Nicholas Skinner

CS 475

Spring 2019

**Assignment 6 - CUDA Monte Carlo**

**Machine:**

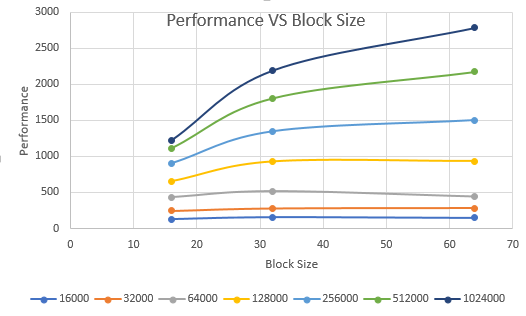
The machine I ran this on was the rabbit machine provided for the class to use.

**Tables:**

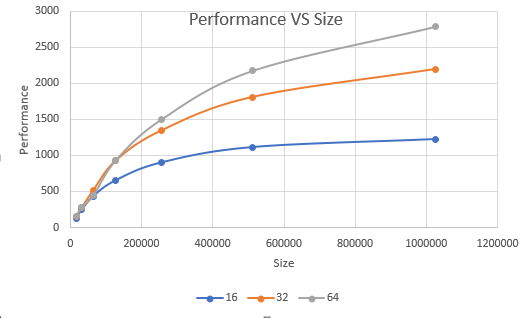
|  |  |  |  |
| --- | --- | --- | --- |
| **ArraySize/Block Size** | **16** | **32** | **64** |
| **16000** | **137.82** | **165.67** | **154.56** |
| **32000** | **253.23** | **282.41** | **288.18** |
| **64000** | **441.21** | **520.02** | **448.53** |
| **128000** | **663.68** | **936.55** | **942.29** |
| **256000** | **909.3** | **1351.12** | **1506.59** |
| **512000** | **1121.47** | **1814.06** | **2181.62** |
| **1024000** | **1232.38** | **2199.31** | **2791.83** |

**Graphs:**

**Performance vs. Block Size:**

****

**Performance vs. Size**

****

**Patterns in Performance:**

As we increase the block size, performance seems to go up as well, with 16 block size giving poor performance, we can almost see the performance peak at out 1024000 arrsize.

Also, I noticed (though it was not mentioned in the assignment notes) that the chance to hit the plate was within the range of ~ 41.3 percent on average.

**Why are these patterns there?**

I believe that these patterns exist due to the concept we learned called “warp”, where in cuda, threads are executing in parallel, and for optimal performance, multiples of 32 threads are recommended. I presume that due to lacking this requirement, the 16 block size has reduced performance

**Why is 16 BLOCKSIZE worse for performance?**

As stated in the previous segment, I believe that warps are causing some performance differences, as 16 is not divisible by 32, there appears to be some reduced performance from not possessing the excess slots.

**What does this mean for the proper use of GPU computing?**

For optimal cuda performance, we should be looking into applying multiples of 32 for projects, to help optimize the assets that cuda brings to the table.