Amy Stockinger (stockina@oregonstate.edu)
CS475 Project01 – Monte Carlo Simulation
Due: April 17, 2019

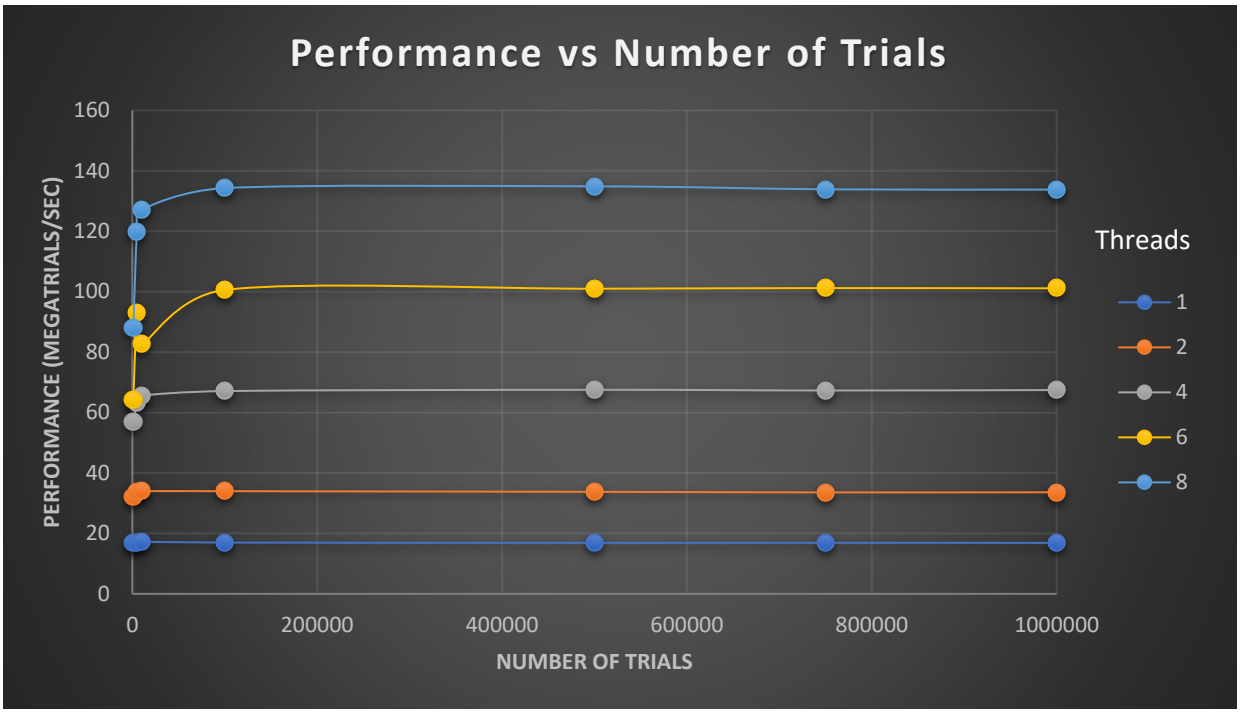
In this program, we are trying to calculate the probability that a laser aimed at 45deg in the origin will hit a variable size/location circle and bounce downward to hit an infinite horizontal plane. We are given ranges for the x and y coordinates of the center of the circle and for the circle radius. There are four potential cases: A) the laser misses the circle completely, B) the circle is so large that is engulfs the laser, C) the laser hits the circle, but reflects upward and does not hit the plane, and D) the laser hits the circle, reflects downward and hits the plane.

The program generates random numbers for our center and radius variables and places them in arrays that are the size of defined NUMTRIALS. Some calculations are performed on the numbers to determine which scenario happens (A-D), and a counter variable only increments if scenario D is calculated given the numbers in the arrays.

