Agent Technology Practical - Lab Assignment 3

Viral network

Name 1: Tim Chandler
Student number 1: s3173593
Name 2: Darren Rawlings
Student number 2: s3612309
Lab session group: 8

February 18, 2021

Question 1

a) We ran the model 10 times, for 3 of those times the model stopped nearly instantly with no infected nodes, 1 or 2 resistant nodes and the remaining node susceptible. In the other 7 times the virus managed to catch hold and was propagated through a sizeable proportion of the network infecting nodes that, infect other connected nodes until they become resistant. Once the simulation has finished pockets of susceptible nodes remain surrounded by resistant nodes.

b) In these cases the infection becomes trapped until the infected nodes become resistant. This happens as nodes can only pass the infection to connected nodes. As the nodes only connect to a small number of nearby nodes, it becomes possible for them to be 'trapped' by a small number of resistant nodes.

c) The apparent mirroring occurs because all but one node start out as susceptible (approx. 100%) and just the one node (approx. 0%). As a node becomes infected it causes both an increase in the infected rate and a decrease in the susceptible rate, hence the apparent mirroring. The reason they do not mirror is the third state resistant. Only infected machines become resistant hence it takes from the pool of infected machines, and there for distorts the mirroring.

d) Setting the random seed to 6 results in the same setup every time. Setting it to a different value will give a different setup to the seed of 6, but still the same for each setup with the new value. The change in the setup does not correlate linearly to the change in seed. A change of 1 in the seed could result in a vastly different setup. This could be useful for demonstrations, where you want to be sure of what the outcome will look like, for sharing specific examples or for any other situation requiring consistency between runs.

Question 2

a) Added line 8 `has-been-infected?` this is a binary variable to store if the specific node has at some point been infected. Added line 31 added `set has-been-infected? false`, this sets the initial value of this variable to false for all nodes. Added line 73 `set has-been-infected? true` this sets the value to true if a nodes has become infected, at no point will this every change back to false.

b) Line 1; `globals [ unique-infections ]` was added to store the count of total unique infections in.
Line 13; `set unique-infections 0` was added to initialise the counter to 0 during the setup process.
Lines 70 - 72 `if has-been-infected? = false [`
`set unique-infections unique-infections + 1`
`]` were added to increment the infection counter the first time a node became infected.

c) These changes were made in the GUI, we added a Monitor for the `unique-infections` variable and edited the existing plot to have an additional plot pen with the update command `plot (unique-infections * 100 / count turtles)`.

Question 3

a) Below the variables and how they may be influenced are listed.

Number of nodes	:	This could not be realistically influenced as in a real world situation it would represent the total population.
Average node degree	:	This can be controlled in part by social isolation and reduction in contact with other people. Some aspects of it will be harder to control for, as people will still need to buy food, either at supermarkets or via delivery, increasing the potential transmission vectors.
Initial outbreak size	:	This could not easily be controlled, as it is likely that the initial outbreak would not be immediately detected.
Virus spread chance	:	This can be controlled, by taking preventative measures such as wearing masks and social distancing (and potentially other measures dependent on the specific virus)
Virus check frequency	:	This would be the equivalent of a mandatory testing programme.
Recovery chance	:	Dependent on the virus, this could be influenced by medical care. Even if the virus itself could not be directly cured, some of the effects of the virus could be mitigated or reduced by the care, increasing recovery chance.
Gain resistance chance	:	This could not be controlled in it's current form where resistance only comes post infection. Resistance can be controlled directly through vaccination, but it is a different mechanism would need to be implemented.

b) We choose parameter average node degree, which measures the average number of connections nodes have to other nodes. We ran the tests 10 times per step, for values between 2 and 7, below this the virus rarely took hold, above this always resulted in a 100% infection rate. The results may be seen in figure 1. As the average node degree increases it becomes easier/quicker for the virus to spread. With a small number of for the average node degree, you often do not have a fully connected graph which creates isolated pockets of nodes that cannot spread the virus to other pockets. As the value increases the degree of connections within the graph increases, the virus spreads more easily/quickly. Therefore the number of people infected increases. This connection is however not linear. We can see in the graph below when there is a large increase in nodes that have at some point been infected between 4 and 5.

