UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

OPTIMIZING THE SPREAD OF INFLUENCE IN SOCIAL NETWORK COMMUNITY STRUCTURES

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE

By

AHMED TAWFIQ
Norman, Oklahoma
2019

OPTIMIZING THE SPREAD OF INFLUENCE TO SOCIAL NETWORK COMMUNITY STRUCTURES

A THESIS APPROVED FOR THE
School of Industrial and Systems Engineering

BY THE COMMITTEE CONSISTING OF

Dr. Kash A. Barker, Chair

Dr. Andrés D. González

Dr. Shima Mohebbi

Acknowledgements

I appreciate the love and support of my family specially my mother for encouraging me to follow my dreams, I would like to praise my father for setting a great example for me. Special thanks to my amazing wife Dalal for her continuous motivation and support in this beautiful journey.

I would like to thank Dr. Barker for his encouragement and guidance throughout my thesis research. I also would like to thank Dr. Mohebbi and Dr. González for their contribution to my study. Last but not least, I would like to thank my friends and OU family for being a great part of my journey. BOOMER SOONER!

Table of contents

List of Tables

List of Figures

Abstract

Natural disasters and disruptive events were a major reason for researchers to study networks and community vulnerability. Detecting communities is considered a key element to better understand social networks. This detection will allow researchers to discover community structures inside the network and apply several methods to determine influencers inside each community, which in terms will help in evaluating community vulnerability. In this study, Girvan Newman community detection algorithm is applied to detect communities in social networks. This algorithm detects communities based on their betweenness centrality. Several methods have been established to study the spread of influence in social networks such as the Linear Threshold model. Understanding the spread of influence inside communities will help in categorizing community vulnerability. After detecting communities, an influence optimization method using Linear Threshold will be applied to help identifying optimal influencers in each community. The proportion of influencers in each community will be the indicator of social vulnerability. The higher the proportion of influencers in the community, the more resilient the community will be in terms of spreading information inside the network. Sensitivity analysis will be implemented to evaluate the behavior of each community when changes are made to thresholds and the number of initial influencers. The main goal of this study is to identify vulnerable communities and prioritize them, which can help in preparedness for any disruptive event such as natural disasters.

Chapter 1.0 Introduction

This chapter establishes the basis of this research including an overview of most used community detection methods that can be applied to social networks, explaining the implementation of influence maximization models to analyze the spread of influence in social networks. This chapter then demonstrate the goals, objectives and methodology for the study.

1.1 Background and Problem Statement

The interest in social networks has been increasing specially after the recognition on the importance of spreading information online. In fact, social networks serve as an essential medium to spread information. (Nazemian & Taghiyareh, 2012). The phenomenon of studying and understanding the importance of social connections in societies and how it affects the process of decision making was first introduced by (Granovetter, 1978). It also gives suggestions on how to take advantage of social influence (Cao, Wu, Wang, & Hu, 2011).

Researchers are paying more and more attention to social networks, and how it provides natural representation of individuals and their connections. It is also important to emphasize how it can show a pattern of the behavior of an individual in a population (Wasserman & Faust, 1994). The value of social networks appears in how influence can propagate through users' interactions. For example, a twitter user might be followed by hundreds or even thousands of other twitter users. By analyzing his network of followers, a marketing campaign might target individuals to increase the spread of information in their campaign (Zeng et al., 2016).

Many networks display community structures. Based on topology or spatial characteristics, networks can be divided into different groups where nodes inside the group are densely connected and the connection between groups are lighter (Rocco, Barker, Moronta, & Ramirez-Marquez, 2018). Based on (Shen, 2013) community holds two definitions, local and global. The local definition is based on the information and properties of the community itself. While the global definition highlights the characteristics of the whole network instead of just focusing on communities. Unfortunately, there is not a common widely used definition of community as it is considered a qualitative concept. However, we can define communities based on the application and purpose of the study (Shen, 2013).

The ability to discover the structure of communities is very important in terms of understanding the network. Community detection methods can be divided into two main classes, agglomerative and divisive methods. Where agglomerative methods depend on calculating similarities between vertices to add edges to form a network, while divisive methods remove edges from the network to form communities (Newman & Girvan, 2004)

1.2 Purpose Statement

This study's purpose is to identify vulnerable communities in social networks, giving them the priority in spreading influence at time of disruption such as natural disasters. By detecting influencers in these communities earlier, the vulnerable communities can be more prepared when needed. The definition of community is almost the same in every type of network, but the characteristics of the community might change depending on the aspect of which network is being studied. For example, communities in social networks can be represented by grouping individuals of similar backgrounds or by grouping individuals of common interests. While communities in web networks can be represented by pages on similar topics. The ability to detect these communities would be beneficial in better understanding the network in a more efficient way (Girvan & Newman, 2002).

1.3 Research Goal

The goal of this research is to propose a framework to first, identify the community structures within a network. Second, determine how influence propagates to those communities. Third, measure the extent to which influence reaches those communities.

The second chapter is the literature review which presents background information related to this study. The third chapter covers the proposed methodology with a detailed explanation of the applied methods. The fourth chapter is covering the implementation of the model and discusses results. The final chapter concludes and summarizes the study’s findings.

Chapter 2.0 Literature review

This chapter presents background information related to this study, this includes most common community detection algorithms, methods and models to detect and optimize influence in social networks.

2.1 Community Detection

Network Science is considered a progressive field in modern research, whereas many systems can be studied as networks of links and nodes. These nodes create sub-sections, where each section holds its unique function. The nodes are connected internally and externally by links forming what is called communities (Lancichinetti & Fortunato, 2009). Researchers have developed many community detection methods using tools and techniques from different disciplines such as biology, physics, social science and many other. Even though these methods have proven their eligibility, there is still not a clear indication on which method is more reliable. (Lancichinetti & Fortunato, 2009).

Community detection methods can be categorized according to several classification such as, searching mechanism, structure to be found, objective function and hypothesis of the model. Many methods use a combination of several approaches for a more powerful algorithm. Detection methods can be classified to the following divisions based on theoretical approach, edge removal, modularity optimization, spectral partitioning, dynamic processes and statistical inference (Dao, Bothorel, & Lenca, 2018).

Edge betweenness algorithms such as the Girvan Newman algorithm detects community by calculating betweenness centrality to remove edges with the higher value in each iteration, while in edge clustering coefficient, the betweenness centrality is replaced by clustering coefficient.

Modularity Optimization methods iteratively aggregates connected communities by greedily maximizes the modularity function Q. Some of the most common modularity methods are Greedy Optimization, Louvain Method, and Spectral Method (Dao et al., 2018).

2.2 Influence Optimization

Studying and analyzing social networks have been the focus of several scientists for decades, and the application of influence propagation were found in several fields such as, viral marketing, social media analytics, the spread of rumors, studying the adoption of an innovation, behavior targeting and social search (Chen, Lakshmanan, & Castillo, 2013). Our daily life has been penetrated by new technologies and social media, providing new ways to communicate and spread information. The influence of social media became dominant on politics, business and society. which has drawn the attention towards findings new methods to study and analyze the spread of influence. Kempe in (Kempe, Kleinberg, & Tardos, 2003) has developed two widely used models to study influence, Independent Cascade Model and Linear Threshold Model (Wang & Street, 2018).

The independent Cascade Model simulates the spread of information in social networks. When a node is activated it will try to activate its inactive neighbor under a probability. Each active node has one trial only to activate its neighbor. The process is independent in the sense that each node will try to activate its neighbor node independently from other neighbors trying to activate the same node. The Independent Cascade model is considered a probabilistic model, and the process of activation is uncertain where results may change for the same network in different simulations (Liu, Chen, Jeon, Chen, & Chen, 2019).

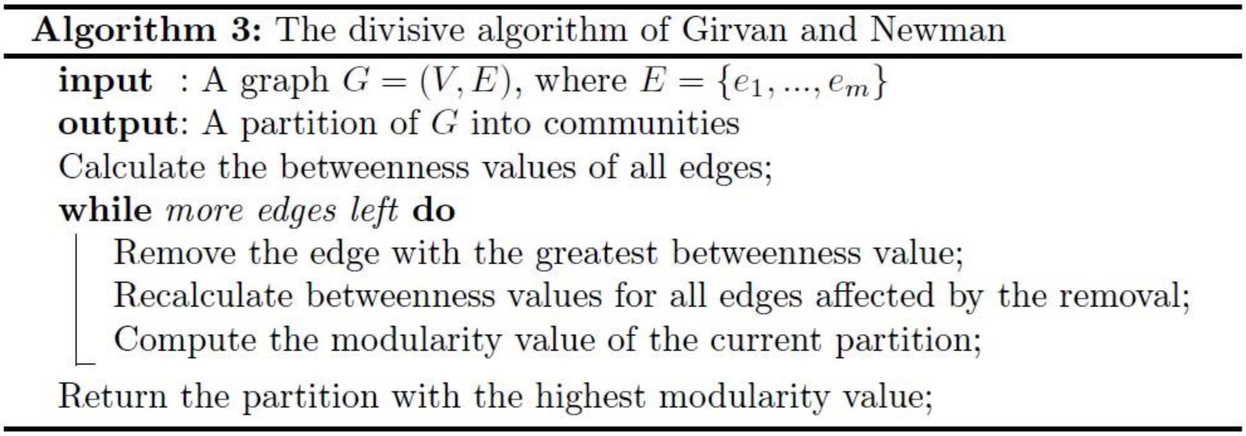
The Linear Threshold Model is considered an accumulation model where each node will be affected from all active neighbors. A specific threshold will be assigned to each node in the network, this threshold represents the minimum amount of influence required for the node to be activated. If the amount of accumulated influence reaching a node from all active neighbors is greater than or equal the threshold, the node will be activated, and it will contribute in activating inactive neighbors. Otherwise, it will remain inactive. Each link in the network will have a propagation probability representing the amount of influence to be sent from each active node (Liu et al., 2019).

Chapter 3.0 Proposed methodology

This chapter details the process in which methods are used to detect communities, study and optimize influence in the network.

