Juichi Lee

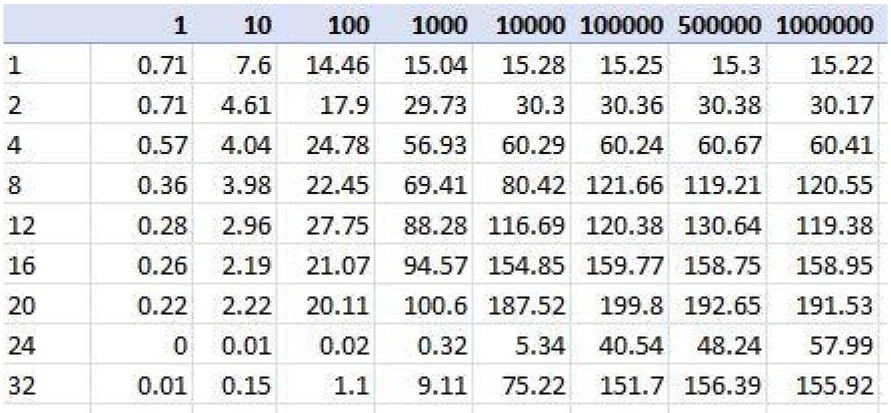
Email: leejuic@oregonstate.edu

CS 475 Parallel Programming

4/15/2022

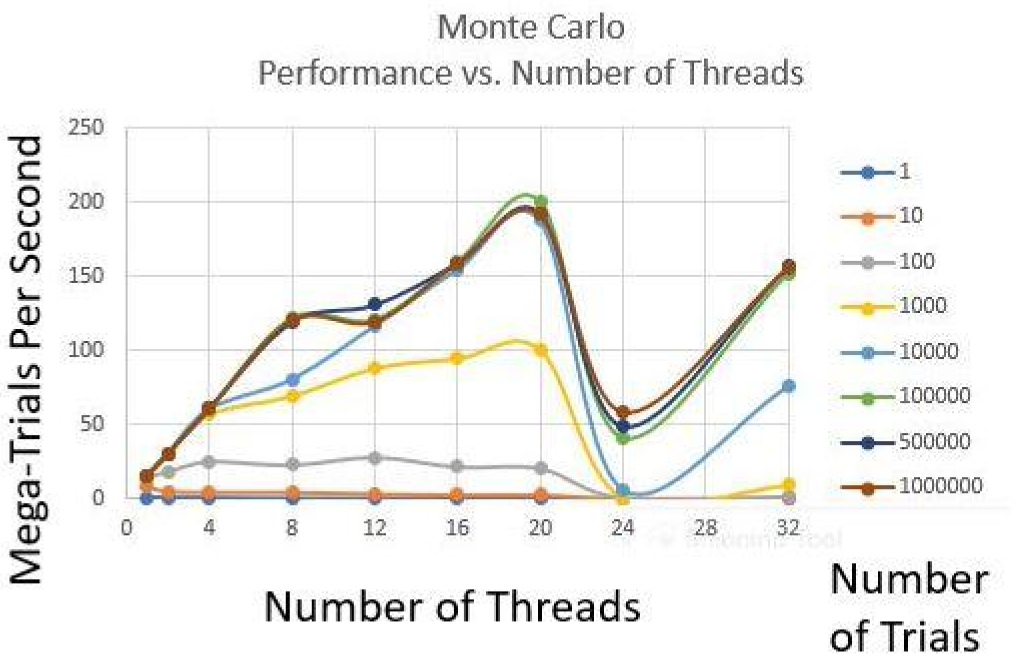
Project #1: Monte Carlo Simulation

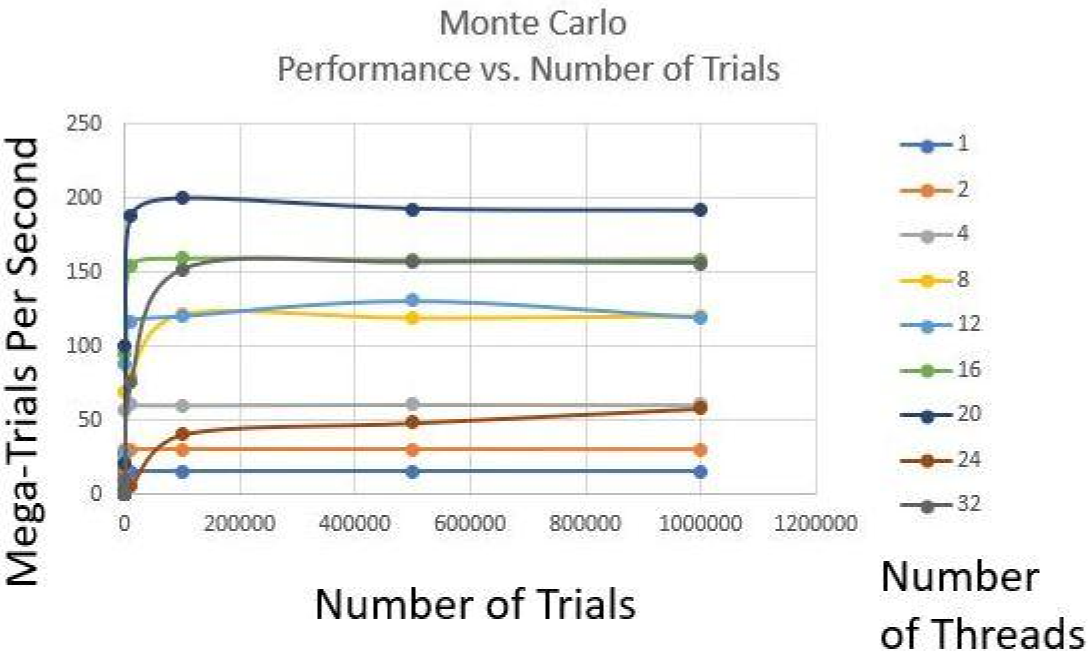
1. Table



	1	10	100	1000	10000	100000	500000	1000000
1	0.71	7.6	14.46	15.04	15.28	15.25	15.3	15.22
2	0.71	4.61	17.9	29.73	30.3	30.36	30.38	30.17
4	0.57	4.04	24.78	56.93	60.29	60.24	60.67	60.41
8	0.36	3.98	22.45	69.41	80.42	121.66	119.21	120.55
12	0.28	2.96	27.75	88.28	116.69	120.38	130.64	119.38
16	0.26	2.19	21.07	94.57	154.85	159.77	158.75	158.95
20	0.22	2.22	20.11	100.6	187.52	199.8	192.65	191.53
24	0	0.01	0.02	0.32	5.34	40.54	48.24	57.99
32	0.01	0.15	1.1	9.11	75.22	151.7	156.39	155.92

2. Graphs





3. Actual Probability
    - The probability of hitting the truck, according to a run with the maximum number of trials (1000000) and 32 cores, is roughly 29%

4. Parallel Fraction Fp for this Computation
    - S (Speedup from 1 - 32 threads) = 155.92 / 15.22 = 10.24
    - Fp = (n/n-1) * (1 - (1/S) = (32./31.) * (1. - (1./10.24)) = ~0.93
5. Commentary
    - For the Monte Carlo simulation, I included the ranges for the random numbers, included the relevant equations for calculating snowball x and truck x at a given time t, included a check for determining if the snowball hit the truck using the prior information, wrote a shell script for inputting a wide array parameters programmatically, exported the data to Excel, and graphed the Performance vs. Number of Trials and the Performance vs. Number of Threads. I found the overall assignment to be straightforward, save for the creating the graphs in Excel since I seldom use Excel for graphing. Interestingly enough, in the Performance vs. Number of Threads

graph, the Mega-Trials per Second dropped significantly upon reaching 24 threads. I am not sure why this happened but my guess is that the flip server only has so many threads it can provide and so going over that number causes the performance to suffer.