CSC 591 Graph Data Mining (GDM): Theory, Algorithms, and Applications

Project 5 Report- Virus Propagation: July 16, 2017

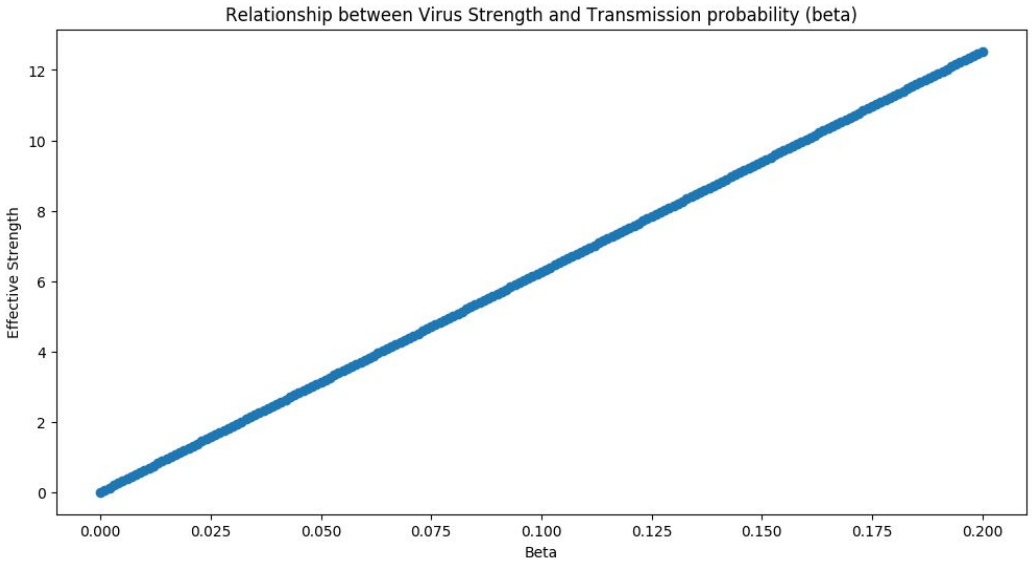
Project Members:

1. Kriti Shrivastava (Unity ID: kshriva)
2. Manjusha Trilochan Awasthi (Unity ID: mawasth)
3. Rachit Thirani (Unity ID: rthiran)

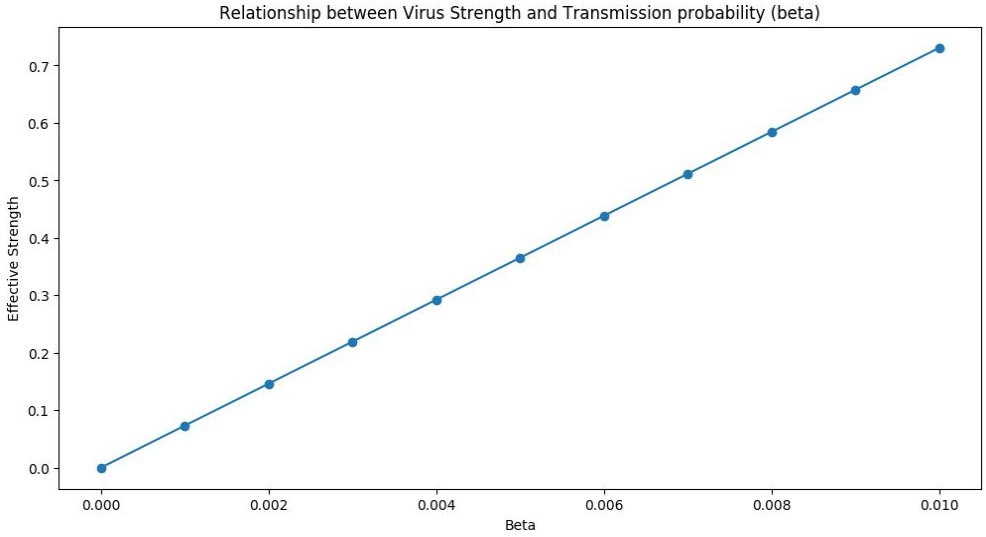
Option 1: Virus Propagation on Static Networks