3.1 Girvan Newman Algorithm

In this study, Girvan Newman algorithm is applied in order to detect communities in our network. The algorithm is considered one of the popular community detection methods in network science. It applies the concept of edge betweenness centrality by calculating the shortest path between all vertices. Each link will have a weight, links between communities will have a higher weight than others. The algorithm is a divisive algorithm, which means the link with the highest weight will be removed in each iteration until we cannot move inside our network. (Newman, 2004)



```
Algorithm 3: The divisive algorithm of Girvan and Newman
input  : A graph $G = (V, E)$, where $E = \{e_1, ..., e_m\}$
output: A partition of $G$ into communities
Calculate the betweenness values of all edges;
while more edges left do
    Remove the edge with the greatest betweenness value;
    Recalculate betweenness values for all edges affected by the removal;
    Compute the modularity value of the current partition;
Return the partition with the highest modularity value;
```

Figure 1 Girvan Newman Algorithm (Newman, 2004)

A dendrogram will be represented throughout the entire process of the algorithm.

$$\boldsymbol{Betweenness\ Centrality\ (BC)} = \frac{\boldsymbol{\sigma_{i,j\,(v)}}}{\boldsymbol{\sigma_{i,j}}} \qquad \textbf{(1)}$$

Where $\sigma_{i,j\ (v)}$ is the total number of shortest paths between i and j that passes through link *v*, and $\sigma_{i,j}$ is total number of shortest paths between *i* and *j*.

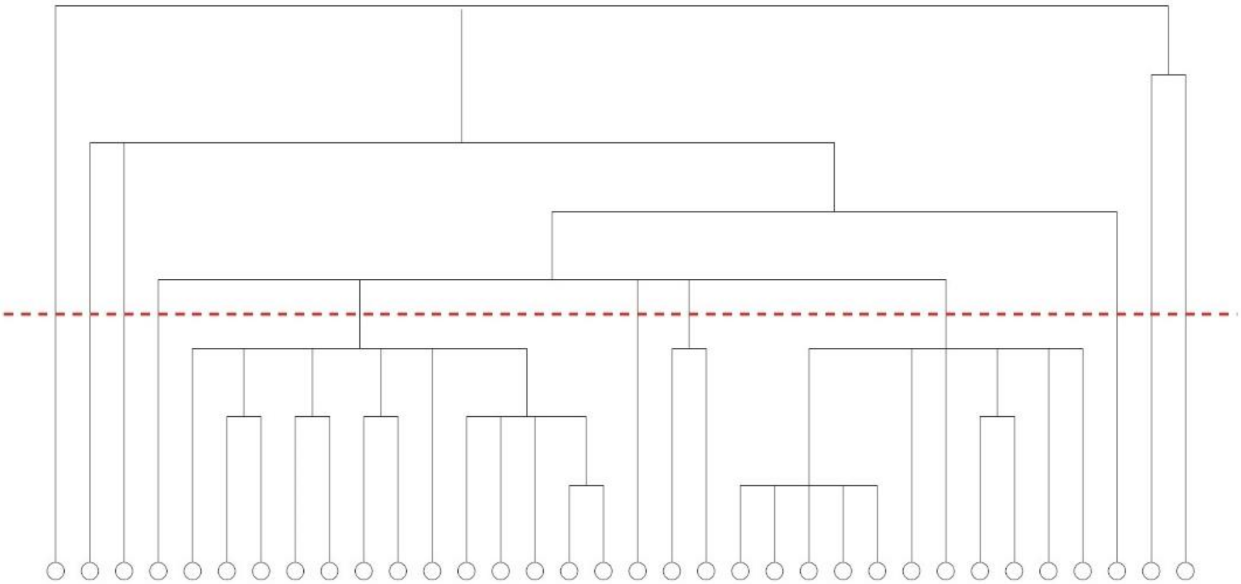


Figure 2 Dendrogram (Newman, 2004)

3.2 Linear Threshold Model

The linear threshold model is one of the methods used to analyze the spread of influence in social networks. The model is considered a stochastic diffusion model represented by Kempe (Kempe et al. [2003]) which represents the behavior of influence. (Chen et al., 2013). For an individual to change their behavior and adopt a new concept, multiple exposure to many sources is required. This type of behavior is represented in the linear threshold model. (Chen et al., 2013)

In this model, a weight $\boldsymbol{b_{i,j}}$ will be assigned between nodes ***i*** and ***j***. this weight is set to represent the probability of an influence to be passed. The sum of weights between each node must be less or equal to one $\sum \boldsymbol{b_{i,j}} \ \leq \mathbf{1}$.

A threshold **θ_i** will be given as well to each node which signifies the required amount of influence needed for a node to be activated. If the sum of weights from active neighboring nodes is equal to or higher than the threshold, the node will be activated, otherwise the node will remain inactive $\sum_{\boldsymbol{Active\ neighbor\ of\ i}} \boldsymbol{b_{i,j}} \geq \boldsymbol{\theta_i}$.

3.3 Optimization of Influence

Influence Optimization model is applied using linear thresholds in order to detect the optimal influence and maximize it in our network. As mentioned previously, the linear threshold model will help in analyzing the spread of influence and detect influencers in the network, while the optimization model will help in maximizing the spread of influence in our network. This model was explained by (Kempe et al., 2003).

Set ***N*** will be representing nodes (individuals) in the network, where we have ***N^I*** our influencers, and ***N^J*** are influenced individual, and ***N^I*** ∪ ***N^J* = *N***.

We have set ***L*** for our links between ***i*** and ***j***. A binary variable ***y_i*** for initial influencers and ***S_j*** for our influenced nodes. The model will try to maximize ***X_j*** the total number of influenced nodes, ***X_j*** will be updated after each iteration trying to spread influence in the network. The parameter ***p_{ij}*** is the propagation probability for each link, and ***M*** is a large number for the maximum bound of ***X_j*** (Yeagle, 2018).

The objective function is to maximize ***σ*** which is equal to the summation of ***X_j*** for every ***j*** belongs to the set ***N^J***.

$$\boldsymbol{Maximize\ \sigma} = \sum_{j \in N^J} X_j \qquad \textbf{(2)}$$

The first constraint

$$\boldsymbol{u_j} = \sum_{j \in N^J} \boldsymbol{p_{ij}} \cdot \boldsymbol{y_i} \qquad \forall\ (\mathbf{j}) \in \boldsymbol{L} \qquad \textbf{(3)}$$

Where ***uj*** holds the summation of propagation probability of each link with initial influencers. In order for a node to be influenced, the sum of the probability propagation has to be equal to or more than the threshold associated to each node, and this will include a binary variable ***Si*** to determine if the node is activated and influenced or not which is shown in the next constraint.

$$\boldsymbol{\theta_j}\ \boldsymbol{S_j}\ \leq\ \boldsymbol{u_j} \qquad \forall\ \mathbf{j}\ \in\ \boldsymbol{N} \qquad \textbf{(4)}$$

The next constraint is to set a limit to the initial influencers in the network.

$$\sum_{i \in N^I} \boldsymbol{y_i}\ \leq \boldsymbol{C} \qquad \boldsymbol{i}\ \in\ \boldsymbol{N^I} \qquad \textbf{(5)}$$

The total number of influenced nodes ***Xj*** must be at least equal to influenced nodes, this is shown in the next constraint.

$$\boldsymbol{X_j}\ \geq\ \boldsymbol{S_j} \qquad \forall\ \mathbf{j}\ \in\ \boldsymbol{N} \qquad \textbf{(6)}$$

The total number of influenced nodes must be at least equal to initial influencers.

$$\boldsymbol{X_j}\ \geq\ \boldsymbol{y_i} \qquad \forall\ (\mathbf{i},\mathbf{j})\ \in\ \boldsymbol{N} \qquad \textbf{(7)}$$

The next two constraint is to manage the spread of influence.

$$\boldsymbol{X_j}\ \leq\ \boldsymbol{s_j}\ .\boldsymbol{Mb_j} \qquad \forall\ (\mathbf{j})\ \in\ \boldsymbol{N} \qquad \textbf{(8)}$$

$$\boldsymbol{X_j}\ \leq\ \boldsymbol{y_i}\ .\boldsymbol{M}(\mathbf{1} - \boldsymbol{b_j}) \qquad \forall\ (\mathbf{j})\ \in\ \boldsymbol{N} \qquad \textbf{(9)}$$

Then we need to define binary variables in our model (Yeagle, 2018).

$$\boldsymbol{b_{ij}}\ \in \{\mathbf{0},\mathbf{1}\} \qquad \forall\ (\mathbf{i},\mathbf{j})\ \in\ \boldsymbol{N} \qquad \textbf{(10)}$$

$$\boldsymbol{y_i}\ \in \{\mathbf{0},\mathbf{1}\} \qquad \forall\ \mathbf{i}\ \in\ \boldsymbol{N^I} \qquad \textbf{(11)}$$

$$\boldsymbol{S_i}\ \in \{\mathbf{0},\mathbf{1}\} \qquad \forall\ \mathbf{i}\ \in\ \boldsymbol{N^I} \qquad \textbf{(12)}$$

Chapter 4.0 Model Implementation and Results

In this chapter, the model is implemented in two networks. First a network with 16 nodes for simplifications, results will be discussed to determine communities and the spread of influence in each community. following, influencers proportion in each community will be calculated to find the most vulnerable community. As a result, a community resilient will submerge. As a following step, the same model is applied in a social network with 84 nodes representing 84 individuals, and 853 links to represent the connections between all individuals.