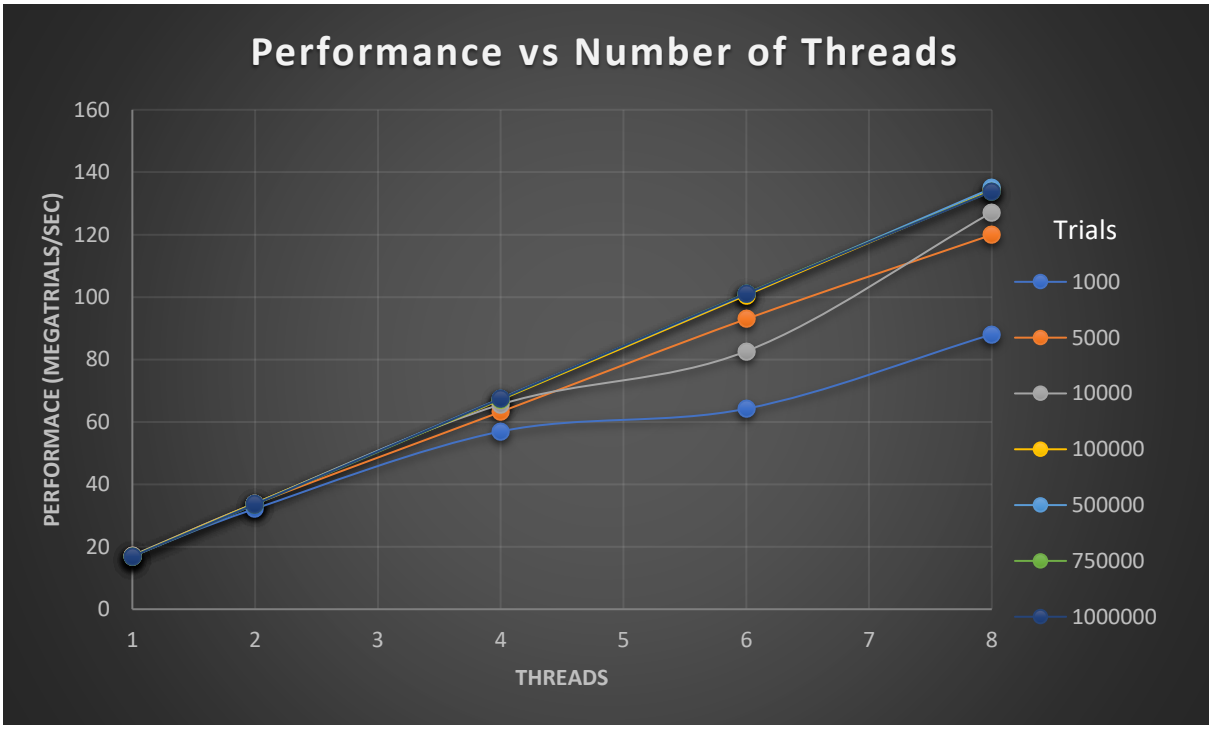
For this program, I tested on flip2 while the uptimes were all around 3.00. The program was compiled using g++ and without -O3. I took the preset number of trials (1,000,000) and made that the high end of my testing range. I used 1000, 5000, 10,000, 100,000, 500,000, 750,000 and 1,000,000 as NUMTRIALS. For threads, I used 1, 2, 4, 6 or 8. Below are the table and graphs of performance vs number of trials, and performance vs number of threads.

Calculated Performance for each run

Threads \ Number of Trials	1000	5000	10000	100000	500000	750000	1000000
1	17.1	17.02	17.24	16.97	16.88	16.9	16.87
2	32.2	33.75	33.98	33.98	33.77	33.57	33.62
4	56.96	63.23	65.6	67.12	67.55	67.3	67.45
6	64.23	92.96	82.65	100.59	100.98	101.19	101.11
8	88.03	119.9	127.13	134.29	134.87	133.89	133.8



From this graph, we can see the 'jumpy' part for the smaller datasets (>100,000 trials) and larger thread numbers (>4 threads), but it evens out very quickly—well before the higher end of the trials range. The differences in performance for the smaller data sets varies more as the number of threads increases. However, within each trial run, the performance is always better with more threads.



The above graph shows a clear performance advantage to using more threads. However, the performance ranges for the lower range of trials become more spread out as the number of threads increases. The performance readings for >100,000 trials show steady lines that are basically all on top of each other.

Parallel Fraction
Below are parallel fraction calculations for each number of threads. I used the 1,000,000 trials performance numbers in all calculations.

2 threads: $S = P_2/P_1 = 33.62/16.87 = 1.99$
$F_p = (2 / 1) * (1 – (1 / 1.99)) = 0.99$

4 threads: $S = P_4/P_1 = 67.45/16.87 = 4.00$
$F_p = (4 / 3) * (1 – (1 / 4)) = 1.00$

6 threads: $S = P_6/P_1 = 101.11/16.87 = 5.99$
$F_p = (6 / 5) * (1 – (1 / 5.99)) = 1.00$

8 threads: $S = P_8/P_1 = 133.80/16.87 = 7.93$
$F_p = (8 / 7) * (1 – (1 / 7.93)) = 1.00$

F_p (avg) = (0.99 + 1 + 1 + 1)/4 = 3.99/4 = 1.00

With the exception of the 4 threads calculation, F_p all numbers were just under 1.00 (e.g. 0.9986) and rounded up. The average also came out to 0.9975, which I rounded to 1.0 as well. Since only the main loop is accounted for in the execution times and performance, that part of the program is very parallelizable.

Probability
Below is a table showing the probabilities computed during each run:

Calculated Probability for each run

Threads \ Number of Trials	1000	5000	10000	100000	500000	750000	1000000
1	18.1	19.12	18.71	18.94	19.11	19.03	19.04
2	20.2	19.2	18.84	18.98	18.96	19.02	19.05
4	19.3	18.6	19.05	19.14	19	19.01	19.13
6	18	18.68	19.37	19.14	19.12	19.09	19.12
8	19.4	19.68	18.16	18.92	19.05	19.05	19.01

The probability range from all data is 18.0%-20.2%. These numbers will vary slightly depending on which random data was generated for the radius and center coordinates. The numbers are more agreeable for the larger number of trials. Interestingly, the closest of all is the 750,000 trial run with 0.8% being the largest difference in that set. The 1,000,000 trial run is the largest data set, and that shows a 0.11% difference. So, the probability that a laser at a 45deg angle will hit a spherical object with a radius in range 0.5 to 2.0, and center coordinates with x in range -1.0 to 1.0 and y in range 0.0 to 2.0, is **about 19±0.1%.**