The initial values used were as follows:

- `number-of-nodes`: 150
- `initial-outbreak-size`: 3
- `virus-spread-chance`: 2.5%
- `item-check-frequency`: 1 tick
- `recovery-chance` 5.0%
- `gain-resistance-chance` 5%

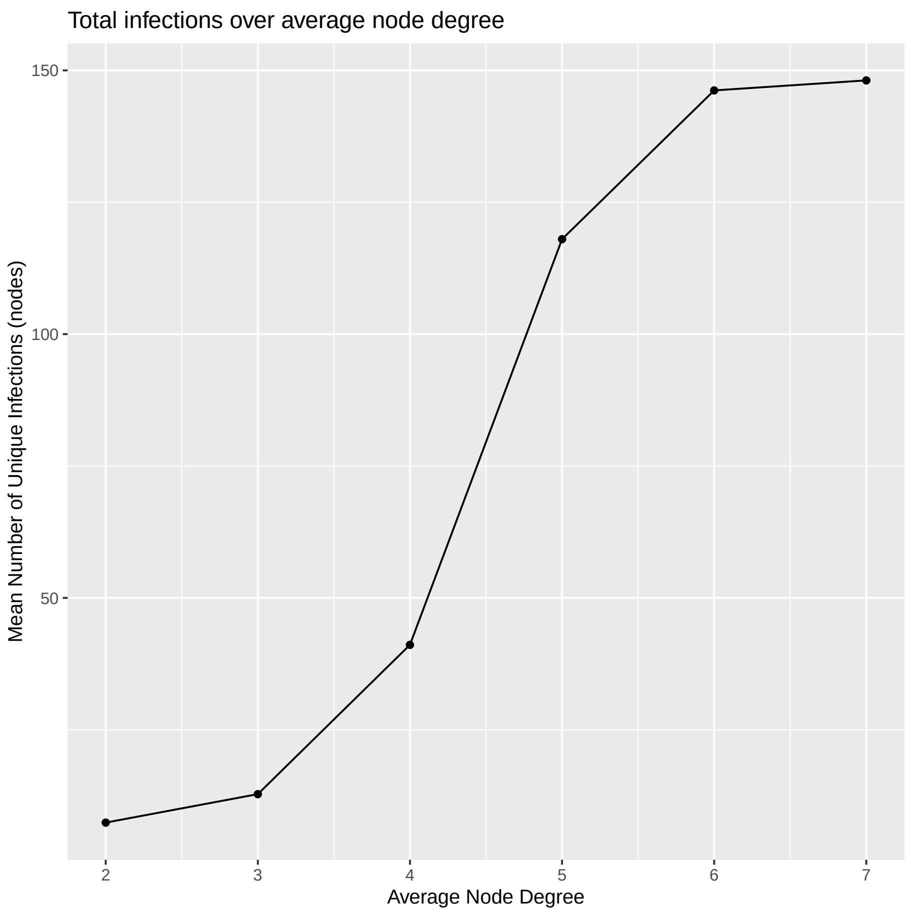


Figure 1: The mean total number of infections per node degree, over 10 runs

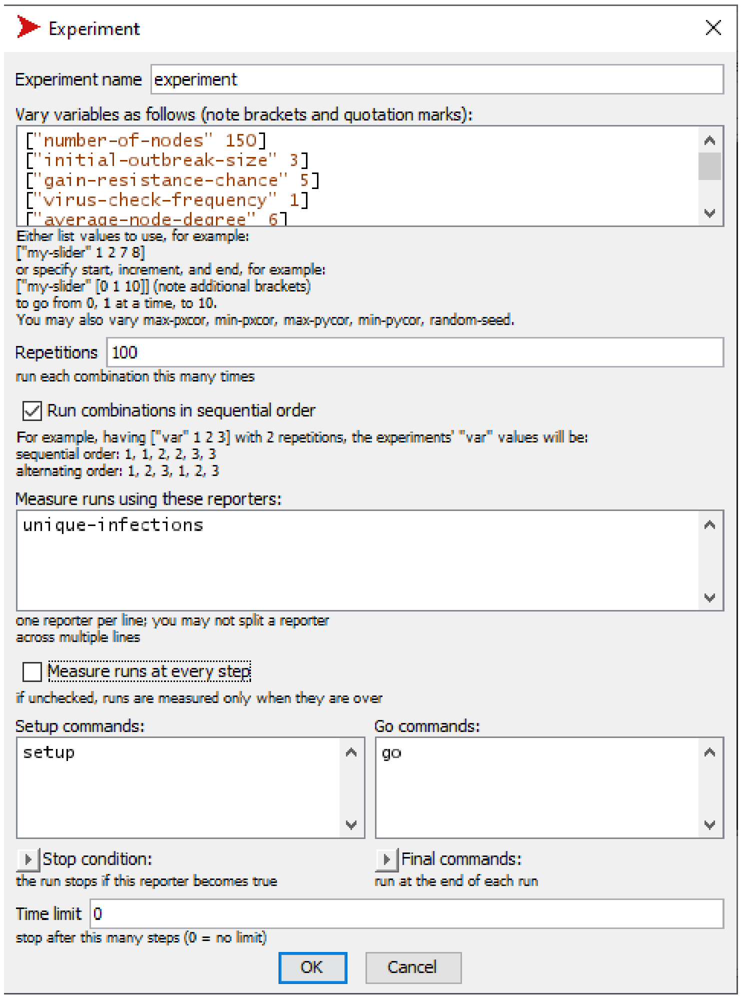
c) We choose to investigate the effect of the parameter `virus-spread-chance` in this experiment. When setting up the BehaviorSpace we made the following changes:

- `["virus-spread-chance" 3]` was changed to `["virus-spread-chance" [0.1 0.1 6]]`, testing with zero clearly results in only 3 infects, our initial infection, initial smaller trials with values over 6 resulted in a consistent 150 infections, so we chose a range of (0,6]. We used 0.1 as our interval so we could get more granular results, than a larger interval but still retain a reasonable processing time.
- Repetitions was set to 1000 this value was chosen as a compromise between ensuring the reliability of our data and the time taken to run the

experiments, in this case 60000 runs.

- We set the "Measure runs using these reporters:" to `unique-infections` as this was our dependent variable.
- We unchecked "Measure runs at every step" as we were only interested in the final values.

These can be seen in Figure 2. All other values we left at their default settings.



Experiment

Experiment name: experiment

Vary variables as follows (note brackets and quotation marks):

```
["number-of-nodes" 150]
["initial-outbreak-size" 3]
