Project #1

OpenMP: Monte Carlo Simulation

Name: Sahana Nagendra Hosamani
Email: nagendrs@oregonstate.edu
OSU ID: 934634588

Results of the execution of the program

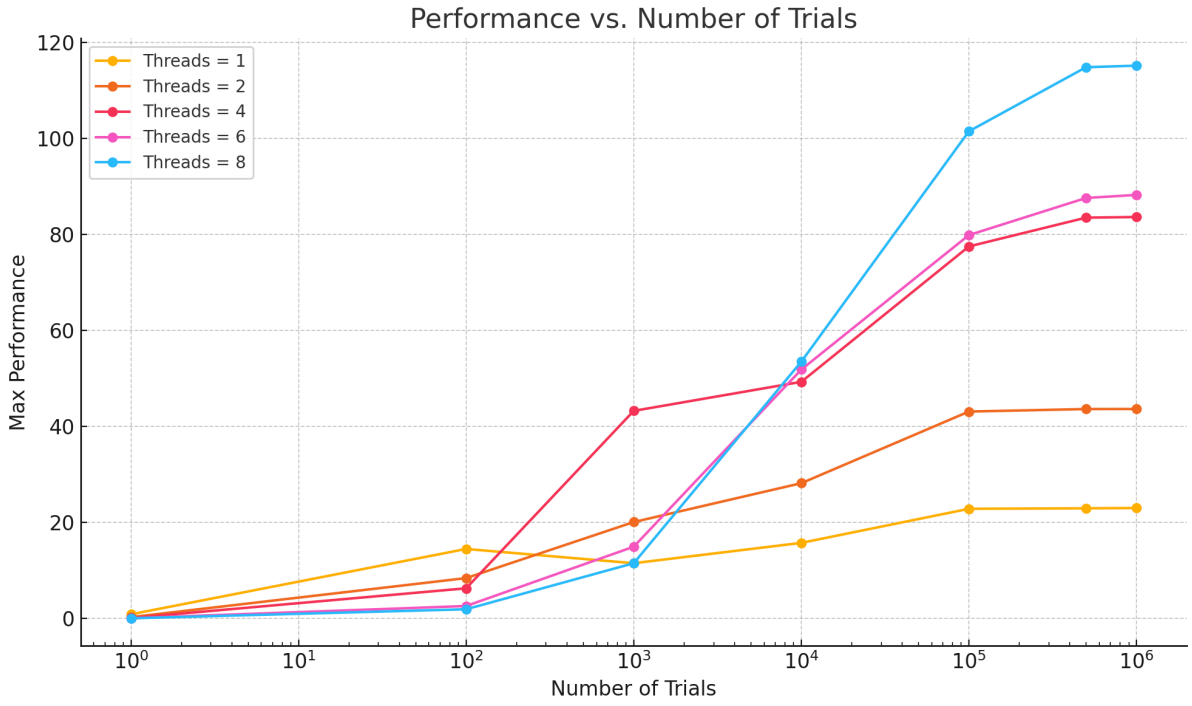
NUMT	NUMTRIALS	Probability	MaxPerformance
1	1	0	0.84
1	100	26	14.46
1	1000	23.5	11.49
1	10000	22.96	15.73
1	100000	23.21	22.83
1	500000	23.28	22.93
1	1000000	23.19	22.99
2	1	0	0.25
2	100	26	8.39
2	1000	23.5	20.07
2	10000	22.96	28.17
2	100000	23.21	43.1
2	500000	23.28	43.62
2	1000000	23.19	43.63
4	1	0	0.07
4	100	26	6.26
4	1000	23.5	43.24
4	10000	22.96	49.29
4	100000	23.21	77.51
4	500000	23.28	83.51

4	1000000	23.19	83.64
6	1	0	0.03
6	100	26	2.57
6	1000	23.5	14.93
6	10000	22.96	51.85
6	100000	23.21	79.88
6	500000	23.28	87.61
6	1000000	23.19	88.25
8	1	0	0.01
8	100	26	1.89
8	1000	23.5	11.49
8	10000	22.96	53.5
8	100000	23.21	101.51
8	500000	23.28	114.86
8	1000000	23.19	115.19

