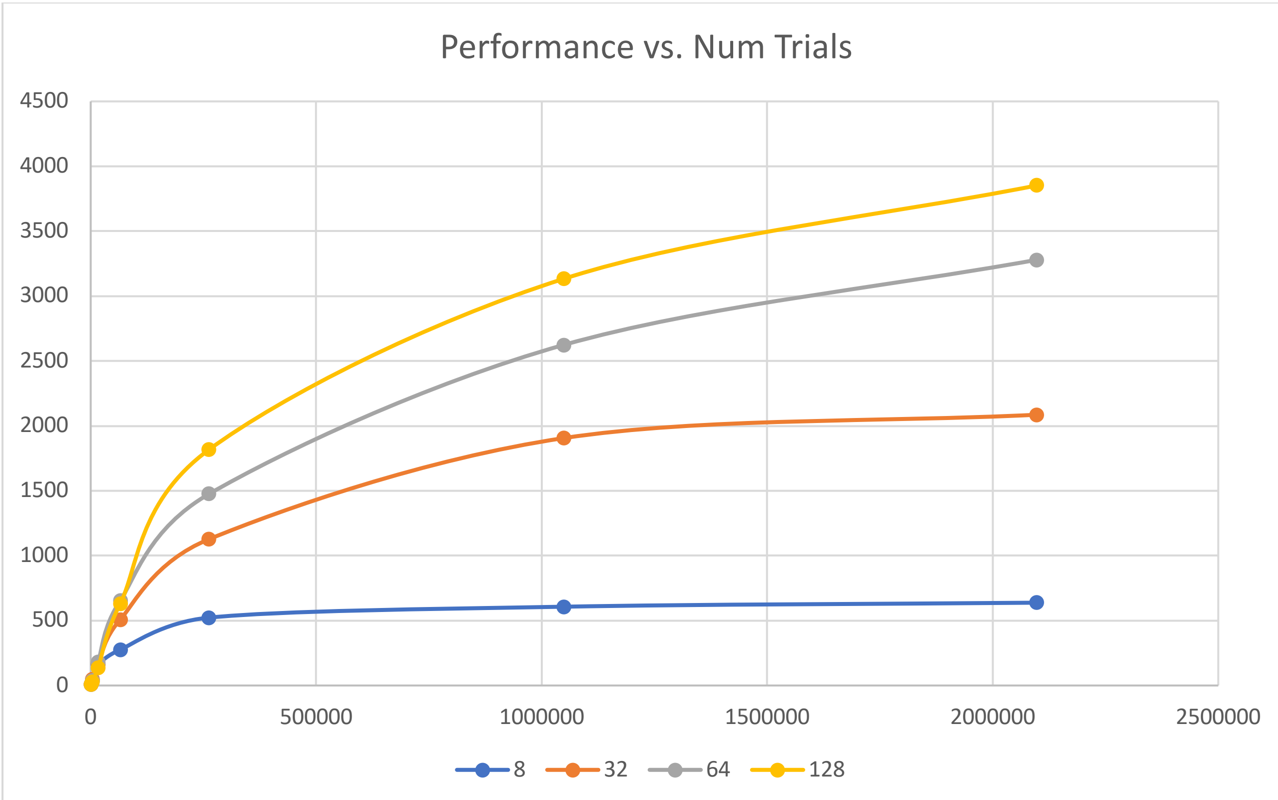
Project Title: CUDA: Monte Carlo Simulation