Part 1. For the SIS (susceptible, infected, susceptible) Virus Propagation Model (VPM), with transmission probability $\beta = \beta 1$, and healing probability $\delta = \delta 1$, calculate the effective strength (s) of the virus on the static contact network provided (static.network). See supplementary material provided for details on the SIS VPM and on how to calculate the effective strength of a virus.
Answer the following questions:
a. Will the infection spread across the network (i.e., result on an epidemic), or will it die quickly?
Answer: For $\beta = \beta 1$ (0.20) and $\delta = \delta 1$ (0.70):
Effective strength: 12.5299130745
Since Effective strength is greater than 1, the *infection will result in an epidemic.*
For $\beta = \beta 2$ (0.01) and $\delta = \delta 2$ (0.60):
Effective strength: 0.730911596012
Since Effective strength is less than 1, the *virus will die quickly.*

b. Keeping δ fixed, analyze how the value of β affects the effective strength of the virus (suggestion: plot your results). What is the minimum transmission probability (β) that results in a Networkwide epidemic?
Answer: For $\beta = \beta 1$ (0.20) and $\delta = \delta 1$ (0.70):
Minimum transmission probability for epidemic: 0.0159618026726

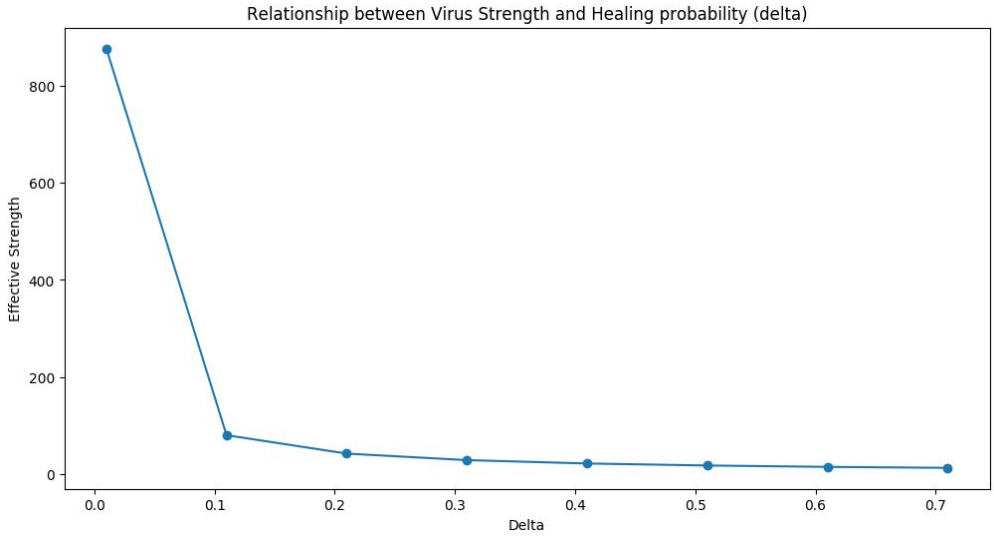


For $\beta = \beta 2$ (0.01) and $\delta = \delta 2$ (0.60):
Minimum transmission probability for epidemic: 0.0136815451479

