Assignment 3: Buffon's Rusty Needle

Dr. William Kreahling

April 6, 2021

1 Overview

For this assignment you will write a multi-threaded application that estimates the value of PI by Dropping needles on a floor marked with evenly spaced lines. Each thread will calculate multiple "drops" of a needle. You may not use the contant PI (i.e., std::f64::consts::PI in your calculations).

One way to calculate your estimate is: $\frac{2N(hits + misses)}{L(hits)}$

2 Instructions

- The main thread will print the correct answer AFTER all threads have finished.
- Create a 'Class' Experiment in lib.rs. (You may find the Copy and Clone traits useful)
- `Experiment` will prompt the user for data, in this specific order for:
  1. the length of the needle(N)
  2. and the distance between lines (L) (for this experiment the size of the needle should be less than the distance between lines).
  3. for the total number of needles to drop in the entire experiment
  4. the number of threads (each thread should drop a roughly equal amount of needles)
- All threads must be started and run concurrently.
- The main thread, should start accessing data from threads as soon as the data becomes available.
- Each thread has a unique id, this may be useful for debugging
- Each thread instance should know (at least): the distance between lines, the length of the needle and the number of needles it to drop.
- You must use message passing to transport data, you may **not** use shared memory.
- Style and code correctness count.
- Watch method/function size
- No static methods or variables (except main), without good reason.
- All errors should be handled in an appropriate manner.
- Correct input is not guaranteed – you should print useful error messages!
- Helper methods/functions are allowed
- You must use Rust for this assignment.
- use `cargo run --release` for testing

3 Examples:

I will demonstrate a simplified version in class.

4 Notes on Collaboration

You may work in teams of up to two on this assignment. Note that all members of a team will receive the same grade on the assignment.

5 Hand-In Instructions

This assignment is due by 11:59 PM on Wednesday April 21st. A **single** version version of the assignment is due from each team. Submit all source files associated with the program as well as the Makefile. To submit your files, use the *handin* command on agora. Handin works as follows:

```
handin.<course#>.<section#>  <assignment#>  <files>
```

Therefore, to submit this assignment, you must use the following command (assuming each of the files is in your current working directory):

```
handin.370.1  2  *.java
```