4.1 Sample network

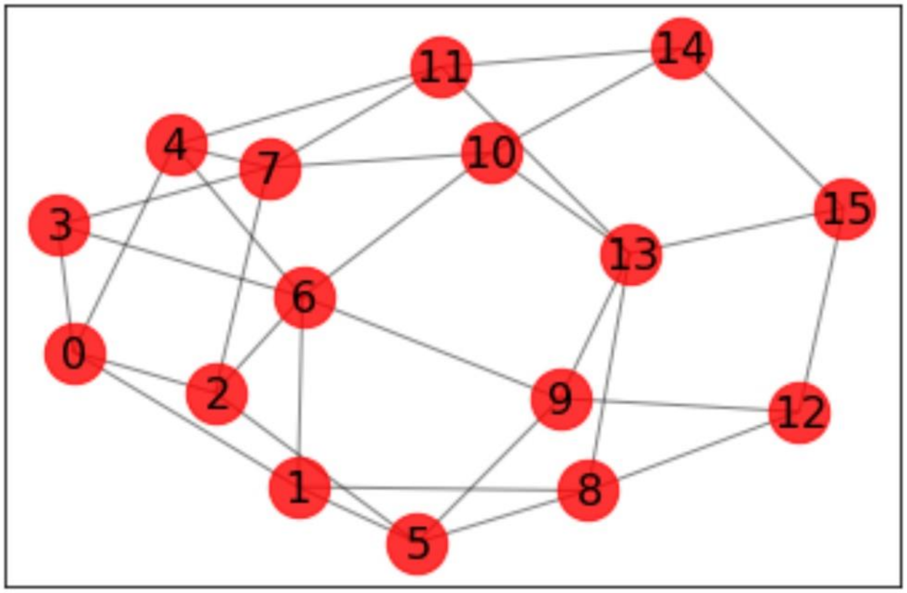


Figure 3 Sample Network

After applying the Girvan Newman Algorithm, communities were detected as shown in figure 3. The first community consists of the following nodes: 0, 1, 2, 3, 6. The second community consists of nodes: 4, 7, 11. The third community consists of nodes: 5, 8, 9, 12, 13, 15. And finally the fourth community consists of nodes 10, 14.

Community	**Nodes in the community (Individuals)**
_A_ (RED)	0, 1, 2, 3, 6
_B_ (Blue)	4, 7, 11
_C_ (Yellow)	5, 8, 9, 12, 13, 15
_D_ (Green)	10, 14

Table 1 Communities detected in Sample Network

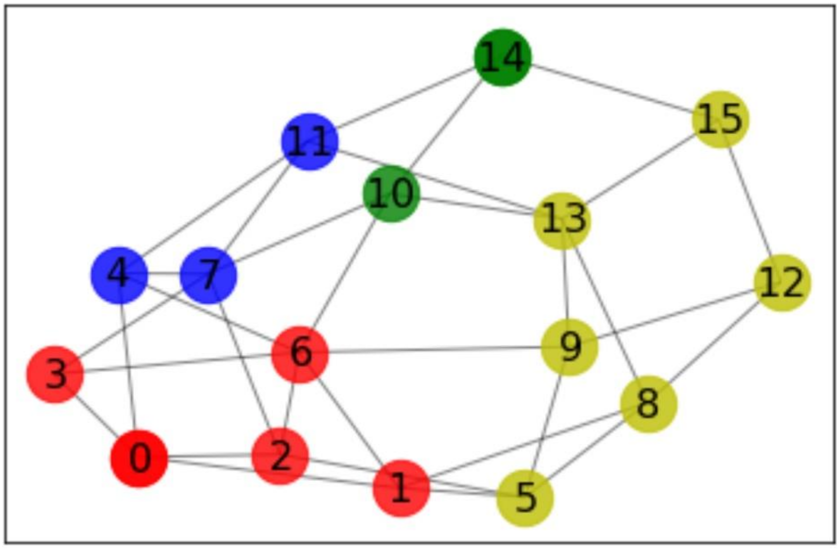


Figure 4 Communities detected in Sample Network

The model chose nodes 0, 14 as an initial influencer to start spreading influence in the network. The influenced nodes at the end were 1, 2, 3, 15.

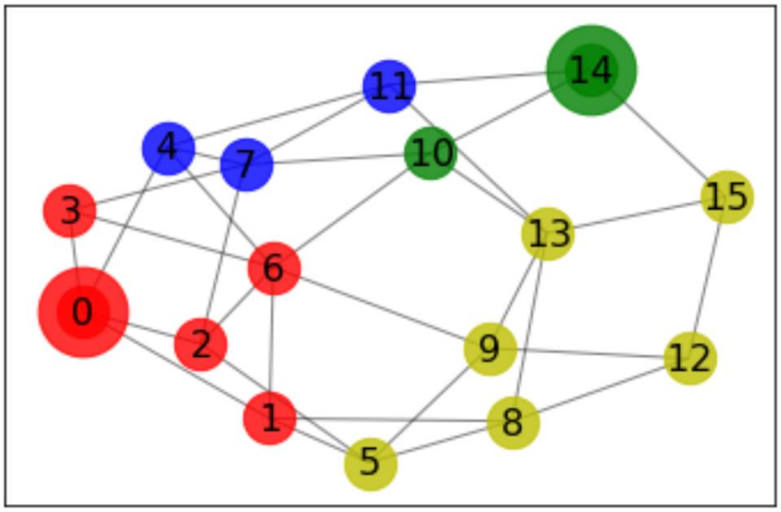


Figure 5 Initial Influencers in Sample Network

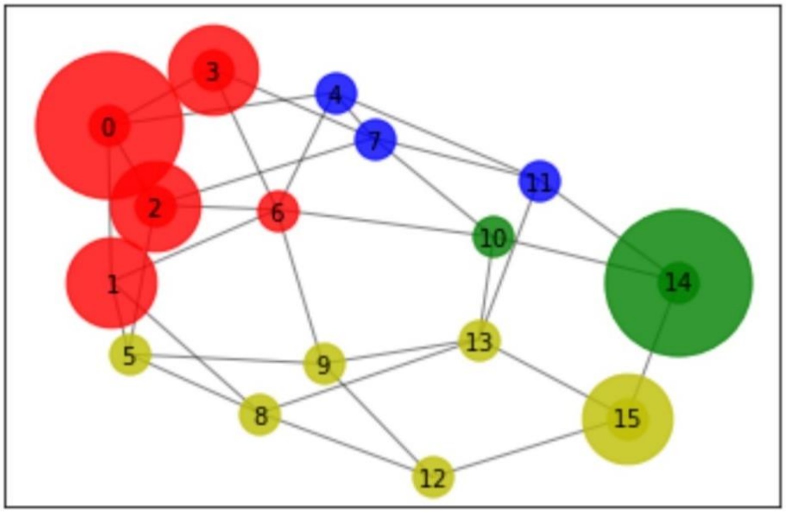


Figure 6 Optimized Influence in Sample Network

4.2 Social Network Implementation

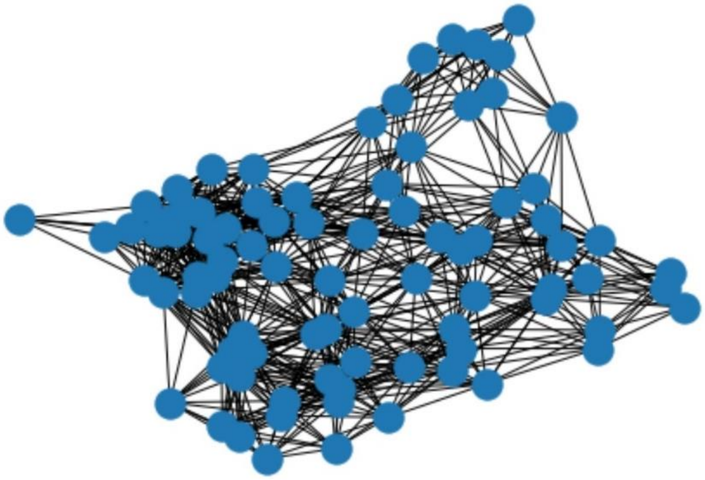


Figure 7 Social Network

After applying Girvan Newman algorithm, four communities were detected as shown in table 2.

Community	Nodes in the community (Individuals)
A (RED)	0, 3, 5, 6, 7, 8, 9, 11, 12, 20, 21, 22, 23, 30, 32, 34, 36, 37, 39, 41, 42, 43
B (Blue)	1, 2, 4, 10, 13, 14, 15, 16, 17, 18, 19, 24, 25, 26, 27, 28, 29, 31, 33, 35, 38
C (Yellow)	40, 44, 45, 46, 47, 48, 49, 50, 51, 52, 54, 55, 57, 58, 59, 60, 61, 64, 65, 66, 67, 68, 69, 71, 72, 77, 79, 80, 81, 83
D (Green)	53, 56, 62, 63, 70, 73, 74, 75, 76, 78, 82

Table 2 Detected Communities in Social Network

The model chose nodes 5, 14, 16, 44, 47 to be the initial influencers in the network. The spread of influence have reached and activated nodes 5, 6, 7, 9, 11, 14, 15, 16, 17, 18, 19, 21, 22, 24, 25, 26, 27, 28, 29, 30, 31, 33, 35, 38, 40, 44, 45, 46, 47, 48, 49, 51, 54, 55, 57, 58, 59, 60, 61, 64, 65, 66, 67, 68, 69, 71, 72.

The following step would be calculating the proportion of influencers in each community, and this would help in evaluating the vulnerability and the resilience of each community. The ability to spread information inside each community will be higher if we have more influencers inside the community.