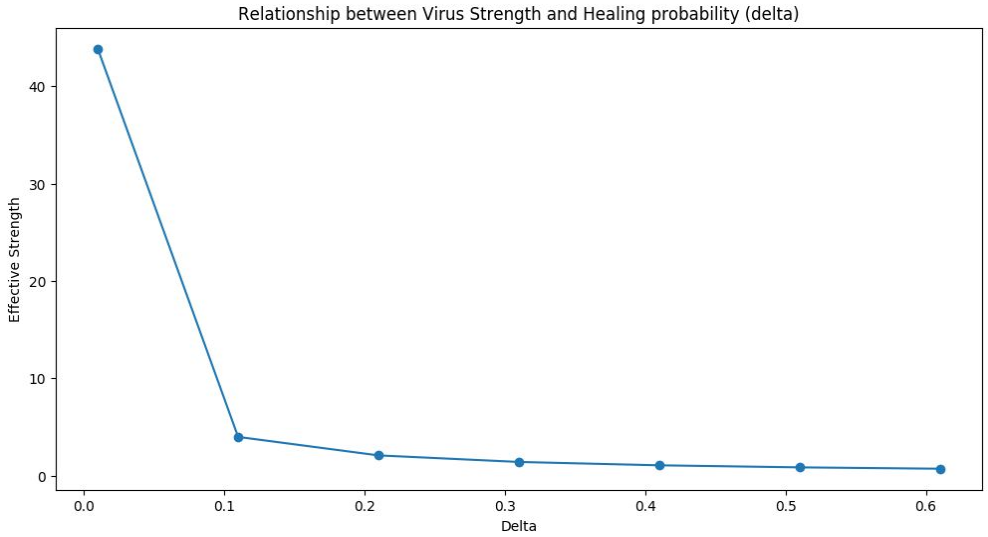


c. Keeping β fixed, analyze how the value of δ affects the effective strength of the virus (suggestion: plot your results). What is the maximum healing probability (δ) that results in a network wide epidemic?

Answer: For $\beta = \beta 1$ (0.20) and $\delta = \delta 1$ (0.70):
Maximum healing probability for epidemic: 8.77093915215



For $\beta = \beta 2$ (0.01) and $\delta = \delta 2$ (0.60): Maximum healing probability for epidemic: 0.438546957607



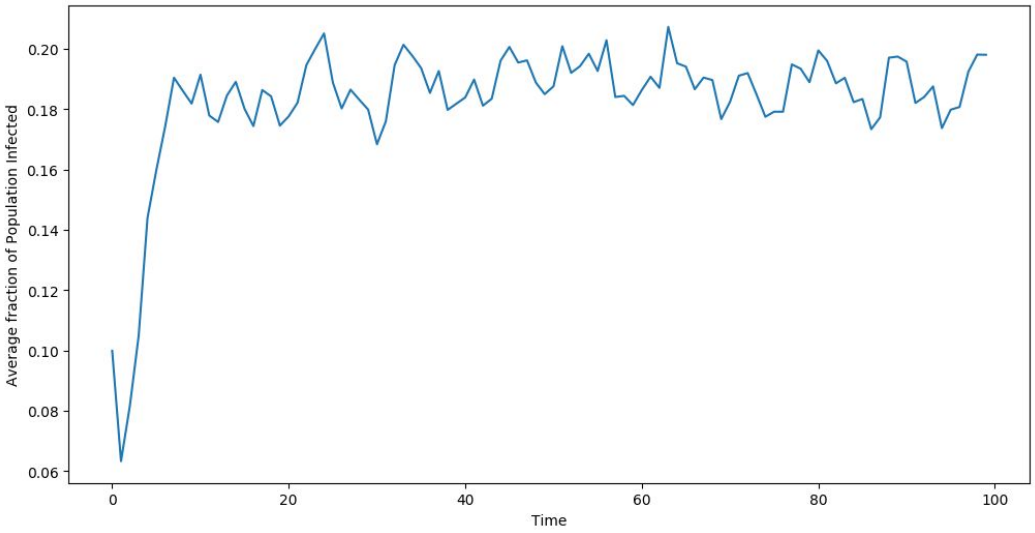
Part 2. Write a program that simulates the propagation of a virus with the SIS VPM, given a static contact network, a transmission probability (β), a healing probability (δ), a number of initially infected nodes (c), and a number of time steps to run the simulation (t). The initially infected nodes should be chosen from a random uniform probability distribution. At each time step, every susceptible (i.e., non-infected) node has a β probability of being infected by neighboring infected nodes, and every infected node has a δ probability of healing and becoming susceptible again. Your program should also calculate the fraction of infected nodes at each time step.

a. Run the simulation program 10 times for the static contact network provided (static.network), with $\beta = \beta 1$, $\delta = \delta 1$, c = n/10 (n is the number of nodes in the network), and t = 100.

b. Plot the average (over the 10 simulations) fraction of infected nodes at each time step. Did the infection spread across the network, or did it die quickly? Do the results of the simulation agree with your conclusions in (1a)?

