CS 475/575 -- Spring Quarter 2021

Project #5

CUDA Monte Carlo

Name: Chun-Yu, Chen

E-mail: chench6@oregonstate.edu

1. **Tell what machine you ran this on**

   Machine: **DGX**

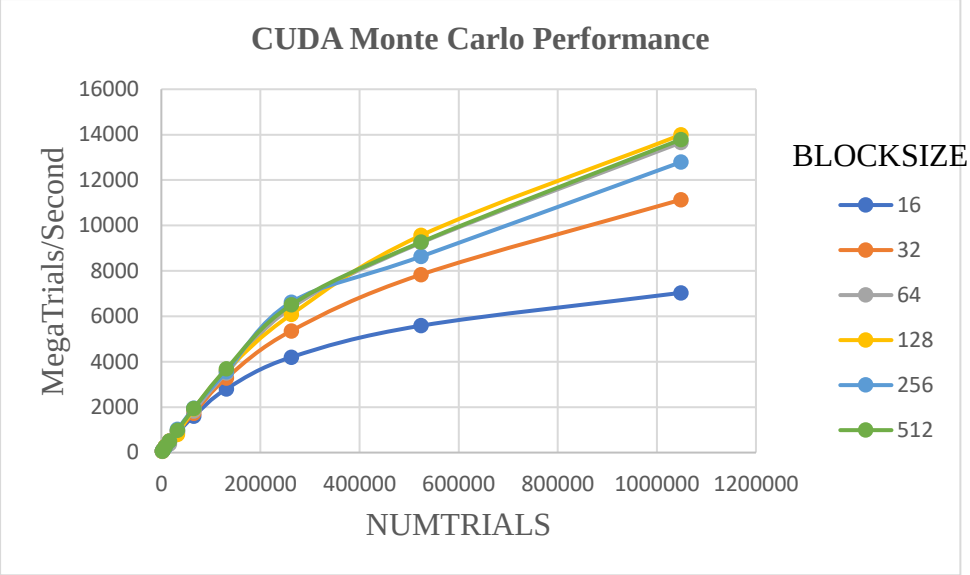
   Number of Trials = **1048576**

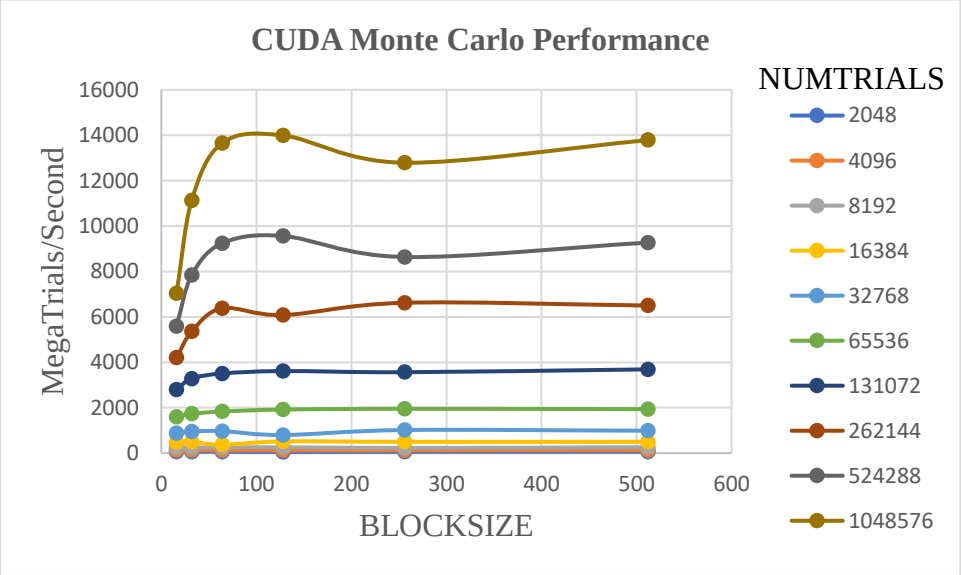
   MegaTrials/Second = **13785.4**

   Probability = **10.058%**

2. **Show the table and the two graphs**

BLOCKSIZE / NUMTRIALS	16	32	64	128	256	512
2048	62.5	64.5161	64.5161	54.0541	60.6061	64.5161
4096	125	125	125	125	125	129.032
8192	250	250	250	258.065	228.571	258.065
16384	484.848	500	372.093	516.129	500	500
32768	877.464	955.224	969.697	798.752	1018.91	988.417
65536	1600	1732.66	1833.48	1921.2	1954.2	1937.56
131072	2801.64	3282.05	3506.85	3615.18	3564.84	3690.09
262144	4201.03	5357.75	6380.06	6081.66	6622.47	6501.59
524288	5587.99	7839.23	9240.83	9564.51	8636.79	9272.21
1048576	7034.78	11134.2	13653.3	13991.5	12790	13785.4





3. Commentary

Based on the result, one thing can be found that BLOCKSIZE of 16 is much worse than the others since every 32 threads constitute a ”Warp”. Each thread in a Warp simultaneously executes the same instruction on different pieces of data. However, it is likely that a Warp’ execution has to stop at some point to wait for a memory access. This would make the execution go idle. The patterns of performance in each graph stably rise because of the increment of BLOCKSIZE and NUMTRIALS. The performance is much better than the project1 since there are more threads being used to computed in every grid. In this project, the mean for the proper use of GPU parallel computing is setting suitable and reasonable BLOCKSIZE and NUMTRIALS.