Complex Systems and Network Science: Why Your Friends Have More Friends than You

Alex Citeroni
1052175

alex.citeroni@studio.unibo.it

Abstract. This is the report of the Complex Systems and Network Science project, carried out through NetLogo, and based on the article published by Scott L. Feld in the American journal of sociology Why Your Friends Have More Friends than You. I will try to simulate the evolution over time of relationships between people, with the aim of connecting all the people present in the simulation to each other through a relationship. This paper presents the explanation of the interface, the model used, and the results achieved.

1 Introduction

This document describes the model that has been implemented to simulate the basic logic described in the paper by Scott L. Feld, it also describes the interface created in NetLogo, some of the tests that have been performed using the implemented model and the results obtained.

Scott L. Field's basic logic can be described in very simple terms: if there are some people with many friendship ties and some with few, those with many ties show up disproportionately in friend groups.

To try to understand this phenomenon we can look at the distribution of the number of friends and the distribution of the number of friends of friends.

The distribution of each friend's friend considers some individuals more than once. Each individual's friend contributes to the average number of friends of final friends as many times as there are friends.

Whenever any individual compares their number of friends to the average number of friends of friends, the comparison is unfair.

The average number of friends of each individual is

$$AVG_f = \frac{\sum_{i \in I} x_i}{N} \quad (1)$$

where x_i is the number of friends of individual i, N the total number of individuals and I the set of individuals. On the other hand, an individual's average number of friends is

$$AVG_{ff} = \frac{\sum_{i \in I} x_i^2}{\sum_{i \in I} x_i} \quad (2)$$

since an individual has x_i friends and contributes to the mean of each of his friends.

2 Network Model

The simulation environment will be populated by *N* agents and the number of agents can be freely chosen by the user and will affect the duration of the entire simulation (more iterations are required to connect more people).

Each agent represents a single individual who can be connected to other agents via a link. Each link represents a relationship. Initially no agent has relationships with other agents.

By clicking the "*go one*" button, the first iteration of the simulation will be carried out, creating the first relationship between agents randomly, with a probability called *connection-probability* (freely decided by the user) whether the relationship is created or not.

Using the "*go*" button, the iterations will automatically continue in a loop, until the end of the simulation. The simulation will end when all agents are connected to each other.

To generate noise, a parameter called *remove-edge-probability* was created that you can modify to your liking, which represents the probability of removing a relationship.

2.1 Erdős-Rényi network configuration model

The model chosen is that of Erdős-Rényi. Initially, people are created without any type of connection and proceeding with the iterations, connections are created one at a time in a random way, based on the probability of connection that has been chosen (each connection is random and independent of the current network structure).

I chose this model because it seems to me the simplest, most intuitive and similar to a very simplified reality, given that at the beginning of our life we are born without having any relationship (except with our family members), continuing with our life we make friends very often randomly: we know who is around us, sometimes not even by our choice, just because we ended up together in that situation, and with some we manage to create lasting relationships (which can end voluntarily or not), while with others not.

In this project there is also a possibility of removing previously created links. The links will be removed randomly and independently of the current network structure.

2.2 Add-edge method

```
to add-edge
  let node1 one-of turtles
  let node2 one-of turtles
  ask node1 [
    ifelse link-neighbor? node2 or node1 = node2
    [add-edge]
    [create-link-with node2]
]
end
```

This is the method used to add a connection between two people.

When you have a node in your network that is already connected to most of the other nodes, if that node is chosen as one of the two nodes that we try to create an edge between, it will likely be rejected because it is likely already connected with the another chosen node.

```
to add-edge
  let node1 one-of turtles
  let node2 one-of turtles
  ask node1 [
    ifelse link-neighbor? node2 or node1 = node2
    [add-edge]
    [create-link-with one-of other turtles with [ not link-
neighbor? myself]]
]
end
```

In this alternate version of add-edge, NetLogo explicitly looks for a node that isn't already connected to the first one (although there are very few). This could make a big difference.

Depth first. Identification of connected components is done using a standard search algorithm called "depth first search". "Depth first" means that the algorithm goes deep into a branch of connections first, tracing them to the end. For a given node, it scans its neighbor's neighbors (and then their neighbors, etc.) before moving on to its next neighbor. The algorithm is recursive so in the end all the nodes reachable from a particular starting node will be explored. Since all reachable nodes need to be found, and since it doesn't matter what order they are in, another algorithm such as "breadth first search" would have worked just as well. Thorough search was chosen because it is the easiest to code.

2.3 Connection Probability

If we choose a connection probability equal to 1, in each iteration there will certainly be a connection between two people.