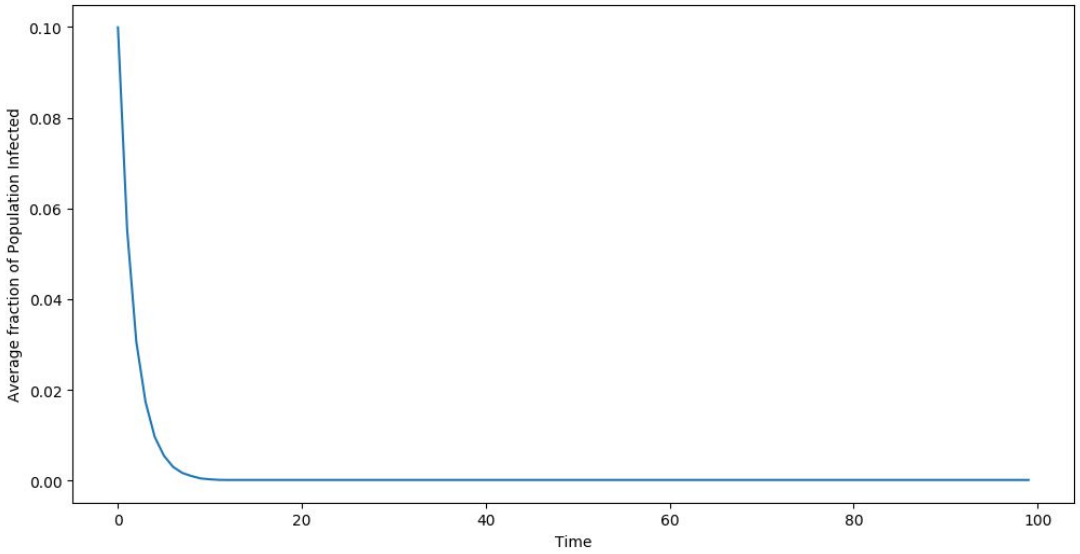
Answer:

For $\beta = \beta 1$ (0.20) and $\delta = \delta 1$ (0.70):



The infection resulted in an epidemic. The result of the simulation is in sync with the conclusion in (1a).

For $\beta = \beta 2$ (0.01) and $\delta = \delta 2$ (0.60):



The infection died quickly and there was no epidemic. The result of the simulation is in sync with the previous conclusion in (1a).

Part 3. Write a program that implements an immunization policy to prevent the virus from spreading across the network. Given a number of available vaccines (k) and a contact network, your program should select k nodes to immunize. The immunized nodes (and their incident edges) are then removed from the contact network.

a. What do you think would be the optimal immunization policy? What would be its time complexity? Would it be reasonable to implement this policy? Justify.

Answer: The optimal immunization policy will be finding the subset of k nodes among all n choose k possible subsets that result in the largest drop in the maximum eigen value. The time complexity of this solution is of polynomial order. Since this solution is computationally intractable it won't be possible to implement this policy.

For your program, use the following heuristic immunization policies:

Policy A: Select k random nodes for immunization.

Policy B: Select the k nodes with highest degree for immunization.

Policy C: Select the node with the highest degree for immunization. Remove this node (and its incident edges) from the contact network. Repeat until all vaccines are administered.

Policy D: Find the eigenvector corresponding to the largest eigenvalue of the contact network's adjacency matrix. Find the k largest (absolute) values in the eigenvector. Select the k nodes at the corresponding positions in the eigenvector.

For each heuristic immunization policy (A, B, C, and D) and for the static contact network provided (static.network), answer the following questions:

b. What do you think is the intuition behind this heuristic?

Answer:

- *Policy A*: This is a random approach where the intuition is to immunize random nodes in the network and expect that immunizing these nodes will prevent the virus from spreading across the network.
- *Policy B*: This is a better approach than policy A and aims to control the virus spread by targeting the nodes with the highest degree rather than selecting random nodes. The intuition here is that the highest degree nodes pose more threat as they have the highest connectivity in the network, so immunizing these will stop the virus from spreading in the network.
- *Policy C*: This is a better approach than policy B since it recalculates the node to be immunized instead of determining all nodes to be immunized in the beginning. The intuition here is that after immunizing and removing the highest degree node from the network, the degree of its neighbours will change and can also change the node with the next highest degree.
- *Policy D*: This is the best among the given policies and utilizes the largest eigen value to decide the nodes to be immunized. The intuition here is that as the largest eigen value plays a key role in determining the virus prevalence. So removing the k nodes that correspond to k largest values in eigen vector will reduce the strength of the virus and prevent it from spreading across the network.

c. Write a pseudocode for this heuristic immunization policy. What is its time complexity?

