Lawrence Oks
CAP6135
March 27, 2022

Programming Assignment 3: Internet Worm Propagation Simulation

1. **Design**

   In this programming assignment we simulate internet worm propagation in two different ways: randomly and sequentially. Of an IP address space containing 100,000 nodes, 1,000 of which are vulnerable, a single node begins as infected (specifically IP address 10050). When all 1,000 vulnerable nodes are compromised, the simulation is stopped and results are recorded.

   a. **Random**

      For the random simulation, we begin by initializing the IP address space with 100,000 nodes, and assign the correct clusters to be vulnerable. For my implementation, each node has one of the following values:

      - t = -1: the node is immune to infection
      - t = 0: the node is vulnerable but not yet infected
      - t = 1-10: the node has been infected for t ticks

      We then initialize node 10050 to 10, as it is ready to infect other nodes immediately. We also initialize a counter for the num_infected to 1, a counter for the number of ticks, and an array of values that records the num_infected for each tick.

      Inside of a while loop that stops when num_infected=1000, we generate 5 random numbers for each node that already has a value of 10. After incrementing each of the other nodes between 1-9 (so as not to interfere with each other), we infect any of the random IPs selected that are vulnerable.

   b. **Sequential**

This simulation has the same initialized variables, along with one extra array called scanType which indicates whether each of the 100,000 nodes are

- -1: immune or not yet chosen
- 0: sequential (80% chance)
- 1: random (20% chance)

We pick a scanType for node 10050, then proceed to the while loop. There, we loop through each of the vulnerable clusters. Any infected IPs whose value is already 10, we create a list of 5 IPs for them to scan, either sequentially or randomly depending on their scanType. If their value is instead between 1-9, we increment their value to simulate the next tick. Then we iterate through the accumulated list of IPs to scan, and if any of them are vulnerable and not yet infected, we infect them, along with choosing a scanType for them. Once the number of infected nodes = 1000, we end the loop and plot the results.

2. Results and Analysis

We perform 3 simulations for both random and sequential scanning methods. To display the results, we output a curve for each run that represents the number of infected computers over each time step (tick *t*)

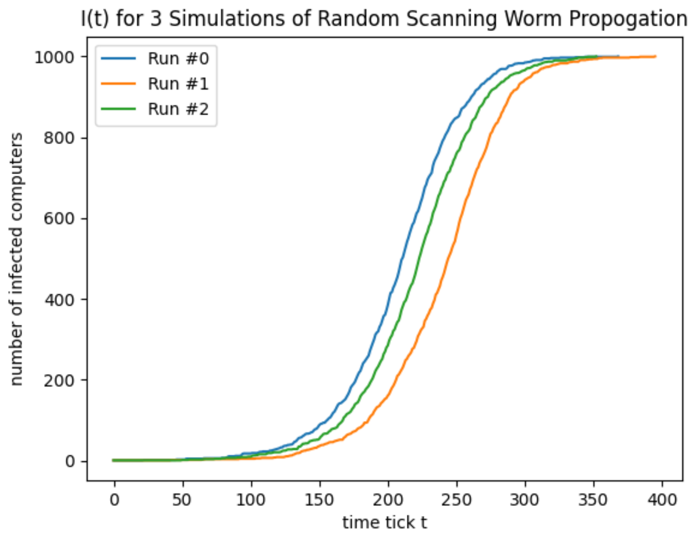


Figure 1: random scanning results over 3 runs

The results for random scanning worm propagation are tight and consistently in the 300-400 ticks range. They generate a sigmoid function as expected, that rapidly increases over time until a ceiling is hit where it is difficult to randomly find the few remaining vulnerable nodes.

