CSE435 FALL 2024 Homework 2

This homework is assigned to each student who is registered to CSE435. Answers to the questions should be student's own work. No group work is allowed. Students may be asked to explain/present their answers if needed.

Deadline and Submission

- Submit until midnight(23:55) of December 18, 2024
- Submit in digital format. Create a zip package which includes source files, outputs of the tests and the report.
- Late Submission:
  - 1 day late -15 pts
  - 2 days late -30 pts
  - 3 days late -45 pts
  - 4 or more days late not accepted

Questions

Show steps of derivations and state any assumptions you made.

Q1 [5 pts] Give information about the computing environment you use

- Operating system
- CPU information (#of cores, cache size etc...)
- Memory information (size, speed etc...)
- Compiler information.

Q2 [10 pts] Suppose we toss darts randomly at a square dartboard, whose bullseye is at the origin, and whose sides are 2 feet in length. Suppose also that there's a circle inscribed in the square dartboard. The radius of the circle is 1 foot, and it's area is π square feet. If the points that are hit by the darts are uniformly distributed (and we always hit the square), then the number of darts that hit inside the circle should approximately satisfy the equation

$$\frac{number\ in\ circle}{total\ number\ of\ tosses} = \frac{\pi}{4}$$

since the ratio of the area of the circle to the area of the square is $\pi/4$. We can use this formula to estimate the value of π with a random number generator. This is called a "Monte Carlo" method, since it uses randomness (the dart tosses).

Write a single threaded C program that uses a Monte Carlo method to estimate π.

```
number_in_circle = 0;
for ( toss = 0; toss < number_of_tosses; toss++) {
    x = random double between -1 and 1;
    y = random double between -1 and 1;
    distance_squared = x*x + y*y;
    if ( distance_squared <= 1)
        number_in_circle++;
}
```

Q3 [20 pts] Write an MPI program that uses a Monte Carlo method to estimate π. `Process 0` should read in the total number of tosses and broadcast it to the other processes. Use `MPI_Reduce` to find the global sum of `number_in_circle`, and have `process 0` print the result. You may want to use `long long ints` for the number of hits in the circle and the number of tosses, since both may have to be very large to get a reasonable estimate of π.

Generate results of execution for different number of processes.

Q4 [20 pts] Repeat Q3 with `pthreads`. Generate results of execution for different number of threads.

Report [45 pts] For Q2, Q3 and Q4, Try different number of repetitions, time the execution at each case, plot a graph for different runs. show, how accurately you can represent π for different ranges of loops. (you may need to repeat an experiment for a fixed loop range, in order to comment about the accuracy of the calculation).

You don't need to use any graph libs to generate the graph in your program. You can print the parameters to a file and manually generate graphs on a spreadsheet.

Compare performances of Q2, Q3 and Q4.