Answer:

- *Policy A*:

```
def immunizeUsingPolicyA(graph, k):
    # Immunize and remove random k nodes from the network
    1. kRandomNodes = random.sample(range(0, nx.number_of_nodes(graph)), k)
    2. graph.remove_nodes_from(kRandomNodes)
    3. return graph
```

Worst Case Time complexity: **$O(kN^2)$**

Explanation: In policy A, we randomly immunize and remove k nodes from the network.
Removing a node from the graph (adjacency matrix representation) $\rightarrow O(N^2)$.
Hence, removing k nodes and their neighbours $\rightarrow O(k*(N^2)) \rightarrow O(kN^2)$

- *Policy B:*

```
def immunizeUsingPolicyB(graph, k):
    # Immunize and remove k highest degree nodes from the network
    1. kHighestDegreeNodes = [node for node,degree in
       sorted(graph.degree_iter(),key=itemgetter(1),reverse=True)][:k]
    2. graph.remove_nodes_from(kHighestDegreeNodes)
    3. return graph
```

Worst Case Time complexity: **$O(kN^2)$**

Explanation: In policy B, we immunize and remove k highest degree nodes from the network.

Finding degree of each node → $O(N^2)$.
Sorting nodes in decreasing order of their degree → O(NlogN).
Removing top k nodes → $O(kN^2)$.
So the overall complexity for policy B → $O(N^2 + NlogN + kN^2) \rightarrow O(kN^2)$

- *Policy C:*

```
def immunizeUsingPolicyC(graph, k):
  # Immunize and remove the highest degree node iteratively(k times) from the network
    1. for i in range(k):
        a. highestDegreeNode = max(graph.degree_iter(), key=lambda node_degree:
           node_degree[1])[0]
        b. graph.remove_node(highestDegreeNode)
    2. return graph
```

Worst Case Time complexity: **$O(kN^2)$**
Explanation: In policy C, we immunize and remove highest degree node, k times.
Finding degree of each node → $O(N^2)$.
Finding node with highest degree → O(N).
Removing the highest degree node → $O(N^2)$.
Since this entire logic is repeated k times, the overall complexity for policy
C → $O(k*(N^2 + N + N^2)) \rightarrow O(kN^2)$

- *Policy D:*

```
def immunizeUsingPolicyD(graph, k):
    1. largestEigenValue, largestEigenVector = scipy.sparse.linalg.eigs(
       nx.to_numpy_matrix(graph), k=1, which='LM', return_eigenvectors=True)
    2. absValuesWithIndex = []
    3. for index, value in enumerate(largestEigenVector):
        a. absValuesWithIndex.append((index, abs(value)))
    4. kCorrespondingNodes = [absValueWithIndex[0] for absValueWithIndex in
       sorted(absValuesWithIndex, key = lambda absValueWithIndex : absValueWithIndex[1],
       reverse = True)][:k]
    5. graph.remove_nodes_from(kCorrespondingNodes)
    6. return graph
```

Worst Case Time complexity: **$O(N^3)$ if k<<N**
Explanation: In policy D, we remove k nodes corresponding to k largest(absolute) values in the eigen vector corresponding to the largest eigen value.
Finding the largest eigen value and the corresponding eigen vector → $O(N^3)$.
Sorting values of eigen vector in decreasing order of magnitude(absolute value) → O(NlogN).
Finding the nodes at the corresponding positions of eigen vector → O(N).
Removing each of these k nodes → $O(kN^2)$.
Hence, the overall complexity for policy D → $O(N^3 + NlogN + N + kN^2) \rightarrow O(N^3)$ if k<<N

d. Given k = k1 , β = β 1 , and δ = δ 1 , calculate the effective strength (s) of the virus on the immunized contact network (i.e., contact network without immunized nodes). Did the immunization policy prevented a network wide epidemic?