Name: Nirmit Patel

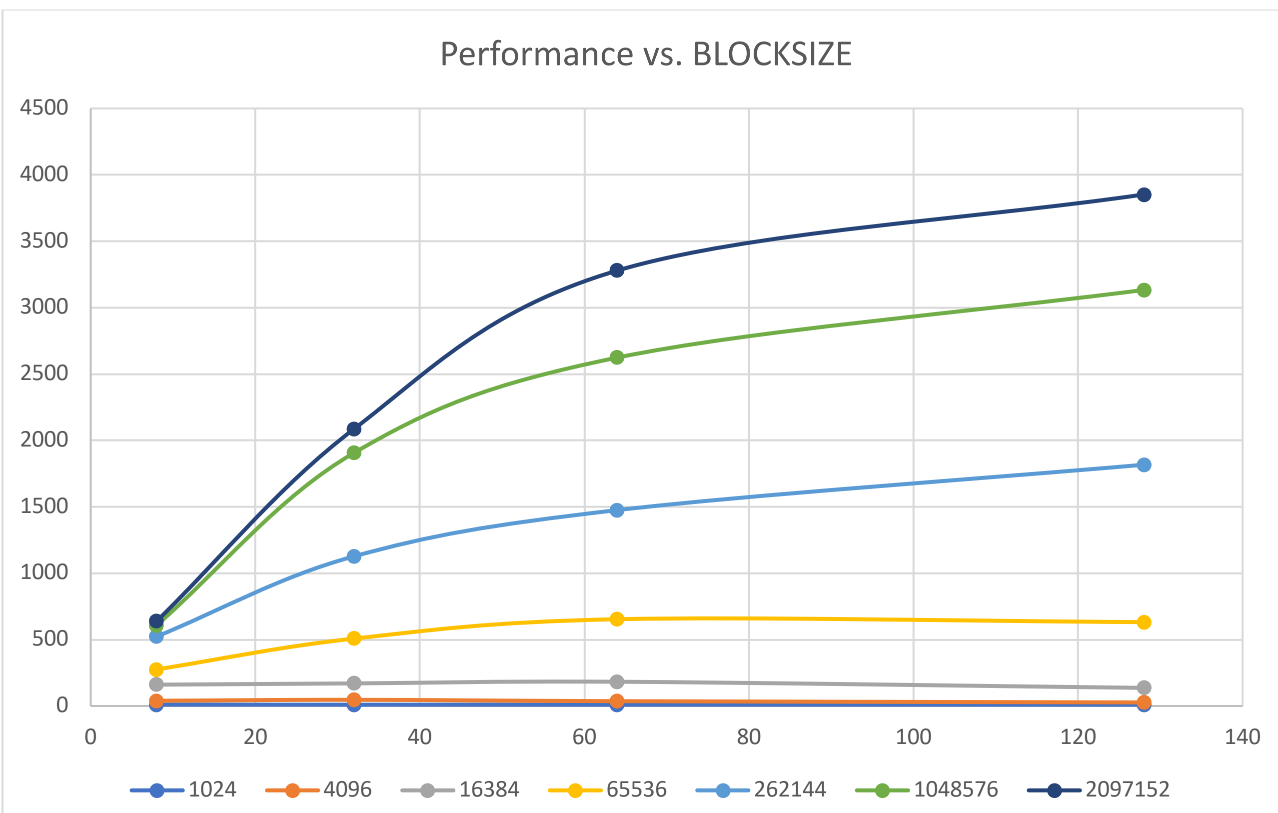
Email: patenirm@oregonstate.edu

I executed the program on the Rabbit server, and the possible probability observed was 74.74 considering the maximum Block size and Number of Trials. The program generated the following data during execution, and the graph below illustrates the results.

Number of Trials	Blocksize	Performance	Probability
1024	8	9.9751	75
1024	32	9.8462	75.59
1024	64	9.4984	73.34
1024	128	9.8371	74.41
4096	8	39.4818	74.8
4096	32	46.5624	74.78
4096	64	36.374	75.05
4096	128	26.936	74.54
16384	8	160.6023	74.58
16384	32	170.1562	75.31
16384	64	182.5312	74.72
16384	128	136.3879	74.54
65536	8	275.5651	74.64
65536	32	507.5588	74.6
65536	64	653.478	74.71
65536	128	631.514	74.69
262144	8	522.9492	74.71
262144	32	1126.9775	74.71
262144	64	1475.2386	74.65
262144	128	1816.8108	74.66
1048576	8	607.0509	74.65
1048576	32	1906.0027	74.65
1048576	64	2623.7488	74.72
1048576	128	3132.9954	74.76
2097152	8	637.9814	74.7
2097152	32	2086.0043	74.72
2097152	64	3279.0954	74.72
2097152	128	3851.8866	74.74



Performance vs. Num Trials
4500
4000
3500
3000
2500
2000
1500
1000
500
0
0
500000
1000000
1500000
2000000
2500000
8
32
64
128



Performance vs. BLOCKSIZE
4500
4000
3500
3000
2500
2000
1500
1000
500
0
0
20
40
60
80
100
120
140
1024
4096
16384
65536
262144
1048576
2097152

The graphs clearly illustrate that both the block size and the number of trials have a significant impact on performance. It is evident that smaller block sizes and fewer trials result in lower performance, while larger block sizes and higher numbers of trials lead to improved performance. As the block size increases, the code benefits from better memory locality and caching, which enhances performance. However, there is a point of diminishing returns, beyond which further increases in block size do not yield substantial performance gains. This could be attributed to factors like cache size limitations and potential contention.

Additionally, the observed performance disparities between different block sizes can be attributed to memory access patterns. Smaller block sizes (Blocksize = 8) require more frequent memory accesses, leading to increased latency and less efficient cache utilization, ultimately resulting in poorer performance. On the other hand, larger block sizes exploit memory locality, enabling more efficient caching and reduced latency, leading to enhanced performance. The relationship between block size and performance is influenced by hardware characteristics and optimizations, highlighting the importance of selecting an optimal block size to achieve the best performance for a given workload.

Compared to Project 1, the use of blocks instead of threads brings about a noticeable and significant improvement in performance. This change is characterized by a dramatic increase in performance, thanks to the higher efficiency and improved parallelization achieved through block utilization. By transitioning to a block-based approach, the program can effectively leverage GPU resources and optimize parallel computing.

The substantial performance improvement obtained by employing blocks underscores the potential of GPU parallel computing. The efficient utilization of GPU resources and parallelization capabilities afforded by blocks lead to enhanced performance, particularly in data-intensive tasks. The utilization of blocks in GPU parallel computing demonstrates the potential for leveraging GPU resources to achieve remarkable performance gains and optimize the execution of computationally intensive tasks.