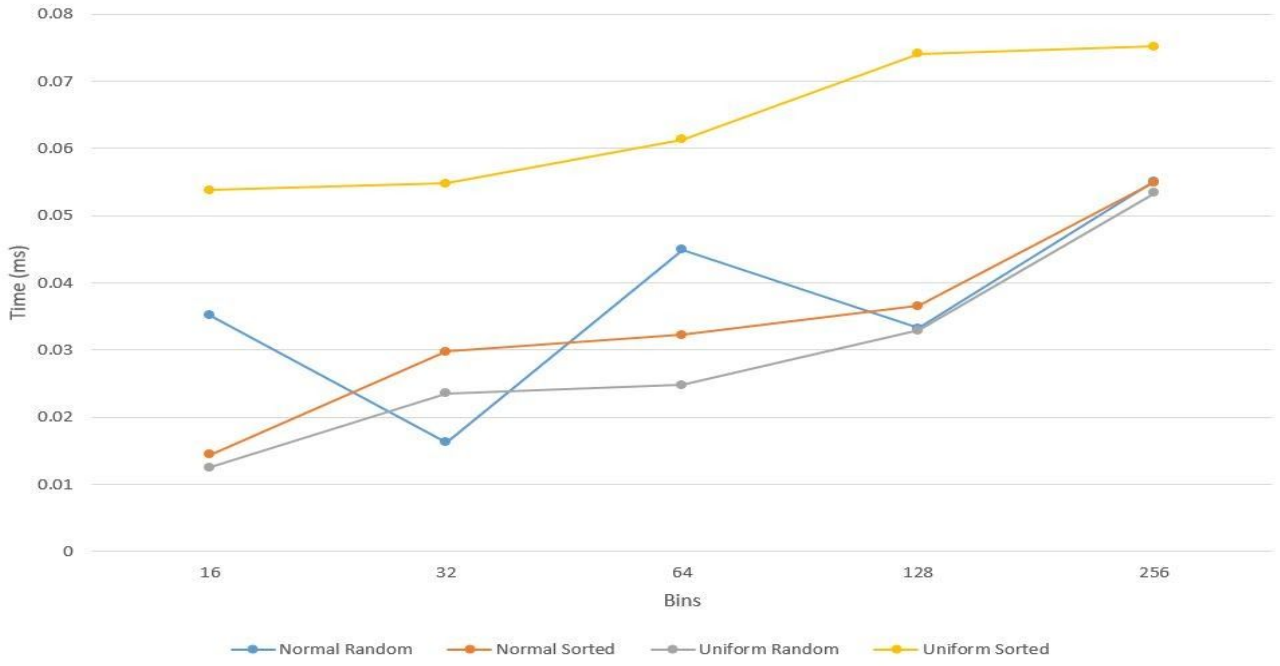
```
./hist_coal <filename> <bins> -4 4 1024 64 1310712
```

The last argument is the number of elements in the data set. I Used this to make fread easier.



The following is the graph for hist_private run with the following command line arguments:

```
./hist_private <filename> <bins> -4 4 1024 64 1310712
```



Observations: For the coalesced version, the results are generally what I expected. The normal_sorted and uniform_sorted datasets take longer to put into bins than the others. This is probably because the sorted nature of the data results in the same memory location being accessed by the threads. The random versions take less time on average because there are less collisions between threads.

For the privatised version, the program takes less time to run on average. This is probably because the shared memory is high bandwidth memory. But there is no real pattern between the sorted and random datasets, except for uniform_sorted, which took way longer than the others.