Answer:

For k=200, β = 0.2 and δ = 0.7

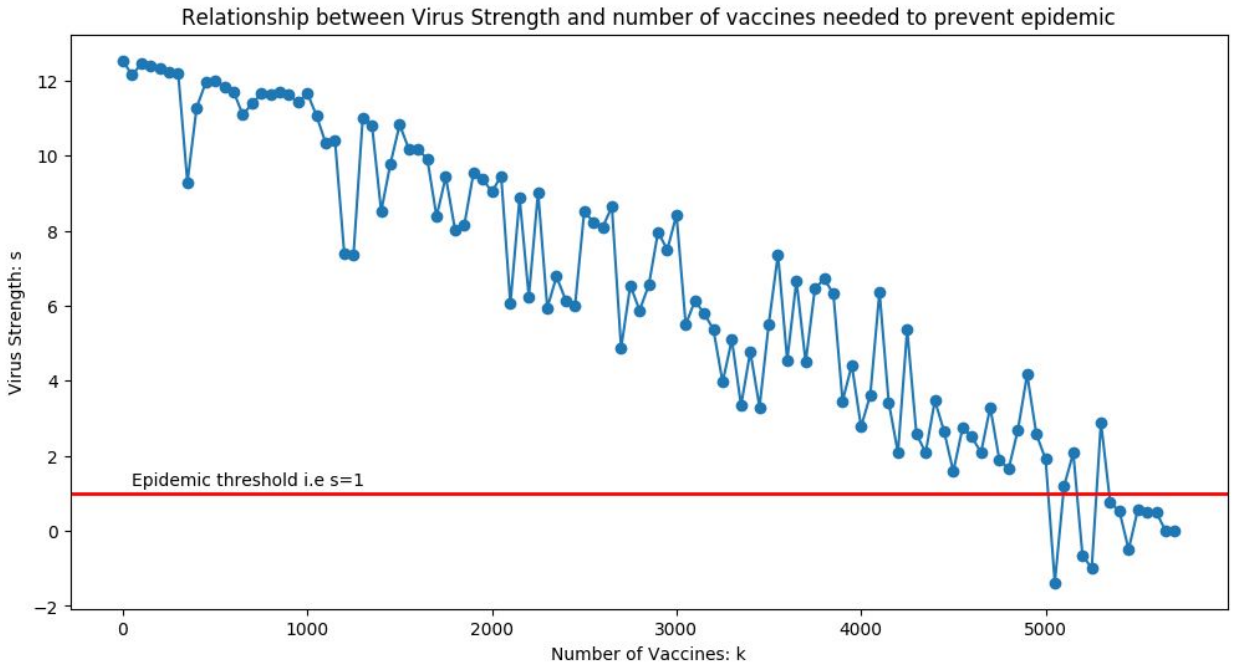
- *Policy A:* Effective strength of the virus = 12.35. Since effective strength is greater than 1, this immunization policy will not prevent the virus from spreading across the network.
- *Policy B:* Effective strength of the virus = 1.08. Since effective strength is almost equal to 1, this immunization policy may prevent the virus from spreading across the network.
- *Policy C:* Effective strength of the virus = 1.084. Since effective strength is almost equal to 1, this immunization policy may prevent the virus from spreading across the network.
- *Policy D:* Effective strength of the virus = 3.07. Since effective strength is greater than 1, this immunization policy will not prevent the virus from spreading across the network.

e. Keeping β and δ fixed, analyze how the value of k affects the effective strength of the virus on the immunized contact network (suggestion: plot your results). Estimate the minimum number of vaccines necessary to prevent a network wide epidemic.