If instead we choose a connection probability equal to 0, in each iteration there will be no new connection.

It is possible to vary at will between these two values to have different results.

2.4 Remove Edge Probability

If we set a remove edge probability of 0, no links will be removed in the next iteration.

If we set a remove edge probability of 1, a link between randomly chosen people will be removed.

You can vary between these two values to your liking to have different results.

2.5 Expected results

I've tried to keep the model simple enough, so that I can predict the outcomes I might have.

I expect to see that by setting the connection probability to 0, the simulation will never end, regardless of the number of people or the probability of disconnecting the connections.

I expect the same result setting the connection probability to 1 and the remove edge probability to 1.

I assume that the simulation will end if we have the connection probability value at 1 and the remove edge probability at 0, the time it will take to finish will depend on how many people are in the simulation, the more people there are, the longer it will take to finish the simulation.

I think the simulation will finish even with a remove edge probability of less than 0.5.

The model should also be able to demonstrate the random graph theory of the mathematician Erdős-Rényi, i.e., that the largest connected component of a network formed by randomly connecting two existing nodes per time step, grows rapidly after the average number of connections per node is equal to 1. In other words, the average number of connections has a "tipping point" where the network undergoes a "phase transition" from a rather disconnected world of a bunch of small fragmented components, to a world where most of the nodes belongs to the same connected component.

3 Graphic interface

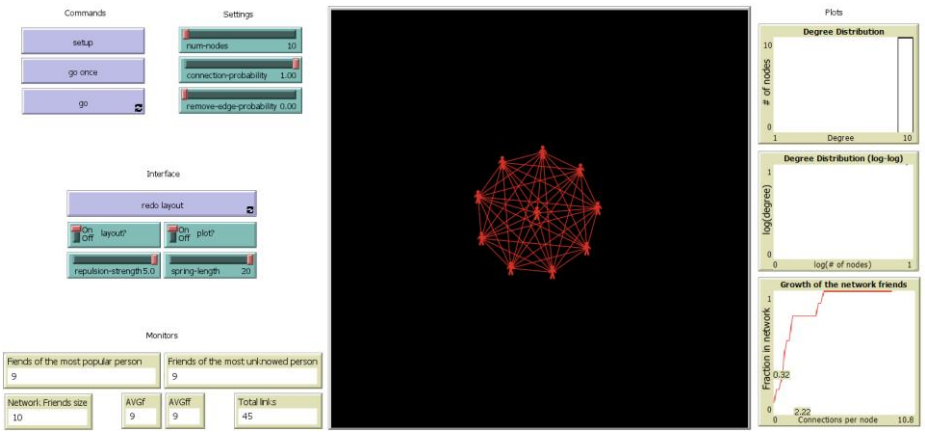

Fig. 1. Graphical user interface of the program.

In the GUI as seen in figure 1, there are several buttons, plots, sliders, switches and monitors. There are mainly five sections: commands, settings, interface, monitors and plots.

We can also see that the largest network of people is colored in red, while the others are colored in grey.

3.1 Commands

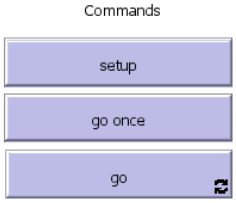

Fig. 2. Command section in the graphical interface, containing three buttons: setup, go once and go.

Let's start by considering the commands, which as can be seen from figure 2 are made up of three buttons: setup, go once and go.

Setup. Button used to create the simulation based on the number of people set in the settings section.

Go Once. Execute an iteration.

Go. Loop, iterating until all people are connected by relationships. It can be useful to reach the end of the simulation faster.

3.2 Settings

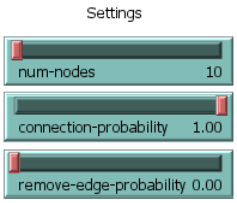

Fig. 3. Settings section in the GUI, containing sliders regarding num-nodes, connection-probability and remove-edge-probability.

As we see in figure 3, the settings consist of three sliders: num-nodes, connection-probability and remove-edge-probability.

Num-nodes. With this slider it is possible to change the number of people initially present in the simulation, from 2 to 500.

Connection-probability. This parameter indicates the probability of connection between two people in an iteration, it goes from 0.00 to 1.00, if it is set to 0.00 there will be no connection, while if it is set to 1.00 there will certainly be a connection during the iteration.

Remove-edge-probability. From this slider you can choose the probability of removing relationships between people.

