Network Graphs as a Means to Demonstrate Infection Spread

Elnoel Akwa
Computer Science
University of Kentucky
Lexington Kentucky
edak223@uky.edu

ABSTRACT

The spread of viral infections has been reintroduced as a topic of research since the 2020 covid pandemic. Serval research has now been conducted in the scope of contact tracing to predict and possibly control the flow of viral infections. Previous research in the computer field has demonstrated that a community of individuals interacting with each other could be represented in the form of a network graph, with the nodes representing the various individuals while the edges indicate their relationships with each other. Through this correlation, we could implement various algorithmic solutions to solve complex problems in real-life communities such as contacting and infection detection. Through shortest paths algorithms such as Dijkstra and A* we could obtain close estimates of the possible flow paths of viral infections enabling us to produce more effective means of containing the infection spread.

CCS CONCEPTS

• Network → Network-Graphs• Computing algorithms →A_star; Dijkstra Search.

KEYWORDS

Infection Spread, Contact tracing

ACM Reference format:

Elnoel Akwa 2022. Network Graphs as a means to demonstrate infection spread. *May, 5th, 2022, Lexington, KY, USA*. 3 pages.

1 INTRODUCTION

Infectious diseases are illnesses caused by pathogenetic micro-organisms, also known as transmittable or communicable diseases. These diseases usually move from one person to another who are usually in close contact by mediums such as fluids, air, vectors (lower organisms in the food chain such as mosquitos) and direct contact. From the year 2020 till date, the world has been stroked with a respiratory illness pandemic caused by the SARS-coV-2, commonly known as COVID-19. From the year 2019 till date, there have been over 6.2 million confirm deaths and above 517 million cases worldwide [3]. As seen from the data above not only has this caused a great lose to human life but it has also caused significant negative impact on social development and world economy.

Previous years of computation research has proven a community of individuals living an interacting with each other could be represented using network graphs[1]. In addition, serval standardized network graphs such as the caveman and gaussian graphs have been developed to represent a particular community type. The idea being everyone is represented by a node while an edge represents the two individuals are in communication/contact with each other. Combining this with what we know about pathogenic infection spread we could come out with a means to predict the most likely flow pattern of a pathogenic infection in a community represented by a network graph. In addition to predicting the flow path of an infection, given that we have all relationship between nodes identified we could establish a contact tracing mechanism which will be my goal in this project.

This project report will serve as a walkthrough on the various step taken the author used to demonstrate Infection spreads using network graphs and computer algorithms. Various tradeoffs will be discussed about too and possible areas of improvement.

1.1 Motivation

As mentioned above the motivation I had for this project was primarily the newly introduced SAES-coV-2 pandemic and the new research been done to develop cheap and efficient contact tracing means to detect and possibly contain the infection flow. Also, one thing that is certain solution to the above defined problems could be implemented on serval other scenarios in the biomedical and network fields.

2 IMPLEMENTATION

The program was designed to detect the possible flow path of bacterial/viral infection in a sample community and possibly serve as a means for contact tracing. For instance, if a member (node) in a community gets infected by an

illness the program will be able to provide the high-risk nodes based on the connections with the infected node. The main idea here is that each edge represents the interconnections between nodes, we should be able to determine the probability a node gets infected based on the number of edges separating the node in question from the infected node.

A combination of two algorithms was used as a backbone for this project. The A Star (A*) search algorithm was used to get the length of edges between the contaminated and target node. Based on the length the probability of infection and hence the node color is determined. Also based on that probability we can determine if the node is a high-risk node or not. The second algorithm used was Dijkstra's algorithm. Passing the source (infected) and target nodes, we could get the list of all possible viral flow paths and hence implement a contact tracing mechanism.

The program was implemented using Python 3 and a Layered architectural pattern where each subtask has a unique function called sequentially as the program is being executed. Each function acted individually with the exception of a few that used global variables to intercommunicate with each other. Two python packages were used in the project implementation namely:

- NetworkX for the creation and manipulation of the network graphs used in the illustration of our project
- Plotly was used simultaneously with NetworkX to create an interactive graph visual of the graph

The Program execution can be broken down into three main parts; graph creation, spread implementation and contact tracing.