["gain-resistance-chance" 5]
["virus-check-frequency" 1]
["average-node-degree" 6]
```

Either list values to use, for example:
["my-slider" 1 2 7 8]
or specify start, increment, and end, for example:
["my-slider" [0 1 10]] (note additional brackets)
to go from 0, 1 at a time, to 10.
You may also vary max-pxcor, min-pxcor, max-pycor, min-pycor, random-seed.

Repetitions: 100
run each combination this many times

[✓] Run combinations in sequential order

For example, having ["var" 1 2 3] with 2 repetitions, the experiments' "var" values will be:
sequential order: 1, 1, 2, 2, 3, 3
alternating order: 1, 2, 3, 1, 2, 3

Measure runs using these reporters:

```
unique-infections
```

one reporter per line; you may not split a reporter
across multiple lines

[ ] Measure runs at every step

if unchecked, runs are measured only when they are over

Setup commands:

```
setup
```

Go commands:

```
go
```

Stop condition:
the run stops if this reporter becomes true

Final commands:
run at the end of each run

Time limit: 0
stop after this many steps (0 = no limit)

Figure 2: BehaviorSpace settings used for Question 3

d) The first six rows of the data contain information about the setup used to run the experiments.

- Contains the version of Netlogo it was ran on: `"BehaviorSpace results (NetLogo 6.2.0)"`
- Contains the ”Virus on a Network.nlogo”
- ”experiment”
- ”02/17/2021 18:27:46:491 +0100”
- ”min-pxcor”,”max-pxcor”,”min-pycor”,”max-pycor”
- ”-20”,”20”,”-20”,”20”

The following rows contain the variables from our experiment including the dependent variable in our case `unique-infections`.