3.3 Interface

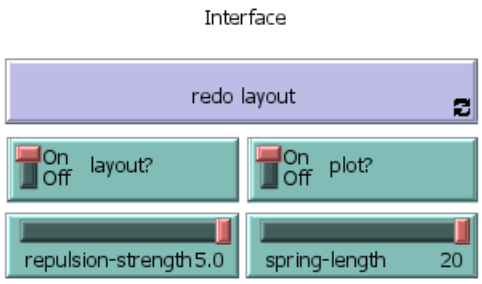

Fig. 4. Layout settings section of the graphical interface, containing two sliders, a button and two switches.

As we see in figure 4, this section consists of a button called redo layout, two switches, called layout and plot, and two sliders, called repulsion-strength and spring-length.

Redo layout. Clicking this button keeps people moving until their animation ends (may affect performance).

Layout. Through this switch it is possible to decide whether people will move from their positions when they are connected by a relationship or not (if redo layout is active this switch loses its function). It can be useful if we need more performance.

Plot. Setting this switch to Off will no longer update plots and vice versa.

Repulsion-strength. This slider allows you to change the distance between one person connected to another, allowing you to change the parameter from 0 to 5.

Spring-length. With this slider it is possible to modify the length of the relationship between the people, allowing to set the parameter from 2 to 20.

3.4 Monitors

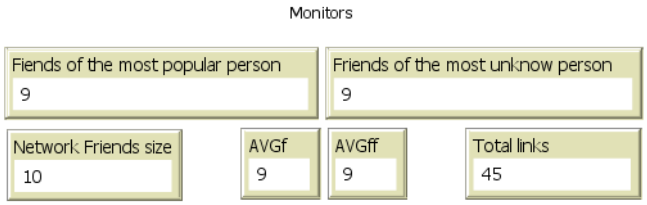

Fig. 5. Monitors used to show data of interest to us regarding the current simulation.

As we see in figure 5, six monitors are available in this section: friends of the most popular person and friends of the most unknown person, AVG_f and AVG_{ff}, total links and network friends' size.

Friends of the most popular person. Number of friends of the most closely connected person.

Friends of the most unknow person. Number of friends of the least connected person.

$\boldsymbol{AVG_f}$**.** In this monitor it is possible to view the average number of friends of all the people present in the simulation.

$\boldsymbol{AVG_{ff}}$**.** Here we can view the average number of friends of friends of all people in the simulation.

Total links. Total links present in the simulation.

Network friends' size. Number of people connected to the network with the most connections (giant component).

3.5 Plots

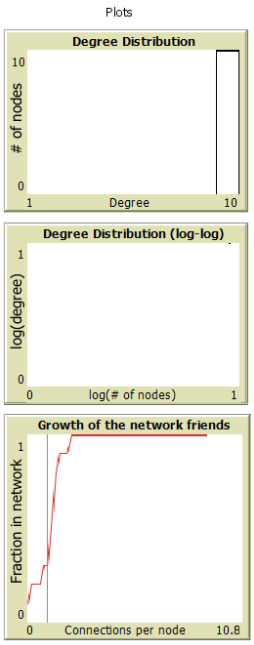

Fig. 6. Plots showing the evolution of the current simulation.

As we see in figure 6, in this section we find three different plots: Degree Distribution, Degree Distribution (log-log) and Growth of the network friends. These plots will only be updated if the Plot switch is On.

Degree Distribution. In this bar plot we can visualize on the x-axis the number of connections between people; instead, on the y-axis there is the number of people who have that number of connections.

Degree Distribution (log-log). This line plot shows the same thing as the Degree Distribution plot but on a logarithmic scale.

Growth of the network friends. Here we can see the growth of the network, the x-axis shows the number of connections per person, while the y-axis shows the fraction of the network that is connected. It can also be seen when the value 1 is exceeded (critical point of the Erdős-Rényi model).

3.6 Turtle

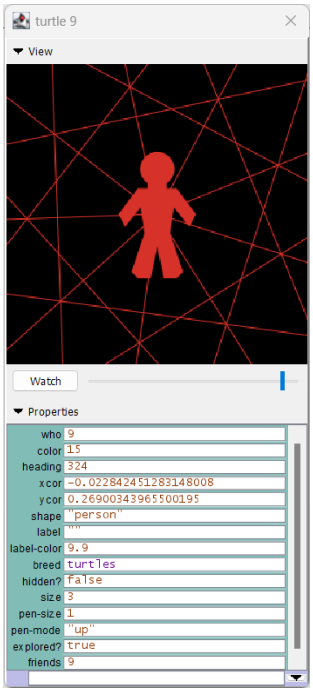

Fig. 7. Data present on every single person.

By clicking on a person with the left mouse button and navigating on "inspect turtle", it is possible to view his personal data, where we can also see the number of friends he currently has, as we see in figure 7.