2.1 Graph Creation

The graph was not made to model any particular community but instead to work for a wide different type of communities. For this reason, a random geometric graph function was used where the function's two parameters were community size and probability of edge between nodes.

2.2 Spread Implementation

Once the network graph has been created the next step will be to predict the flow. As we know pathogens uses a medium to transfer from one individual to another. In other words, this means there must be sort of a connection between two people for an infection to happen. In the case of the SARS-coV-2 virus, the two individuals in question must have had a physical contact with the more amount of contact meaning an increase chances of contamination. According to a study done by the researchers from the National Key research the probability a person gets infected giving a handshake to a infected individual is 78% with that number increasing to 87% in case of any immune deficiency[4]. However, the same research states, prior to a person gets infected (2-3 days after being in contact with an infected person) the chances the carrier is a vector of the disease himself is a lower 63%. Assuming an edge is a direct contact between two nodes and none of the individuals an immune compromised, a secondary edge can be defined as Num_edges =2 and so forth. From this we could induce the more edges separating the infected node form a particular node the less likely the node is to become contaminated.

Implementing an A*star length algorithm, we can get the distance to the infected node from a starting node and hence implementing a for loop get the probability of infection for every member/node in the sample community. Passing this information to ploty we could get a graph similar to the one in **figure 1** where the infected node is purple and the whiter the nodes are the less likely they are from being infected with the virus.

2.3 Contact Tracing

Contact tracing is the process of identifying assessing and managing people who have been exposed to a disease in order to contain the infection a managing people who have been exposed to prevent further infection. How this is implemented in, or program is by use of the Dijkstra algorithm where if we got a confirm infected node other than the primary infected node zero, the two nodes id are passed as arguments to a modified Dijkstra function. The modification in this case is we save each output are run the function recursively until no two outputs are same. This way we get all the possible paths the infection could have used to travel and insolate all nodes.

3 TESTING

The program was tested for various errors during execution. Each run was successful for sample sizes between 0 and 2300. For numbers

above that the program took a significantly longer amount of time to load and often became unresponsive. Different other graph types were used when testing program namely caveman and graph to test the program's portability and it did still perform well.

3.1 Additional Features

Additional features that were implemented using the program are:

- High risk cases: from the various node probabilities, the program was able to get the list of high-risk node case whose probability exceeded 50% likely infected.
- Cluster Detection: Based on the number of edges connected to a node, the node sizes can be made bigger or smaller that way we know the most venerable nodes in the community where the infection could have a greater spread radius.

4 LIMITATION

The primary limitation the program has is the that our current algorithm cannot account t for more than one infected node. In some sense it is realistic in our current scope since there is usually just a single patient zero for every given infection but when going down to subcommunities it is possible to have 2 or more "patient-zero".

4.1 Future works

Going forward additional works and fixes to be done to the program will center around:

- Implementing a solution for two or more infected nodes in sample community
- Extract and interpret more relevant data with respect to the field of viral spread in network graphs
- Explore various ways in which this implementation could be used in other fields out of biomedics.

5 CONCLUSION

This article is aimed at researchers in the areas of network graphs and computational biology. Researchers interested in using network graph and algorithmic techniques in solving biomedical problems will also be interested in this article and are expected to gain important insights. This article will also attract network system designers to find out if their various research projects and implemented solutions could be used in a wider scope

REFERENCES

[1]David Kosiur. 2001. *Understanding Policy-Based Networking* (2nd. ed.). Wiley, New York, NY.. Collected algorithms from ACM. (1964). Algorithms department of Communications of the ACM CACM.
[2]Toshiya Kuramochi, Naoki Okada, Kyohei Tanikawa, Yoshinori Hijikata, and Shogo Nishida. 2012. Community Extracting Using Intersection Graph and Content Analysis in Complex NetworkAUTHOR (YEAR). *Title of publication.* Publisher.
[3]Amee Trivedi and Deepak Vasisht. 2020. Digital contact tracing: technologies, shortcomings, and the path forward..
[4]Ting Jiang, et Al. 2022. A Survey on Contact Tracing: The Latest Advancements and Challenges