Pivot Table:Performance vs. Number of Trials

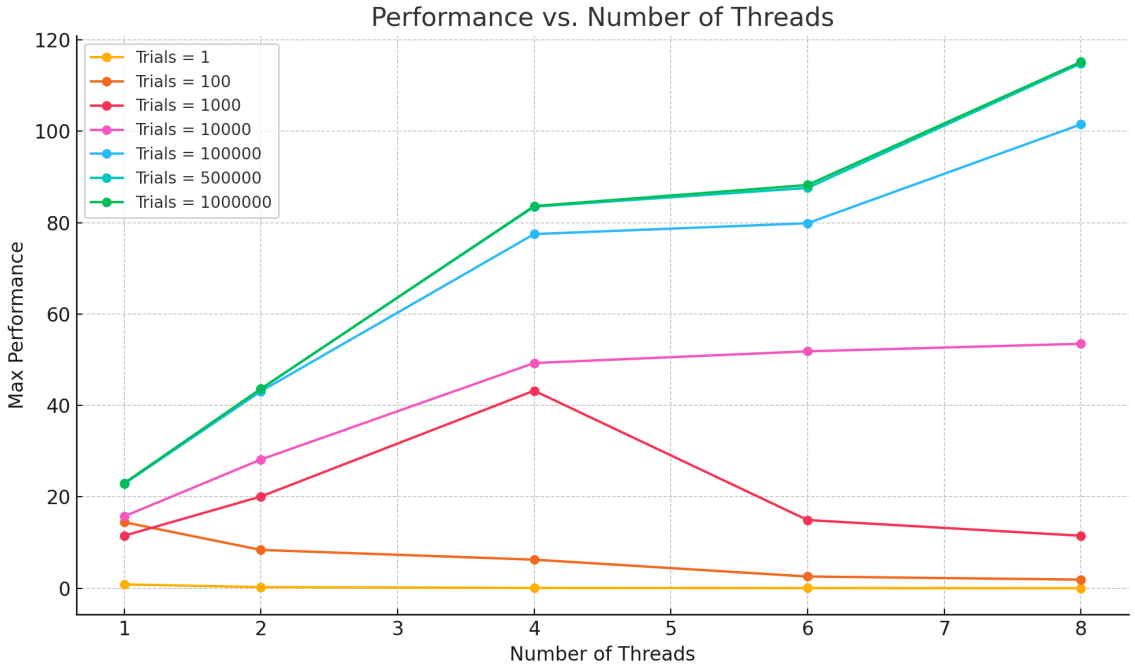
NUMTRIALS	T=1	T=2	T=4	T=6	T=8
1	0.84	0.25	0.07	0.03	0.01
100	14.46	8.39	6.26	2.57	1.89
1000	11.49	20.07	43.24	14.93	11.49
10000	15.73	28.17	49.29	51.85	53.5
100000	22.83	43.1	77.51	79.88	101.51
500000	22.93	43.62	83.51	87.61	114.86
1000000	22.99	43.63	83.64	88.25	115.19

Graph of performance vs. number of trials



This graph shows how the maximum performance (in MTrials/sec) changes as the number of Monte Carlo trials increases for different thread counts. As expected, performance improves with more trials due to better utilization of computational resources and reduced relative overhead. This effect is especially noticeable for higher thread counts, where the parallelism becomes more effective at large workloads.

Graph of performance vs. number of threads



This graph illustrates how maximum performance scales with the number of threads for different trial counts. For small numbers of trials, performance gains from adding threads are minimal due to overhead. However, as the trial count increases, performance scales significantly with more threads demonstrating effective parallelism. The curve begins to plateau at higher thread counts, suggesting limits imposed by hardware or diminishing returns from additional parallelism.

Probability

From your output:

- For NUMT=1, NUMTRIALS=1000000: Probability = 23.19%.
- For NUMT=2, NUMTRIALS=1000000: Probability = 23.19%.
- For NUMT=3, NUMTRIALS=1000000: Probability = 23.19%.
- For NUMT=4, NUMTRIALS=1000000: Probability = 23.19%.
- For NUMT=5, NUMTRIALS=1000000: Probability = 23.19%.

Since the probability is consistent across all thread counts for NUMTRIALS=1000000, we can confidently estimate:

Actual Probability: The probability of the cannonball hitting the castle is approximately **23.19%**.

Parallel Fraction Calculation

Using Amdahl's Law:

$$S = \frac{1}{1-Fp + \frac{Fp}{N}}$$

Data (NUMTRIALS=1000000):

- Performance (1 thread): 22.93 MegaTrials/sec
- Performance (6 threads): 88.25 MegaTrials/sec

Speedup(S) for N = 5:

S = 88.25/22.93 ~ 3.85

Substituting values and solving for Fp

Result: The Parallel Fraction is $Fp \sim 0.888$, meaning 88.8% of the computation is parallelizable.

Maximum Speedup Calculation

Smax = 1 / (1 - Fp)

Smax = 1 / (1 - 0.888)

Smax = 88.93

Therefore, with a parallel fraction of 88.9%, the maximum theoretical speedup you could ever achieve with any number of cores would be approximately 89x.