4 Test

Many tests have been carried out with different parameters and the following are just some of the many carried out. It was noticed that when the simulation ends AVG_f and AVG_{ff} are equal and are always equal to the number of people – 1.

num-nodes=10, connection-probability=1, remove-edge-probability=0. Keeping these settings, we can see that the simulation finished in 45 ticks, creating 45 links and a network of 10 people. The number of ticks does not change by repeating the test.

num-nodes=10, connection-probability=0, remove-edge-probability=0. Using these parameters, the simulation will never end, because no links are created and no values change.

num-nodes=10, connection-probability=0.5, remove-edge-probability=0. By carrying out several tests with these parameters, I noticed that the simulation ends most of the time in a range that goes between 80 and 100 ticks.

num-nodes=10, connection-probability=1, remove-edge-probability=0.5. Using these parameters, the simulation ended most of the time in a range from 60 to 150 ticks.

num-nodes=10, connection-probability=0.5, remove-edge-probability=0.5. Using these parameters, the simulation ended most of the time in a range from 500 to 6000 ticks.

num-nodes=20, connection-probability=0.5, remove-edge-probability=0.5. The first time I used these parameters the simulation finished in 35706 ticks, while the second time it finished in 28635 ticks. The third time it took only 8496 ticks, the fourth time 7168 ticks, the fifth time 4523 ticks. Here I was starting to think that the more trials you did the fewer ticks needed to finish. On the sixth I returned to 26642 ticks and on the seventh I even reached 114856 ticks. These other two results have belied what I had started to think, confirming that everything is random.

num-nodes=10, connection-probability=1, remove-edge-probability=1. With these values the simulation will never end because when one link is created another one is deleted, having only one link at a time, $AVG_f = 0.2$ and $AVG_{ff} = 1$.

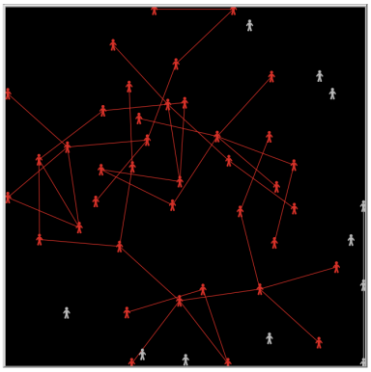

Fig. 8. Test with num-nodes=50, connection-probability=1 and remove-edge-probability=0 at tick 41.

num-nodes=50, connection-probability=1, remove-edge-probability=0. In this test shown in figure 8, at tick 41 the most popular person has 5 friends, while there are still several people who have no friends; the largest network of friends is 39 people and the total links are 41, $AVG_f = 1.64$ and $AVG_{ff} = 2.488$.

5 Results

Most of the results met expectations. In fact, some deliverables can be repeated by setting the same parameters. By setting connection-probability=1 and remove-edge-probability=0, the results will be achieved with the same number of ticks, while the evolution of the network being random is very difficult to repeat in the same way.

As I said, the parameters change the results considerably, in fact, as the tests show, by inserting some parameters the simulation has no end, while by inserting others the simulation ends more or less quickly.

The ticks needed to finish the simulation are greatly influenced by the number of people initially present in the simulation and by the connection-probability and remove-edge-probability parameters.

The higher the remove-edge-probability parameter is, the more ticks will be needed to finish the simulation and vice versa.

The lower the connection-probability parameter, the more ticks it will take to finish the simulation and vice versa.

The more people there are in the simulation, the more ticks it will take to finish it and vice versa.

Placing more people in the simulation will also create more stressful situations for the computer and therefore require more computing power to continue with the simulation.

What I noticed is that AVG_{ff} during the simulation (after doing the first iteration and before finishing the simulation), is always bigger than AVG_f.

This model confirms what Scott L. Feld said in his article: the number of friends of the single individual (AVG_f) is for the most part less than the number of friends of friends (AVG_{ff}), and this can make most people feel disadvantaged.

References

1. Complex Systems and Network Science, Exam Project Specification, Ozalp Babaoglu and Nicolas Lazzari, 2022/2023 Academic Year
2. Scott L Feld. Why your friends have more friends than you do. American journal of sociology, 96(6):1464–1477, 1991
3. Wilensky, U. (2005). NetLogo Giant Component model. http://ccl.northwestern.edu/netlogo/models/GiantComponent. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.
4. Wilensky, U. (2005). NetLogo Preferential Attachment model. http://ccl.northwestern.edu/netlogo/models/PreferentialAttachment. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.
5. ErdosRenyiTwoComponents.nlogo