Monte Carlo Simulation for Pi Estimation

Objective

The objective of this assignment is to estimate the value of Pi (π) using a Monte Carlo simulation technique implemented in C with OpenMP parallelization. This simulation relies on random sampling to approximate the value of Pi by simulating the throwing of darts at a dartboard and analyzing the proportion that land inside a circle.

Implementation Details

Monte Carlo Simulation

1. **Mathematical Basis:**

   The area of a circle of radius r is πr^2, and the area of the square that encompasses this circle is 4r^2. The ratio of these areas is πr^2 / 4r^2 = π / 4. Thus, the approximation of Pi can be obtained from: π ≈ 4 × (H / N) where H is the number of darts that land inside the circle, and N is the total number of darts thrown.

2. **Random Sampling:**

   Each dart is represented by a random pair of coordinates (x, y) where x and y are between -1 and 1. A dart lands inside the circle if (x^2 + y^2 ≤ 1) Up to a billion darts are simulated to achieve a high precision.

Code Structure

1. **Monte Carlo Simulation Code:**
   - Initialize variables and set up OpenMP environment.
   - Generate random coordinates for each dart.
   - Count the number of hits inside the circle.
   - Compute Pi based on the ratio of hits to total darts.

Results and Analysis

Error Plot

The error in the approximation of Pi is computed as:

Error = |π - Estimated Pi|

Where the exact value of Pi is obtained using acos(-1). The error is plotted against the amount of darts on a logarithmic scale for both axes.

Dart Count	Average Error
1000	0.04287034
10000	0.01735932
100000	0.002339115
1000000	0.001114769
10000000	0.000221882
100000000	0.000124268
1000000000	1.29121E-05

