CS311 : Computer Architecture Lab

Assignment 0 - Border Crossing Problem - Report

S.Surzith Raj Pandi(200010049), Sourabh Bhosale(200010004)

August 9, 2022

1 Abstract

In this assignment, we simulate a scenario where an infiltrator from Attacking Country (AC) is trying to cross a border with randomly switched motion sensors. We try to show the relation between width of the border and probability of sensor being switched, with time taken to cross the border.

2 Assumptions & Trivial Parameters

In accordance with the description of the challenge, the length has to be infinitely long. So, the length we have chosen is as follows:

$$L = 1000$$

From the challenge description, the sensor is switched **ON**, if we receive **Heads** in the coin toss, which in turn depends upon the array/spectrum of probabilities we are considering. The following Set P represents the set containing the probabilities of sensors being ON we have considered:

$$P = \{0.2, 0.3, 0.4, 0.5, 0.6, 0.8\}$$

The time taken by the infiltrator to reach DC from AC will increase if width is increased. The following Set W represents the set of width we considered for pairing with $P's$ elements to understand and comprehend on different time results.

$$W = \{3, 6, 9, 12, 15, 18, 21, 24, 27, 30\}$$

The infiltrator only moves ahead from AC to DC and no step backwards it taken. With a time span of every 10 secs, a coin is tossed and based on that upcoming step from the neighbouring 8 is chosen.

3 Approach & Code

The approach was simple. We have to just see the 3 sensors ahead of the infiltrator where he can move and the current sensor on which the infiltrator is currently standing. If the sensor at the coordinates of the infiltrator is activated, then he would wait for the next cycle until that sensor is deactivated. If the sensor on which the infiltrator is standing is off, then we just check if any of the 3 forward sensors are off and we just move forward to any one of them.

We created multiple classes to assist with the simulation. Border class has two members, length and width. Length is fixed at 1000 units whereas we vary width, W was per requirement. Sensor class has a method which takes a parameter, the probability P, and returns a boolean True if the random number generated, X is less than P, else it returns False. The infiltrator class has a method which takes care of the motion of the infiltrator. If any 3 of the cells in front of him are empty, we moves forward, else we stays at his current cell. The method returns the sum of the width and the times he had to wait to move forward. Time class inside Main class takes care of the time elapsed in this simulated world. Main class finally calculates the time taken for the infiltrator to cross the border and outputs it on the System Output.

For the above given sets of P & W, we tried all cross combinations i.e. for a given P, we tried all $W's$ with it and found out all different times.

4 Files used Execution

We have created following files of codes which are as follows:

- Border.java
- Infiltrator.java
- Main.java
- Sensor.java
- graph.py

Each file is embedded with a separate class which finally top inherits to *Main.java* file. The outputs generated are inside ***output.txt*** file in the format probability, border-width, time. And *graph.png* was generated by ***graph.py***.

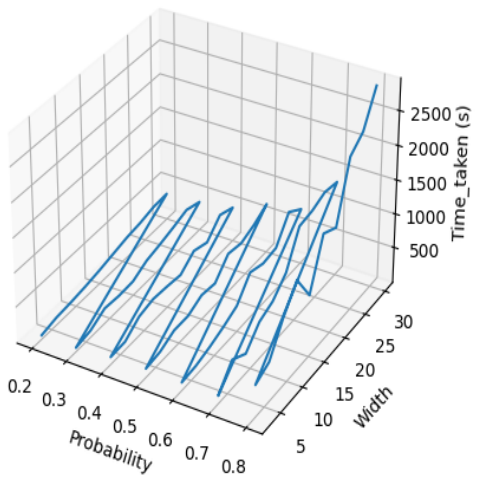

Figure 1: Variation of Probability, Width and Average Time

5 Graph

Figure 1 shows the graph and where we have used the parameters as follows:

- X-Axis: Border-Width
- Y-Axis: Probability
- Z-Axis: Time

The graph just depicts and proves the deductions which we made about variations of time with change in probabilities and widths.

6 Conclusions

We can conclude that, when probability of the sensor being switched on be- comes very large (when it is very likely that the sensor is on!), the infiltrator has to wait for a long time to cross even a single unit across the width. Whereas, for smaller probabilities, it is very unlikely that there is no path for the infiltrator to move forward. Hence, it varies linearly with width.