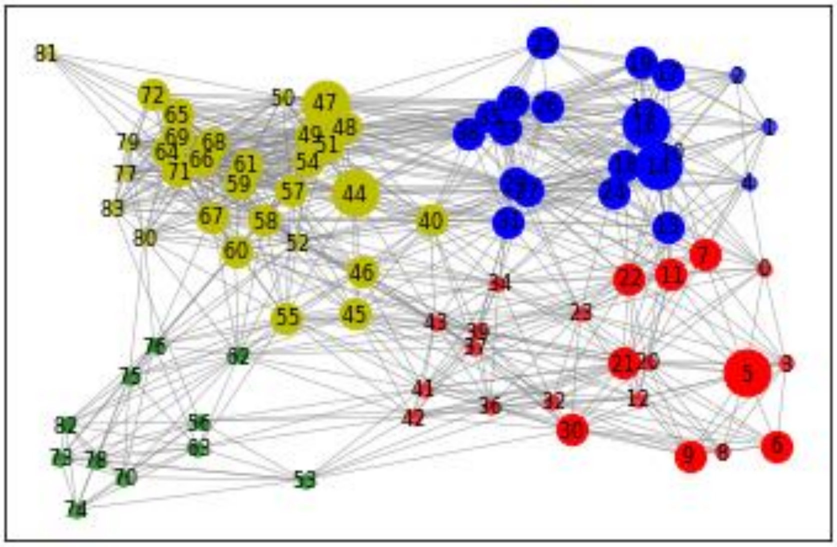


Figure 8 Total Influenced Nodes in Communities

Community	Active nodes proportion
A (RED)	0.363
B (Blue)	0.762
C (Yellow)	0.766 (Most resilient)
D (Green)	0 (Most Vulnerable)

Table 3 Community Vulnerability Proportion Ranking

As it is shown in table 3, Community D was found to be the most vulnerable community between all four with a score of 0. This means, it was difficult to spread any influence in that community which make it hard to reach individuals and spread information. The most resilient communities were B and C, with scores 0.766 and 0.762 respectively. If any disruptive event happens, and some active nodes were removed from the network, spreading information in communities B and C would be easier than communities A and D. To be prepared for any disruption to the network, vulnerable communities such as A and D should be prioritized.

4.3 Threshold Sensitivity Analysis

A Sensitivity analysis is applied to determine how sensitive the model is to changing threshold associated to each link. The threshold of the green community D is reduced to analyze if individuals will be influenced in that community.

After reducing the threshold to the green community node 53 was influenced. And the total influenced nodes were 14, 15, 16, 17, 18, 19, 21, 21, 24, 25, 26, 27, 28, 29, 30, 31, 33, 35, 38, 40, 44, 45, 46, 47, 48, 49, 51, 54, 55 , 57, 58, 59, 60, 61, 63, 65, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 78, 82.

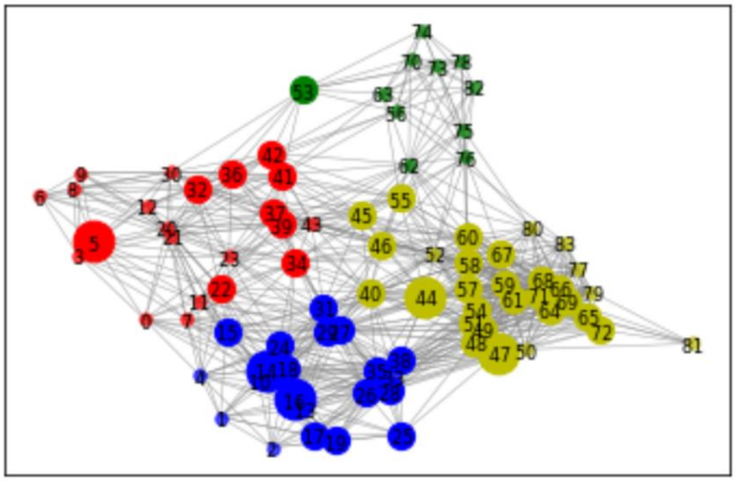


Figure 9 Total Influenced Nodes After First Change to Threshold

Influenced nodes proportion to evaluate vulnerability of each community is shown in table 10.

Community	Active nodes proportion
_A_ (RED)	0.363
_B_ (Blue)	0.762
_C_ (Yellow)	0.766 (Most resilient)
_D_ (Green)	0.09 (Most Vulnerable)

Table 4 Community Vulnerability Proportion After First Change to Threshold

Threshold associated to individuals in green community is reduced even more. The initial influencers were nodes 14, 16, 44, 47, 63, while influenced nodes were as follow: 14, 15, 16, 17, 18, 19, 22, 24, 25, 26, 27, 28, 29, 31, 33, 35, 38, 40, 44, 45, 46, 47, 48, 49, 51, 54, 55, 57, 58, 59, 60, 61, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 78, 82.

And Influenced nodes proportion to evaluate vulnerability of each community is shown in table 11.

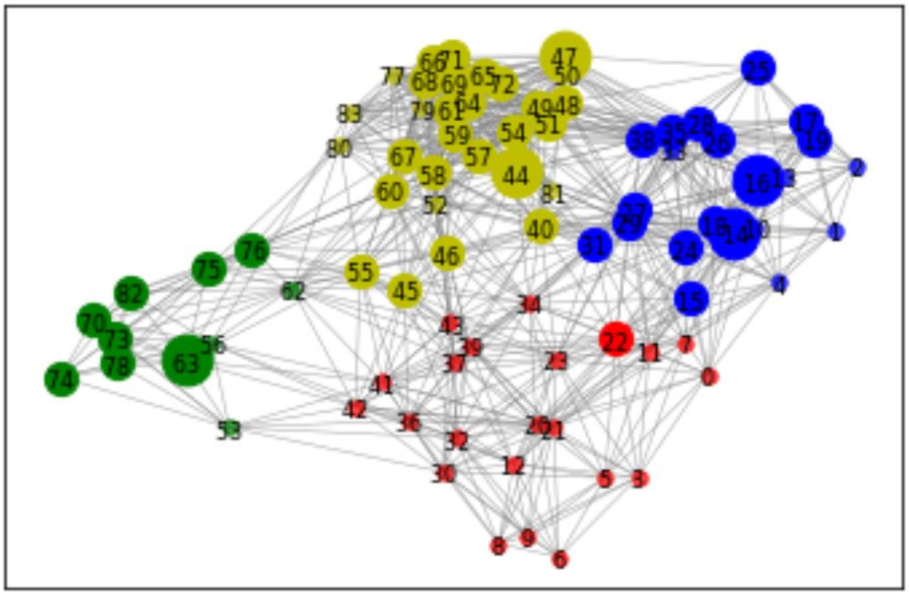


Figure 10 Total Influenced Nodes After Second Change in Threshold

Community	**Active nodes proportion**
<u>A</u> (RED)	0.045 (Most Vulnerable)
<u>B</u> (Blue)	0.762
<u>C</u> (Yellow)	0.766 (Most resilient)
<u>D</u> (Green)	0.727

Table 5 Community Vulnerability Proportion After Second Change in Threshold

To better understand the relationship between thresholds and community vulnerability, thresholds is changed 50 times in a descending order to analyze the behavior of vulnerability.

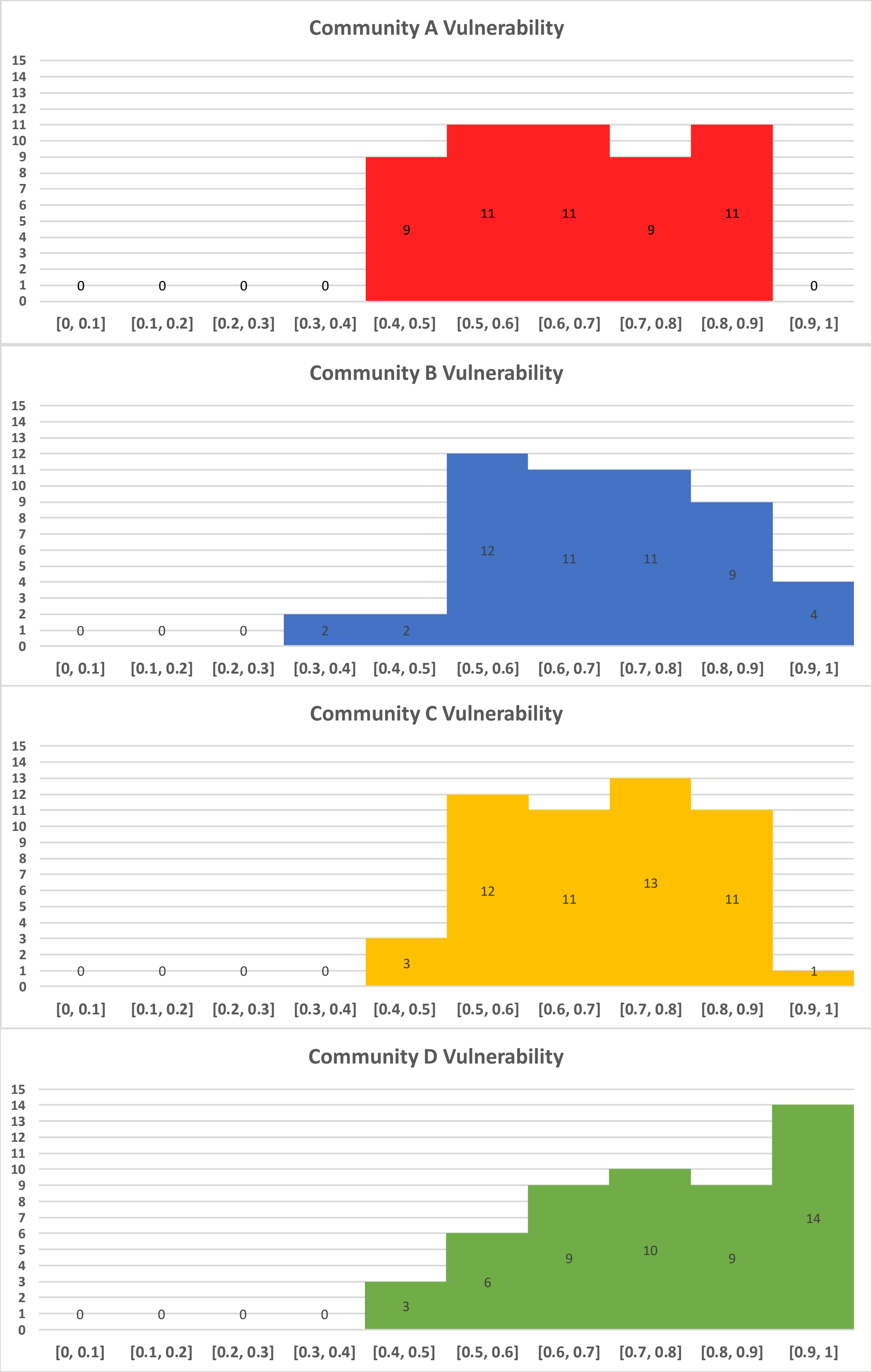


Figure 11 Communities Vulnerability Scores with 50 Decreasing Thresholds

Figures indicate that changing thresholds associated to each node will affect community vulnerability. Even though vulnerability scores of each community is changing with the change associated to thresholds, each community is affected differently. For example, community D has an increasing behavior in vulnerability scores with the increase associated to thresholds and scored between 0.7 and 1 for 33 iterations. While community A is affected differently and always scored between 0.4 and 0.82.