e) The code below reads in the data, reformats it as required and stores it in the dataframe `mean_data`.

```
dataset <- read.csv("0.1-0.1-6.csv",
            skip=6, row.names=1, header=TRUE)
dataset <- dataset[-c(1,2,3,4,5,7,9),]
row.names(dataset)[1] <- "virus_spread_chance"
row.names(dataset)[2] <- "step"
row.names(dataset)[3] <- "unique_infections"
dataset <- as.data.frame(t(dataset))
dataset$virus_spread_chance<-as.numeric(
            as.character(dataset$virus_spread_chance)
        )
dataset$unique_infections<-as.numeric(as.character(
            dataset$unique_infections)
        )
mean_data <- dataset %>%
    group_by(virus_spread_chance) %>%
    summarize(averaged = mean(unique_infections), .groups="keep")
```

The code below creates and shows the plot figure 3.

```
ggplot(data = mean_data, aes(x=virus_spread_chance, y=averaged,
    group=1)) +
    geom_line() +
    geom_point() +
    xlab("Virus Spread Chance") +
    ylab("Mean Unique Infections over 1000 runs (nodes)") +
    ggtitle("Total Infections over Virus Spread Chance")
```

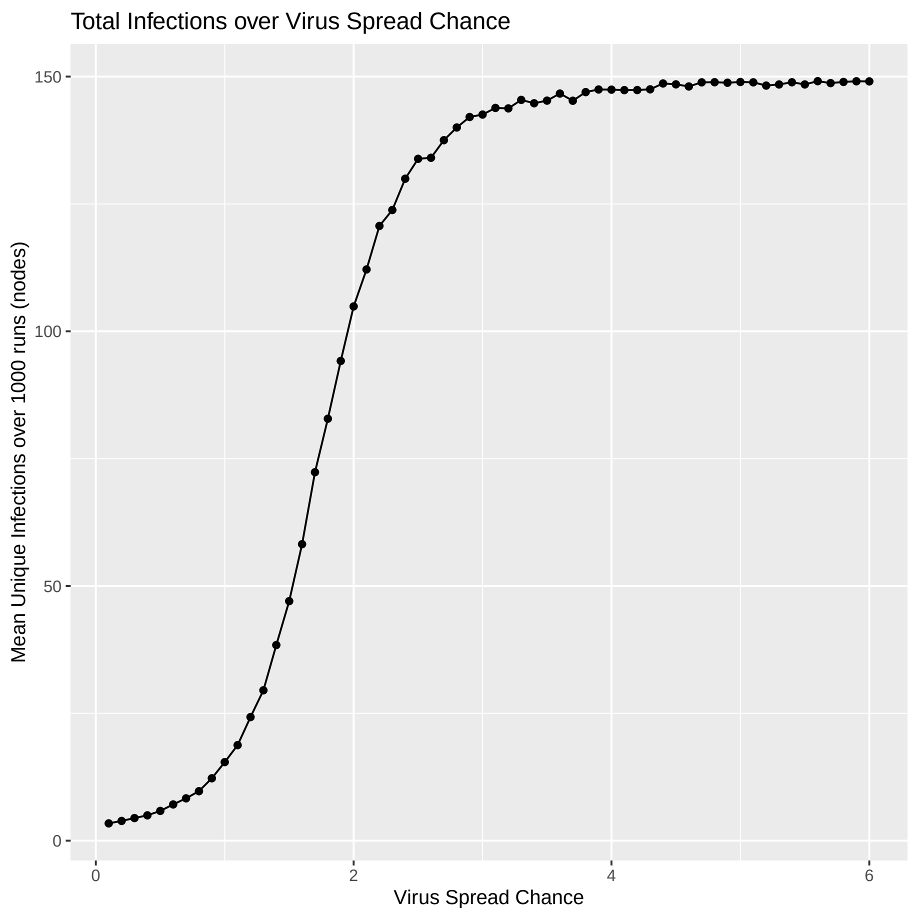


Figure 3: The mean total infections as the virus spread chance increases, taken from 1000 samples per infection chance.

The Figure 3 shows the relations between the chance of a virus spreading and the number of nodes that will have been infected at some point over the test. Clearly with small chances of spreading the number of nodes infected will be small, and with a higher chance more will become infected. The figure shows is the tipping point between these extremes is in the region of 1-2%, this is the point at which it rapidly changes between the two extremes of minimal infections, near universal infections.

f) We could look at a form of non-linear regression to model the relation between the chance of a virus spreading and the total number of nodes infected over the course to the test. Due to the nature of the results we would suggest using spline regression, this allows for modelling data with levels of curvatures that would be difficult to capture with polynomial regression.