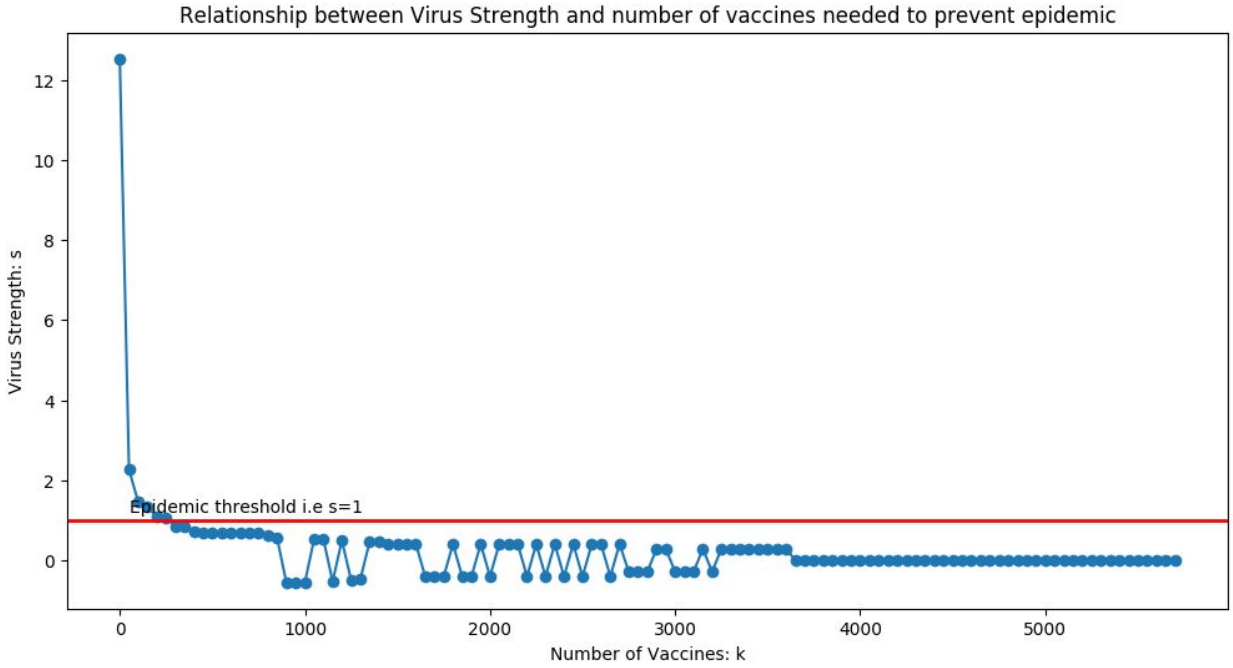
Answer:

For β =0.2 and δ =0.7 and varying k from 0 to 5715 (number of nodes in the network) in steps of 50 for policy A,B and C and steps of 200 for policy D (to reduce the execution time).

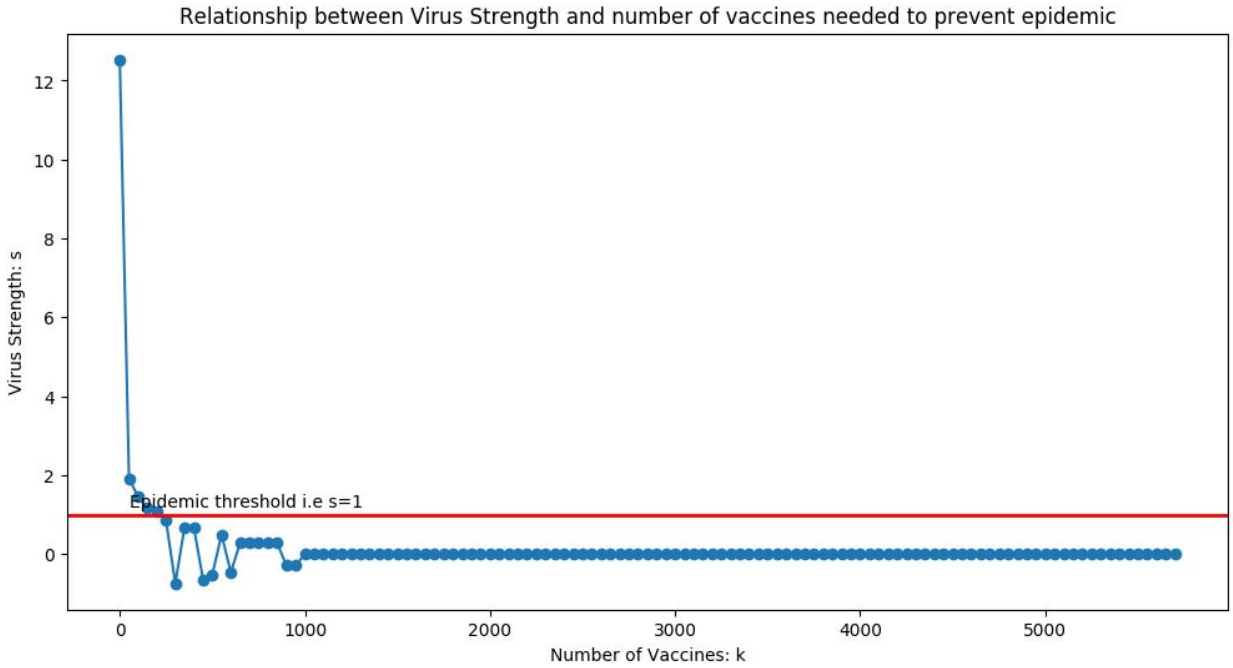
- *Policy A:* The minimum number of vaccines necessary to prevent an epidemic is approximately equal to *5050.*