In table 6, a weighted score was calculated to evaluate the vulnerability of each community. Community D was found to be the most vulnerable between all four communities, and changing thresholds highly affected the level of vulnerability of that community. Meanwhile, Community A was found to be the least vulnerable. It is important to remember that in this section, vulnerability measurement was based on changing thresholds 50 times in a decreasing manner to evaluate vulnerability behavior.

	A	B	C	D
Weighted Score	0.653	0.687	0.689	0.764

Table 6 Community Vulnerability Weighted Scores

4.4 Initial Influencers Sensitivity Analysis:

The number of initial influencers is changed in the model in order to measure its effect on results. Starting with 6 nodes as initial influencers 5, 14, 16, 32, 44, 47. Influenced nodes were 5, 6, 7, 9, 11, 14, 15, 16, 17, 18, 19, 21, 22, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 37, 38, 39, 40, 41, 44, 45, 46, 47, 48, 49, 51, 53, 54, 55, 57, 58, 59, 60, 61, 64, 65, 66, 67, 68, 69, 71, 72.

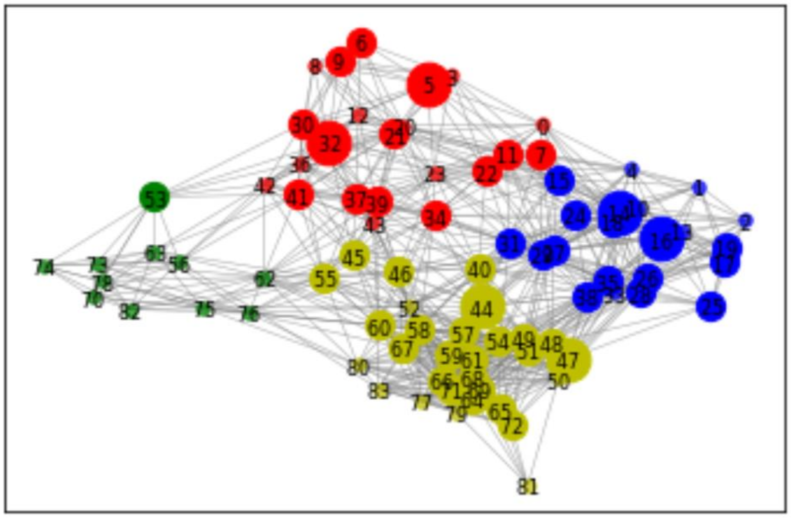


Figure 12 Total Influenced Nodes After First Change in Initial Influencers

Community	**Active nodes proportion**
<u>*A*</u> (RED)	0.6
<u>*B*</u> (Blue)	0.762
<u>*C*</u> (Yellow)	0.766 (Most resilient)
<u>*D*</u> (Green)	0.09 (Most Vulnerable)

Table 7 Community Vulnerability Proportion Ranking After First Change in Initial Influencers

After changing the number of influencers to 7 nodes, initial influencers were 5, 14, 16, 32, 44, 47, 70. And influenced nodes were, 5, 6, 7, 9, 11, 14, 15, 16, 17, 18, 19, 21, 22, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 37, 38, 39, 40, 41, 44, 45, 46, 47, 48, 49, 51, 53, 54, 55, 57, 58, 59, 60, 61, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 76, 78.

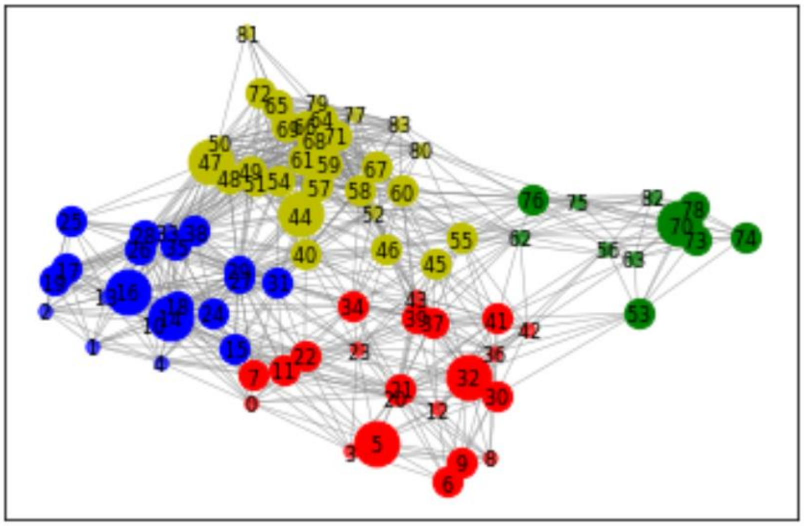


Figure 13 Total Influenced Nodes After Second Change in Initial Influencers

Community	Active nodes proportion
A (RED)	0.59
B (Blue)	0.762
C (Yellow)	0.767 (Most resilient)
D (Green)	0.55 (Most Vulnerable)

Table 8 Community Vulnerability Proportion Ranking After Second Change in Initial Influencers

For more detailed analysis, initial influencers were changed between 1 and 10 to analyze how vulnerability of each community were affected.

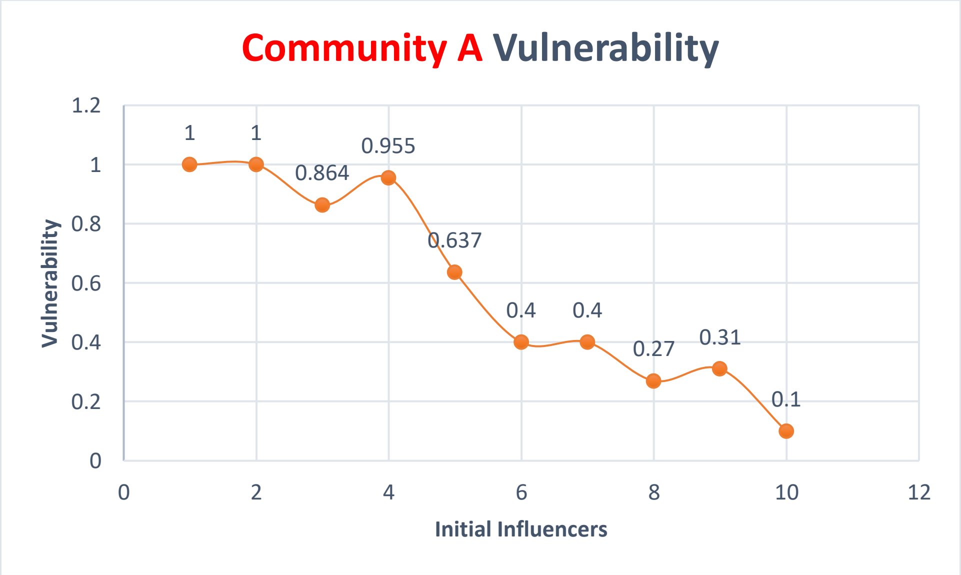


Figure 14 Community A Sensitivity Analysis to Changing the Number of Initial Influencers

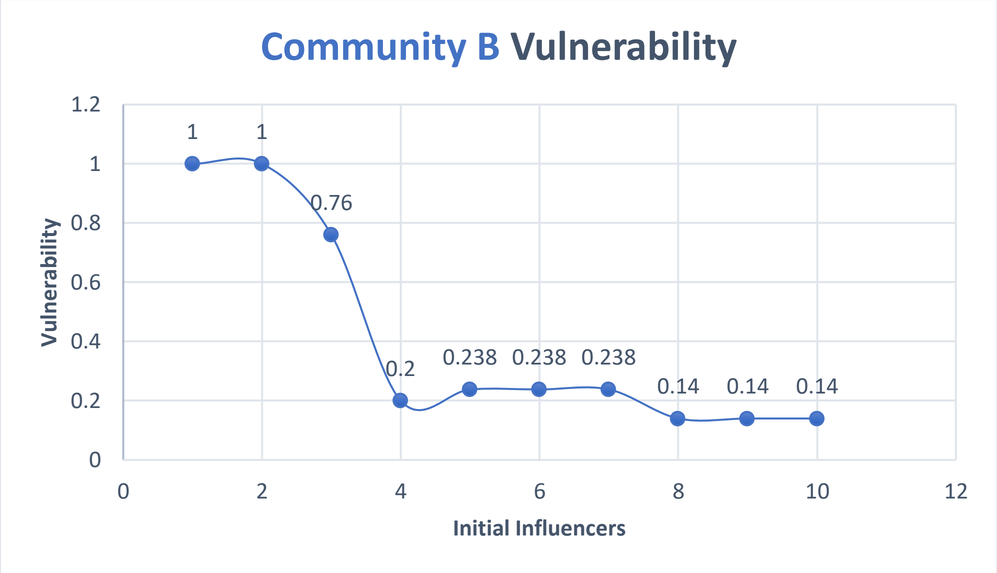


Figure 15 Community B Sensitivity Analysis to Changing the Number of Initial Influencers