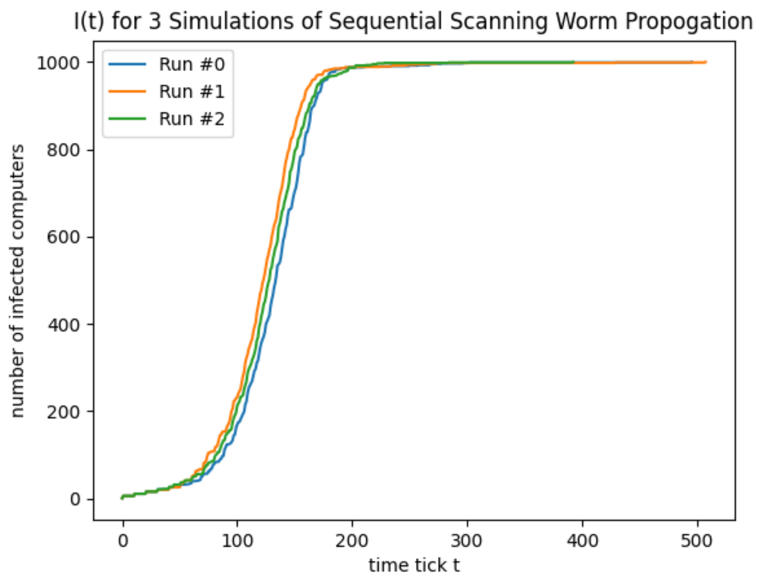


Figure 2: sequential scanning results over 3 runs

The results for the sequential scanning method are significantly less consistent. Over a myriad of trials, runs were found to take mostly between the 300-600 tick range, however there were outliers as small as 280 ticks and as large as 1200 ticks. We notice extremely quick growth at the beginning, but slow convergence to 1000 infected nodes. It can be inferred that perhaps nodes at the front of vulnerable clusters may be difficult to find for sequential scans, as they only scan forward but not backward.

Over many sequential trials, I can report that there is a correlation between the proportion of sequential scan types and the number of ticks. Trials in which the random choice led to slightly more sequential nodes tended to end in fewer ticks.

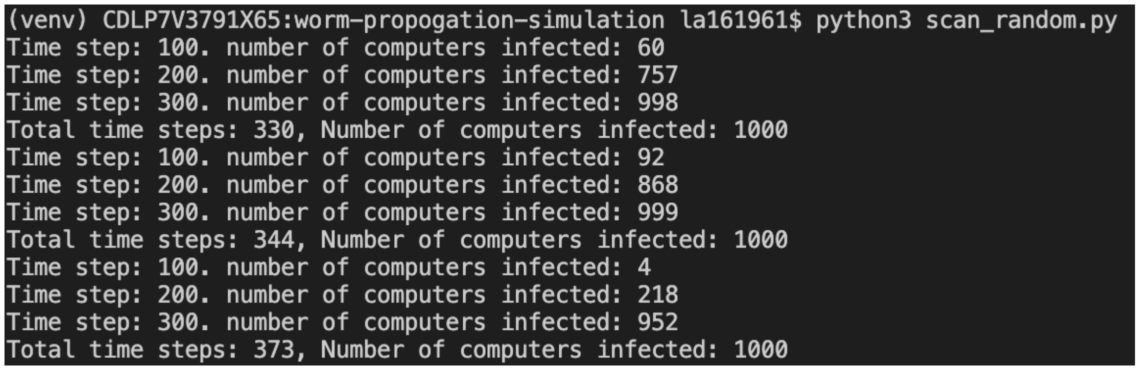
Simulation Method of Worm Propagation	Number of ticks to infect N=1000 nodes		
	Run 1	Run 2	Run 3
Random	368	395	352
Sequential	495	507	392

Table 1: Numerical results for Figures 1&2

Table 1 represents the results of Figures 1 and 2 numerically. We can conclude that neither method of propagation is quicker, however the random method has a much more predictable range of ticks to finish infecting all 1000 nodes.

