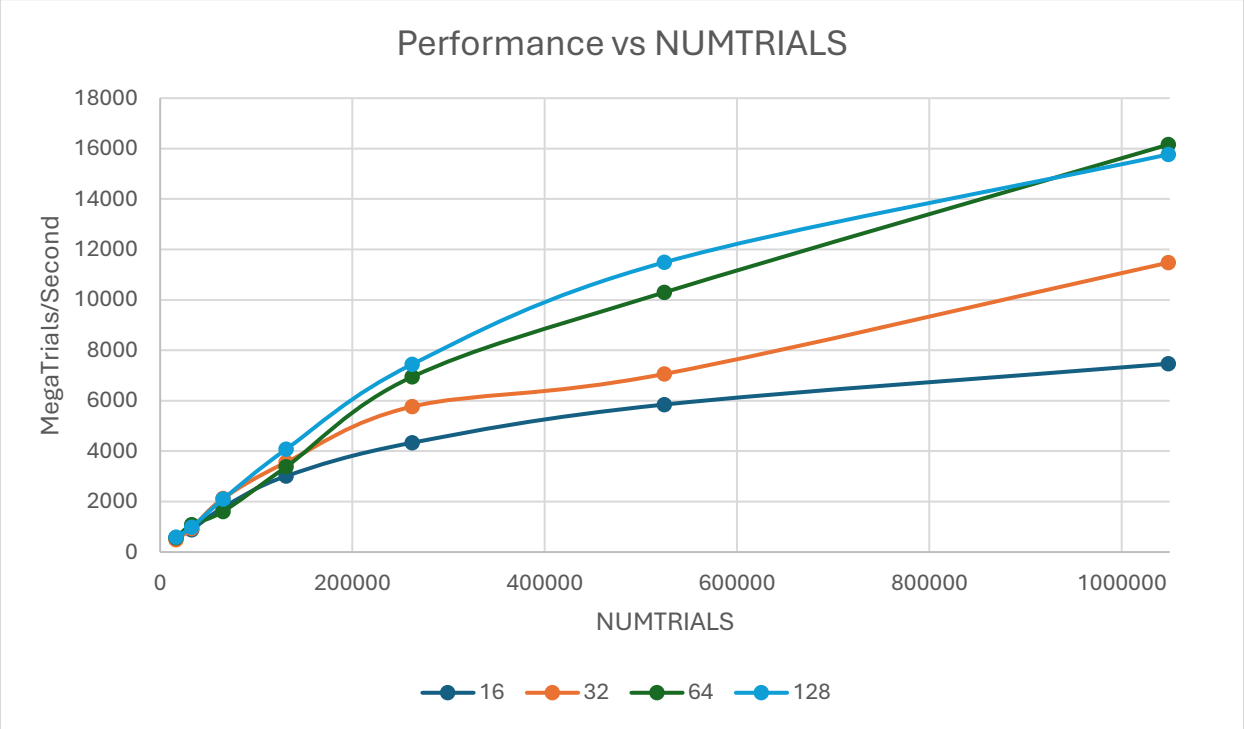
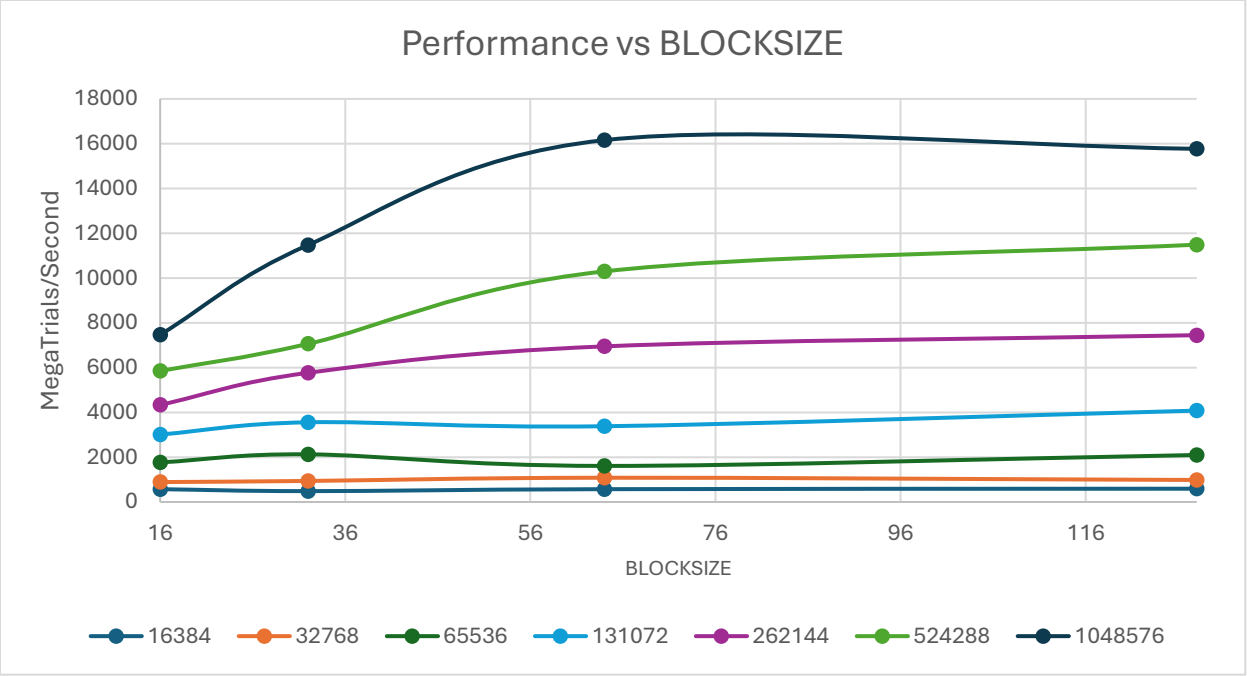
CS 475 – Parallel Programming

Jenny Zhong

Project #5 - CUDA: Monte Carlo Simulation

	16	32	64	128
16384	571.428564	484.848481	571.428564	592.592592
32768	888.888857	938.588415	1082.452448	982.725511
65536	1767.040586	2126.687438	1610.062864	2098.360696
131072	3011.764718	3558.644554	3382.328586	4079.681431
262144	4336.686196	5769.014142	6948.261159	7447.272768
524288	5849.339714	7062.069055	10297.92578	11489.48081
1048576	7467.639212	11477.40771	16157.79033	15769.00874





- Ran on OSU DGX System
- The new probability is likely **42.02%**
- Analyzing the graph for Performance vs NUMTRIALS, the performance has a positive correlation with BLOCKSIZE, as BLOCKSIZE increases performance increases. The lowest BLOCKSIZE 16 has the lowest performance. At a BLOCKSIZE of 128 and larger, performance begins to level out.
- In the Performance vs BLOCKSIZE graph, the largest dataset (or NUMTRIALS) 1048576 has the highest overall performance. CUDA performs better with larger datasets.
- BLOCKSIZE of 16 has the worst performance because using less than 32 BLOCKSIZE results in lower performance. A BLOCKSIZE of 8 would have lower performance than BLOCKSIZE of 16, etc.
- Compared to results from Project 1 using OpenMP, CUDA results in higher performance, which could be better for processing large datasets.
- Based on these results, with GPU parallel computing we are able to compute large datasets using thousands of cores running simultaneously. CUDA would be the most appropriate for tasks that are highly parallelizable.