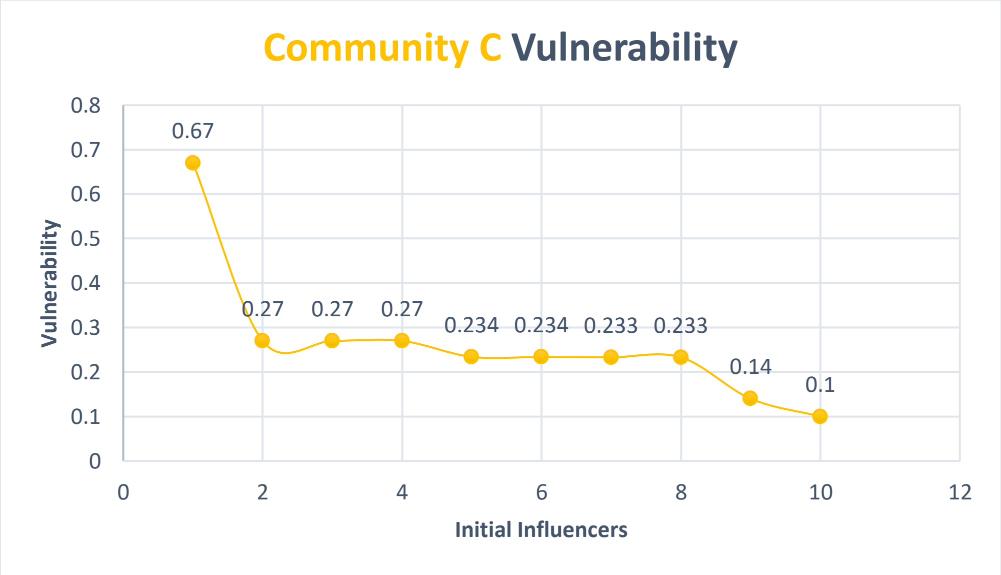


Figure 16 Community C Sensitivity Analysis to Changing the Number of Initial Influencers

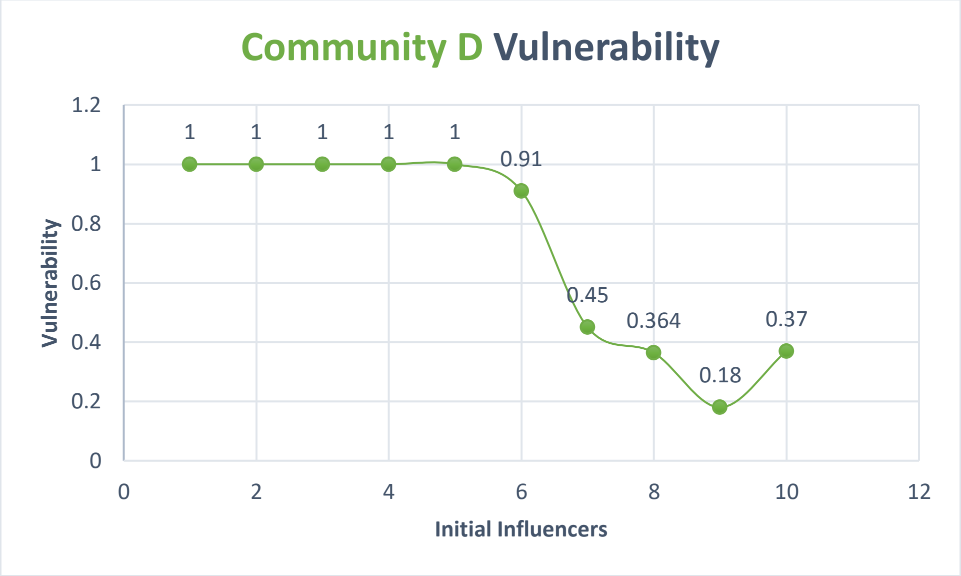


Figure 17 Community D Sensitivity Analysis to Changing the Number of Initial Influencers

We can observe from figures 14, 15, 16, and 17 that vulnerability level of each community is affected by the number of initial influencers trying to spread influence in the network. Since the linear threshold model applies the concept of multiple exposures from different sources is required for an individual to change behavior, we notice that the higher the number of initial influencers, the less vulnerable the community will be. Which means, more people were influenced in each community.

Since community vulnerability can be affected from other factors, such as the number of connected nodes to a specific node, threshold associated to the node, and number of active neighbors trying to spread influence. We can observe that in some cases, although the number of initial influencers increased, vulnerability might not be affected as much. For example, community D vulnerability was reduced only after increasing the number of initial influencers from 1 to 6, while community C only required 2 influencers to reduce vulnerability from 0.67 to 0.27.

After applying the sensitivity analysis to each community, sensitivity analysis is also applied to the whole network to understand the relationship between the number of initial influencers and network vulnerability. We can observe the relationship between initial influencers and network vulnerability from figure 18.

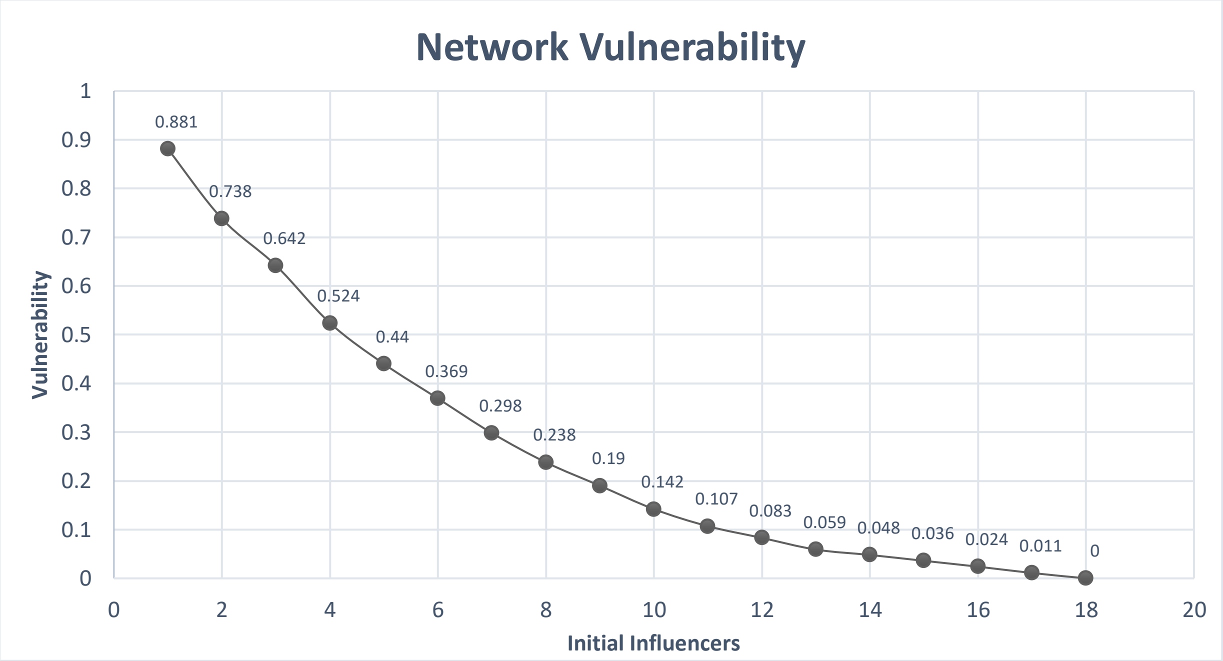


Figure 18 Sensitivity Analysis to Network Vulnerability and Initial Influencers

Chapter 5 Discussion

The model applied in this study was able to detect the vulnerability level of each community in the network. This model will help in investigating community structures in the network, while trying to maximize the spread of influence and detect influencers.

This model can be improved by applying it in a larger social network. Another way to improve the model is to represent a disruptive event by removing influencers in each community and reapplying the model to maximize the spread of influence in a disruptive network. Another way to improve the work is by giving vulnerability scores to each node based on the results of the model and change the objective function to include minimizing vulnerability in each community.

Detecting communities is considered a key element to better understand social networks. This detection will allow researchers to determine influencers inside each community, which in terms will help in evaluating community vulnerability. As a result of evaluating influence vulnerability, vulnerable communities can be prioritized for optimizing the spread of influence within the community to be prepared for any disruptive event.

Many methods can be applied to social networks which will help in analyzing the spread of information and influence in the network. These methods try to find influencers and detects how each influencer tries to influence connected individuals. Combining community detection methods with influence optimization methods will give us a deep understanding on how to accurately prepare communities to be more resilient from a social influence and information spreading perspective.

Disruptive events, such as natural disasters can be very damaging to societies. By utilizing technology and research tools to study social networks before any disruptive event occurs, this will be very beneficial in protecting communities.

References

Cao, T., Wu, X., Wang, S., & Hu, X. (2011). Maximizing influence spread in modular social networks by optimal resource allocation. *Expert Systems with Applications, 38*(10), 13128-13135.

Chen, W., Lakshmanan, L. V., & Castillo, C. (2013). Information and influence propagation in social networks. *Synthesis Lectures on Data Management, 5*(4), 1-177.

Dao, V.-L., Bothorel, C., & Lenca, P. (2018). Community structure: A comparative evaluation of community detection methods. *arXiv preprint arXiv:1812.06598*.

Girvan, M., & Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the national academy of sciences, 99*(12), 7821-7826.

Granovetter, M. (1978). Threshold models of collective behavior. *American journal of sociology, 83*(6), 1420-1443.

Kempe, D., Kleinberg, J., & Tardos, É. (2003). *Maximizing the spread of influence through a social network.* Paper presented at the Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining.

Lancichinetti, A., & Fortunato, S. (2009). Community detection algorithms: a comparative analysis. *Physical review E, 80*(5), 056117.

Liu, W., Chen, X., Jeon, B., Chen, L., & Chen, B. (2019). Influence maximization on signed networks under independent cascade model. *Applied Intelligence, 49*(3), 912-928.

Nazemian, A., & Taghiyareh, F. (2012). *Influence maximization in independent cascade model with positive and negative word of mouth.* Paper presented at the 6th International Symposium on Telecommunications (IST).

Newman, M. E. (2004). Detecting community structure in networks. *The European Physical Journal B, 38*(2), 321-330.

Newman, M. E., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical review E, 69*(2), 026113.