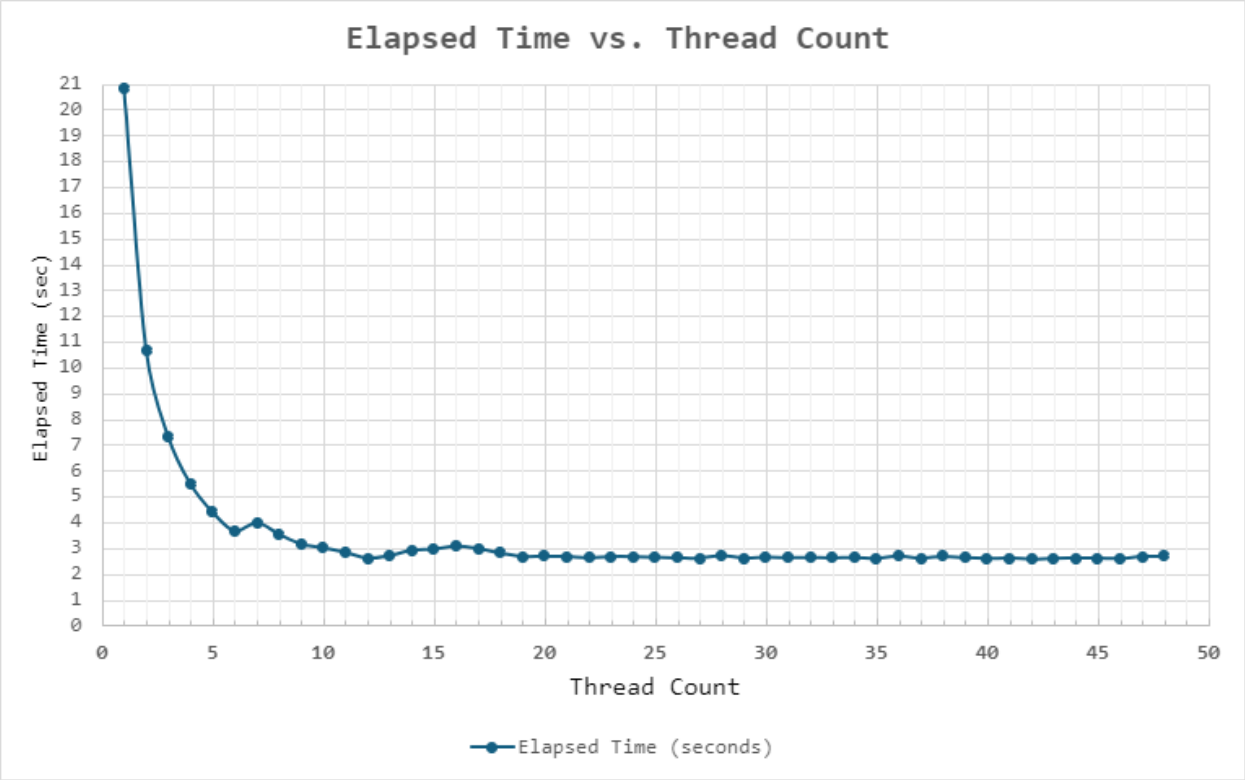
Performance Analysis

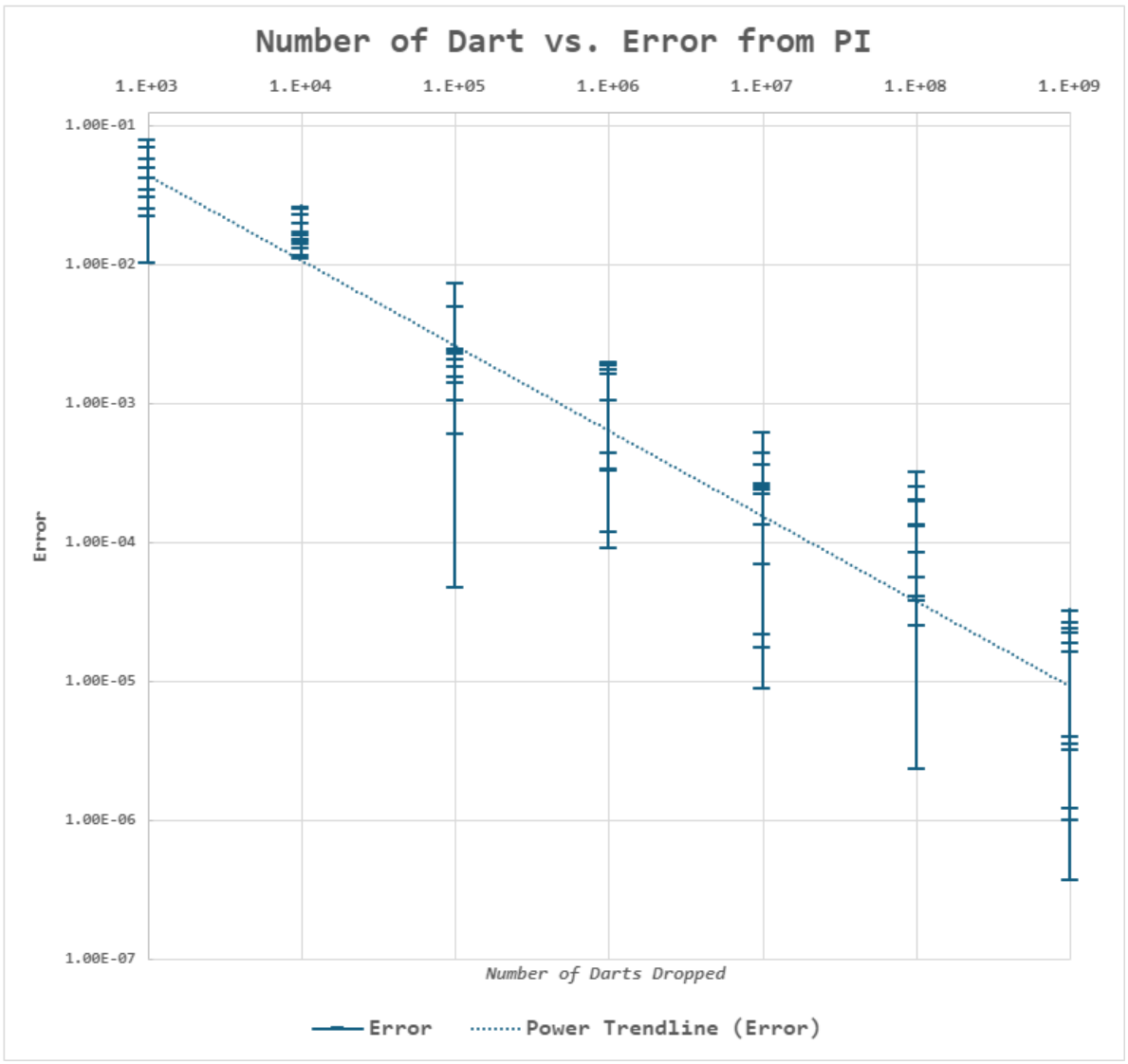
To analyze the performance of the parallel implementation, a strong scaling plot is created by measuring the elapsed time for different numbers of threads.

Key Observations:

- The simulation performance improved significantly with an increasing number of threads up to around 12 threads, where the elapsed time decreased from 20.7934 seconds (1 thread) to 2.6056 seconds (12 threads). After 12 threads, the elapsed time showed variability but did not exhibit a consistent improvement.

Graphs


Elapsed Time vs. Thread Count
Elapsed Time (sec)
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
0
5
10
15
20
25
30
35
40
45
50
Thread Count
Elapsed Time (seconds)



Conclusion

The Monte Carlo simulation successfully estimated the value of Pi with increasing accuracy as the number of darts increases. The use of OpenMP for parallelization improved performance. The provided data and plots demonstrate the effectiveness of the simulation and highlight key performance characteristics.