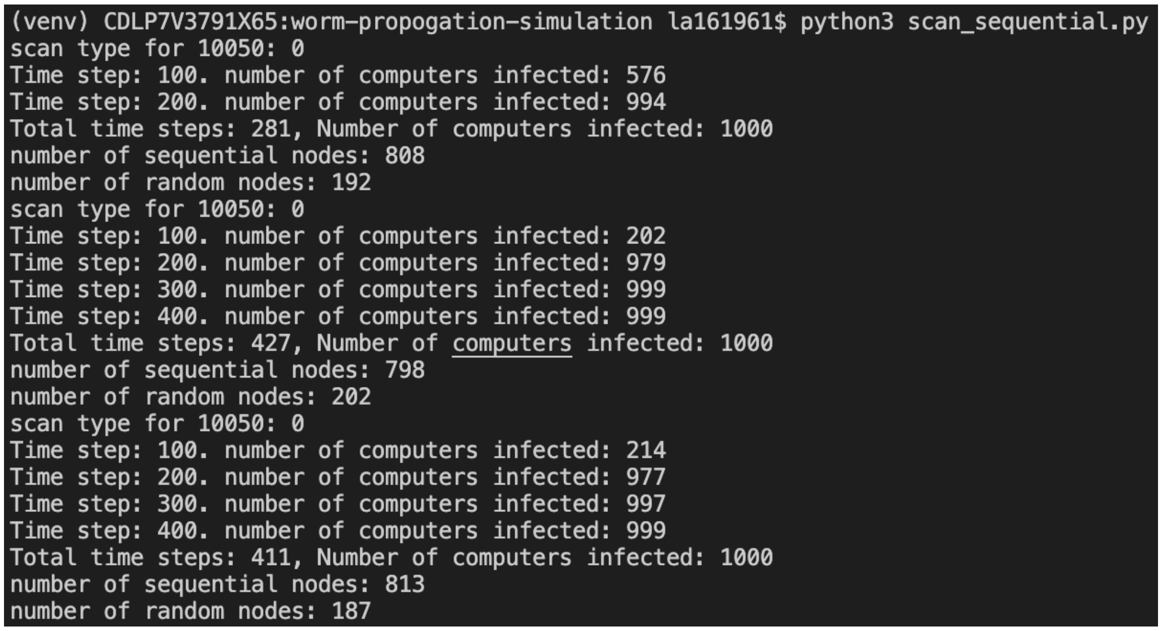
3. **Output**

The following screenshots show outputs for trials of both the random and sequential scanners, respectively:



```
(venv) CDLP7V3791X65:worm-propogation-simulation la161961$ python3 scan_random.py
Time step: 100. number of computers infected: 60
Time step: 200. number of computers infected: 757
Time step: 300. number of computers infected: 998
Total time steps: 330, Number of computers infected: 1000
Time step: 100. number of computers infected: 92
Time step: 200. number of computers infected: 868
Time step: 300. number of computers infected: 999
Total time steps: 344, Number of computers infected: 1000
Time step: 100. number of computers infected: 4
Time step: 200. number of computers infected: 218
Time step: 300. number of computers infected: 952
Total time steps: 373, Number of computers infected: 1000
```

Figure 3: output of random scanning



```
(venv) CDLP7V3791X65:worm-propogation-simulation la161961$ python3 scan_sequential.py
scan type for 10050: 0
Time step: 100. number of computers infected: 576
Time step: 200. number of computers infected: 994
Total time steps: 281, Number of computers infected: 1000
number of sequential nodes: 808
number of random nodes: 192
scan type for 10050: 0
Time step: 100. number of computers infected: 202
Time step: 200. number of computers infected: 979
Time step: 300. number of computers infected: 999
Time step: 400. number of computers infected: 999
Total time steps: 427, Number of computers infected: 1000
number of sequential nodes: 798
number of random nodes: 202
scan type for 10050: 0
Time step: 100. number of computers infected: 214
Time step: 200. number of computers infected: 977
Time step: 300. number of computers infected: 997
Time step: 400. number of computers infected: 999
Total time steps: 411, Number of computers infected: 1000
number of sequential nodes: 813
number of random nodes: 187
```

Figure 4: output of sequential scanning

4. Instructions to Run

- ☐ $ unzip pa3.zip
- ☐ $ cd pa3/
- ☐ $ pip install matplotlib
- ☐ $ python3 scan_random.py
- ☐ $ python3 scan_sequential.py