Rocco, C. M., Barker, K., Moronta, J., & Ramirez-Marquez, J. E. (2018). Community detection and resilience in multi-source, multi-terminal networks. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, 232*(6), 616-626.

Shen, H.-W. (2013). *Community Structure of Complex Networks*: Springer.

Wang, W., & Street, W. N. (2018). Modeling and maximizing influence diffusion in social networks for viral marketing. *Applied network science, 3*(1), 6.

Wasserman, S., & Faust, K. (1994). *Social network analysis: Methods and applications* (Vol. 8): Cambridge university press.

Yeagle, L. (2018). Measuring and Optimizing Influence for Resilient Community Networks.

Zeng, Y., Chen, X., Cong, G., Qin, S., Tang, J., & Xiang, Y. (2016). Maximizing influence under influence loss constraint in social networks. *Expert Systems with Applications, 55*, 255-267.

Appendix A

AMPL Code:

```
reset;
option solver cplex;
option randseed 19;

# Groups of people, influencers, and connections between them
set PEOPLE;
set INFLUENCERS;
set ARCS within {INFLUENCERS, PEOPLE};

# Parameters
#param prop {ARCS} := Uniform01(); # Assign propagation probability to the arcs
param prop{ARCS};      # Assign propagation probability to the arcs
param t{PEOPLE};       # Threshold for each person
param M := 1000;       # For finding maximum
#param sv{PEOPLE};      # Social vulnerability for each person

# Variables
var y{INFLUENCERS} binary;     # Starting influencers
var s{PEOPLE} binary;          # Ending influenced people
var u{PEOPLE};                 # Hold summation of arcs
var x{PEOPLE};                 # Used to find maximum between start and end
var b{PEOPLE} binary;          # Used to find maximum

# Optimize spread: Influenced node * social vulnerability score
maximize spread: sum{j in PEOPLE} x[j];

# Find sum of arcs of active nodes
subject to active {j in PEOPLE}: u[j] = sum{(i,j) in ARCS} y[i] * prop[i,j];

# Determine if node meets threshold limit
subject to threshold {j in PEOPLE}: t[j]*s[j] <= u[j];

# Determine number of starting influencers
subject to limit: sum {i in INFLUENCERS} y[i] <= 5;

# Constraints to find the ending maximum influenced
subject to max1 {j in PEOPLE}: x[j] >= s[j];
subject to max2 {j in PEOPLE}: x[j] >= y[j];
subject to max3 {j in PEOPLE}: x[j] <= s[j] + M*b[j];
subject to max4 {j in PEOPLE}: x[j] <= y[j] + M*(1-b[j]);

# Read in data and solve
data SocialTryyy.dat;
solve;
display y;    # Starting influencers
display x;    # Maximum influencers
display s;    # Influenced people
display b;
```

Appendix B

Python Code:

```
from gurobipy import *
import networkx as nx
import random as ran

seed = 40
rani = ran.Random(seed)

edgelist = [(0, 3),(0, 4),(0, 5),(0, 7),(0, 10),(0, 11),(0, 14),(0, 15),(0, 18),(0, 20),(0, 21),(0, 22),(0, 23),(0, 24),
    (1, 2),(1, 4),(1, 10),(1, 13),(1, 14),(1, 15),(1, 16),(1, 17),(1, 18),(1, 19),(1, 24),(1, 26),(2, 4),(2, 10),
    (2, 13),(2, 14),(2, 16),(2, 17),(2, 18),(2, 19),(2, 25),(2, 26),(2, 28),(3, 5),(3, 6),(3, 7),(3, 8),(3, 9),(3, 11),
    (3, 12),(3, 20),(3, 21),(3, 22),(3, 23),(4, 7),(4, 10),(4, 11),(4, 13),(4, 14),(4, 15),(4, 16),(4, 18),(4, 22),
    (4, 24),(4, 27),(4, 29),(5, 6),(5, 7),(5, 8),(5, 9),(5, 11),(5, 12),(5, 20),(5, 21),(5, 22),(5, 23),(5, 30),(6, 8),
    (6, 9),(6, 12),(6, 20),(6, 21),(6, 30),(6, 32),(7, 10),(7, 11),(7, 14),(7, 15),(7, 1),(7, 20),(7, 21),(7, 22),(7, 23),
    (7, 24),(7, 27),(7, 29),(7, 31),(7, 34),(8, 9),(8, 12),(8, 20),(8, 21),(8, 30),(8, 32),(8, 36),(9, 12),(9, 20),(9, 21),
    (9, 30),(9, 32),(9, 36),(10, 11),(10, 13),(10, 14),(10, 15),(10, 16),(10, 17),(10, 18),(10, 19),(10, 22),(10, 24),
    (10, 26),(10, 27),(10, 28),(10, 29),(10, 31),(10, 33),(10, 35),(10, 38),(11, 12),(11, 14),(11, 15),(11, 18),(11, 20),
    (11, 21),(11, 22),(11, 23),(11, 24),(11, 27),(11, 29),(11, 31),(11, 34),(11, 39),(12, 20),(12, 21),(12, 22),(12, 23),
    (12, 30),(12, 32),(12, 36),(12, 37),(12, 39),(12, 41),(12, 42),(13, 14),(13, 15),(13, 16),(13, 17),(13, 18),(13, 19),
    (13, 24),(13, 25),(13, 26),(13, 27),(13, 28),(13, 29),(13, 33),(13, 35),(13, 38),(14, 15),(14, 16),(14, 17),(14, 18),
    (14, 19),(14, 22),(14, 24),(14, 26),(14, 27),(14, 28),(14, 29),(14, 31),(14, 33),(14, 35),(14, 38),(14, 40),(15, 16),
    (15, 18),(15, 20),(15, 21),(15, 22),(15, 23),(15, 24),(15, 27),(15, 29),(15, 31),(15, 34),(15, 40),(16, 17),(16, 18),
    (16, 19),(16, 24),(16, 25),(16, 26),(16, 27),(16, 28),(16, 29),(16, 31),(16, 33),(16, 35),(16, 38),(17, 18),(17, 19),
    (17, 25),(17, 26),(17, 28),(17, 33),(17, 35),(17, 38),(18, 19),(18, 22),(18, 24),(18, 26),(18, 27),(18, 28),(18, 29),
    (18, 31),(18, 33),(18, 34),(18, 35),(18, 38),(18, 40),(19, 25),(19, 26),(19, 28),(19, 33),(19, 35),(19, 38),(20, 21),
    (20, 22),(20, 23),(20, 30),(20, 32),(20, 34),(20, 36),(20, 37),(20, 39),(20, 41),(20, 42),(20, 43),(21, 22),(21, 23),
    (21, 30),(21, 32),(21, 34),(21, 36),(21, 37),(21, 39),(21, 41),(21, 42),(21, 43),(22, 23),(22, 24),(22, 27),(22, 29),
    (22, 31),(22, 32),(22, 34),(22, 37),(22, 39),(22, 40),(22, 43),(23, 24),(23, 27),(23, 29),(23, 30),(23, 31),(23, 32),
    (23, 34),(23, 36),(23, 37),(23, 39),(23, 40),(23, 41),(23, 43),(23, 45),(23, 46),(24, 26),(24, 27),(24, 28),(24, 29),
    (24, 31),(24, 33),(24, 34),(24, 35),(24, 38),(24, 40),(24, 44),(25, 26),(25, 28),(25, 33),(25, 35),(25, 38),(25, 47),
    (25, 48),(26, 27),(26, 28),(26, 29),(26, 31),(26, 33),(26, 35),(26, 38),(26, 47),(26, 48),(26, 49),(26, 50),(26, 51),
    (27, 28),(27, 29),(27, 31),(27, 33),(27, 34),(27, 35),(27, 38),(27, 40),(27, 44),(27, 46),(27, 48),(27, 49),(27, 51),
```

(27, 52),(27, 54),(27, 57),(28, 29),(28, 31),(28, 33),(28, 35),(28, 38),(28, 47),(28, 48),(28, 49),(28, 50),(28, 51),
(28, 54),(29, 31),(29, 33),(29, 34),(29, 35),(29, 38),(29, 40),(29, 44),(29, 46),(29, 48),(29, 49),(29, 51),(29, 52),
(29, 54),(29, 57),(30, 32),(30, 36),(30, 37),(30, 39),(30, 41),(30, 42),(30, 43),(30, 53),(31, 33),(31, 34),(31, 35),
(31, 37),(31, 38),(31, 39),(31, 40),(31, 43),(31, 44),(31, 45),(31, 46),(31, 52),(31, 54),(31, 57),(31, 58),(32, 34),
(32, 36),(32, 37),(32, 39),(32, 41),(32, 42),(32, 43),(32, 45),(32, 53),(33, 35),(33, 38),(33, 40),(33, 44),(33, 47),
(33, 48),(33, 49),(33, 50),(33, 51),(33, 54),(34, 36),(34, 37),(34, 39),(34, 40),(34, 41),(34, 43),(34, 44),(34, 45),
(34, 46),(34, 52),(34, 55),(34, 58),(34, 60),(35, 38),(35, 40),(35, 44),(35, 47),(35, 48),(35, 49),(35, 50),(35, 51),
(35, 54),(36, 37),(36, 39),(36, 41),(36, 42),(36, 43),(36, 45),(36, 53),(36, 55),(36, 56),(37, 39),(37, 40),(37, 41),
(37, 42),(37, 43),(37, 45),(37, 46),(37, 52),(37, 55),(37, 62),(38, 40),(38, 44),(38, 47),(38, 48),(38, 49),(38, 50),
(38, 51),(38, 54),(38, 57),(38, 59),(38, 61),(39, 40),(39, 41),(39, 42),(39, 43),(39, 45),(39, 46),(39, 52),(39, 55),
(39, 60),(39, 62),(40, 43),(40, 44),(40, 45),(40, 46),(40, 48),(40, 49),(40, 51),(40, 52),(40, 54),(40, 55),(40, 57),
(40, 58),(40, 59),(40, 60),(40, 61),(40, 67),(41, 42),(41, 43),(41, 45),(41, 46),(41, 53),(41, 55),(41, 56),(41, 62),
(41, 63),(42, 43),(42, 45),(42, 53),(42, 55),(42, 56),(42, 62),(42, 63),(43, 44),(43, 45),(43, 46),(43, 52),(43, 55),
(43, 58),(43, 60),(43, 62),(44, 45),(44, 46),(44, 47),(44, 48),(44, 49),(44, 50),(44, 51),(44, 52),(44, 54),(44, 55),
(44, 57),(44, 58),(44, 59),(44, 60),(44, 61),(44, 64),(44, 66),(44, 67),(44, 68),(44, 69),(44, 71),(45, 46),(45, 52),
(45, 55),(45, 57),(45, 58),(45, 60),(45, 62),(45, 67),(45, 75),(45, 76),(46, 52),(46, 55),(46, 57),(46, 58),(46, 59),
(46, 60),(46, 61),(46, 62),(46, 67),(46, 76),(47, 48),(47, 49),(47, 50),(47, 51),(47, 54),(47, 57),(47, 58),(47, 59),
(47, 61),(47, 64),(47, 65),(47, 66),(47, 68),(47, 69),(47, 71),(47, 72),(48, 49),(48, 50),(48, 51),(48, 54),(48, 57),
(48, 58),(48, 59),(48, 61),(48, 64),(48, 65),(48, 66),(48, 67),(48, 68),(48, 69),(48, 71),(48, 72),(48, 79),(49, 50),
(49, 51),(49, 52),(49, 54),(49, 57),(49, 58),(49, 59),(49, 61),(49, 64),(49, 65),(49, 66),(49, 67),(49, 68),(49, 69),
(49, 71),(49, 72),(49, 77),(49, 79),(50, 51),(50, 54),(50, 57),(50, 59),(50, 61),(50, 64),(50, 65),(50, 66),(50, 68),
(50, 69),(50, 71),(50, 72),(50, 79),(50, 81),(51, 52),(51, 54),(51, 57),(51, 58),(51, 59),(51, 61),(51, 64),(51, 65),
(51, 66),(51, 67),(51, 68),(51, 69),(51, 71),(51, 72),(51, 77),(51, 79),(52, 54),(52, 55),(52, 57),(52, 58),(52, 59),
(52, 60),(52, 61),(52, 62),(52, 64),(52, 66),(52, 67),(52, 68),(52, 69),(52, 71),(52, 75),(52, 76),(52, 77),(52, 80),
(53, 56),(53, 63),(53, 70),(53, 73),(53, 74),(53, 78),(54, 57),(54, 58),(54, 59),(54, 60),(54, 61),(54, 64),(54, 65),
(54, 66),(54, 67),(54, 68),(54, 69),(54, 71),(54, 72),(54, 77),(54, 79),(54, 80),(54, 83),(55, 57),(55, 58),(55, 59),
(55, 60),(55, 62),(55, 63),(55, 67),(55, 75),(55, 76),(55, 80),(55, 82),(56,83),(56, 70),(56, 73),(56, 74),(56, 78),
(56, 82),(57, 58),(57, 59),(57, 60),(57, 61),(57, 64),(57, 65),(57, 66),(57, 67),(57, 68),(57, 69),(57, 71),(57, 77),
(57, 79),(57, 80),(57, 83),(58, 59),(58, 60),(58, 61),(58, 62),(58, 64),(58, 65),(58, 66),(58, 67),(58, 68),(58, 69),
(58, 71),(58, 76),(58, 77),(58, 79),(58, 80),(58, 83),(59, 60),(59, 61),(59, 64),(59, 65),(59, 66),(59, 67),(59, 68),
(59, 69),(59, 71),(59, 72),(59, 77),(59, 79),(59, 80),(59, 83),(60, 61),(60, 62),(60, 64),(60, 66),(60, 67),(60, 68),
(60, 69),(60, 71),(60, 75),(60, 76),(60, 77),(60, 79),(60, 80),(60, 83),(61, 64),(61, 65),(61, 66),(61, 67),(61, 68),
(61, 69),(61, 71),(61, 72),(61, 77),(61, 79),(61, 80),(61, 83),(62, 63),(62, 67),(62, 70),(62, 73),(62, 75),(62, 76),
(62, 78),(62, 80),(62, 82),(63, 70),(63, 73),(63, 74),(63, 75),(63, 76),(63, 78),(63, 82),(64, 65),(64, 66),(64, 67),
(64, 68),(64, 69),(64, 71),(64, 72),(64, 77),(64, 79),(64, 80),(64, 81),(64, 83),(65, 66),(65, 67),(65, 68),(65, 69),

```
    (65, 71),(65, 72),(65, 77),(65, 79),(65, 81),(65, 83),(66, 67),(66, 68),(66, 69),(66, 71),(66, 72),(66, 77),(66, 79),
    (66, 80),(66, 83),(67, 68),(67, 69),(67, 71),(67, 76),(67, 77),(67, 79),(67, 80),(67, 83),(68, 69),(68, 71),(68, 72),
    (68, 77),(68, 79),(68, 80),(68, 83),(69, 71),(69, 72),(69, 77),(69, 79),(69, 80),(69, 81),(69, 83),(70, 73),(70, 74),
    (70, 75),(70, 76),(70, 78),(70, 82),(71, 72),(71, 77),(71, 79),(71, 80),(71, 83),(72, 77),(72, 79),(72, 81),(72, 83),
    (73, 74),(73, 75),(73, 76),(73, 78),(73, 82),(74, 75),(74, 78),(74, 82),(75, 76),(75, 78),(75, 80),(75, 82),(76, 78),
    (76, 80),(76, 82),(77, 79),(77, 80),(77, 83),(78, 82),(79, 80),(79, 81),(79, 83),(80, 83)]


print(len(edgelist))



commun = {1: [0, 3, 5, 6, 7, 8, 9, 11, 12, 20, 21, 22, 23, 30, 32, 34, 36, 37, 39, 41, 42, 43],
          2: [1, 2, 4, 10, 13, 14, 15, 16, 17, 18, 19, 24, 25, 26, 27, 28, 29, 31, 33, 35, 38],
          3: [40, 44, 45, 46, 47, 48, 49, 50, 51, 52, 54, 55, 57, 58, 59, 60, 61, 64, 65, 66,
              67, 68, 69, 71, 72, 77, 79, 80, 81, 83],
          4: [53, 56, 62, 63, 70, 73, 74, 75, 76, 78, 82]}



G = nx.Graph()                                   # Creating the Graph base
G.add_edges_from(list(i) for i in edgelist)               # Imputing the edges into the graph

nodelist = list(G.nodes)
edgelist = list(G.edges)



arcprob = dict()


for i in nodelist:
    dum = list()
    for j in nodelist:
        if (j, i) in edgelist:
            dum.append((j, i))

    probranddum = list()
    for k in dum:
        probranddum.append(rani.random()*0.09)

    if sum(probranddum) >= 1:
        chacha = list()
        for l in probranddum:
            chacha.append(l/sum(probranddum))

        for n in dum:
            arcprob[n] = chacha[dum.index(n)]

    for n in dum:
        arcprob[n] = probranddum[dum.index(n)]

chnum = 10
pace = 1/chnum

mu = 1

for index in range(0, chnum + 1):



    tresh = dict()
    count = 0
    while len([* tresh]) <= len(nodelist):
        dumrand = rani.gauss(mu, 0.2)
        if 0 <= dumrand <= 1:
            tresh[count] = round(dumrand, 4)
            count += 1




    M = 1000

    mm = Model("Sensitivity_Analysis")
```

```
mm.setParam('OutputFlag', 0)

x = mm.addVars(nodelist, vtype=GRB.INTEGER)
y = mm.addVars(nodelist, vtype=GRB.BINARY, name="y")
s = mm.addVars(nodelist, vtype=GRB.BINARY, name="s")
u = mm.addVars(nodelist, name="u")
b = mm.addVars(nodelist, vtype=GRB.BINARY, name="b")

obj = quicksum(x[j] for j in nodelist)

for j in nodelist:
    dumhead = list()
    dumtail = list()
    dumtot = list()
    for i in nodelist:
        if (i, j) in edgelist:
            dumhead.append(i)
            dumtail.append(j)
            dumtot.append((i, j))

    # print(dumtot(i,j))

    mm.addConstr(u[j] == quicksum(arcprob[(l, k)] * y[l] for l in dumhead for k in dumtail))

mm.addConstrs(tresh[j]*s[j] <= u[j] for j in nodelist)

mm.addConstr(quicksum(y[i] for i in nodelist) <= 7)

mm.addConstrs(x[j] >= s[j] for j in nodelist)

mm.addConstrs(x[j] >= y[j] for j in nodelist)

mm.addConstrs(x[j] <= s[j] + M * b[j] for j in nodelist)

mm.addConstrs(x[j] <= y[j] + M*(1-b[j]) for j in nodelist)
mm.setObjective(obj, GRB.MAXIMIZE)
mm.optimize()
mm.write('ahmad.lp')



vardict = dict()
for v in mm.getVars():
    if v.varName[0] == "C":
        vardict[int(v.varName[1:])] = round(v.x)

a = 0
b = 0
c = 0
d = 0

for j in vardict:


    if j in commun[1]:
        a += vardict[j]


    elif j in commun[2]:
        b += vardict[j]


    elif j in commun[3]:
        c += vardict[j]


    elif j in commun[4]:
        d += vardict[j]

va = round(1 - a/len(commun[1]), 3)
vb = round(1 - b/len(commun[2]), 3)
vc = round(1 - c/len(commun[3]), 3)
vd = round(1 - d/len(commun[4]), 3)
```

```
# print(a, b, c, d)
# print(va, vb, vc, vd)
print("Mu = ,", round(mu, 2), va, vb, vc, vd)
# print("Mu = ,", round(mu, 2), ",Total number of influenced --->", int(mm.objVal))

mu -= pace
```