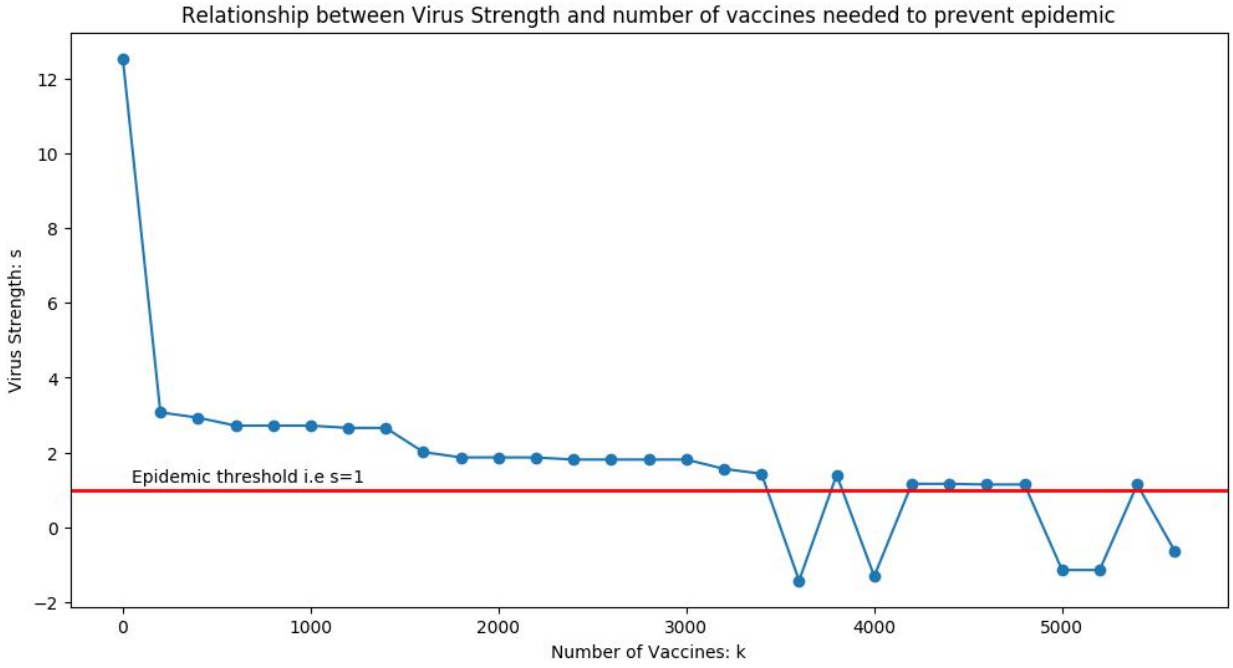
- *Policy B:* The minimum number of vaccines necessary to prevent an epidemic is approximately equal to *300.*



- *Policy C:* The minimum number of vaccines necessary to prevent an epidemic is approximately equal to 2*50.*



- *Policy D:* The minimum number of vaccines necessary to prevent an epidemic is approximately equal to *3700.*

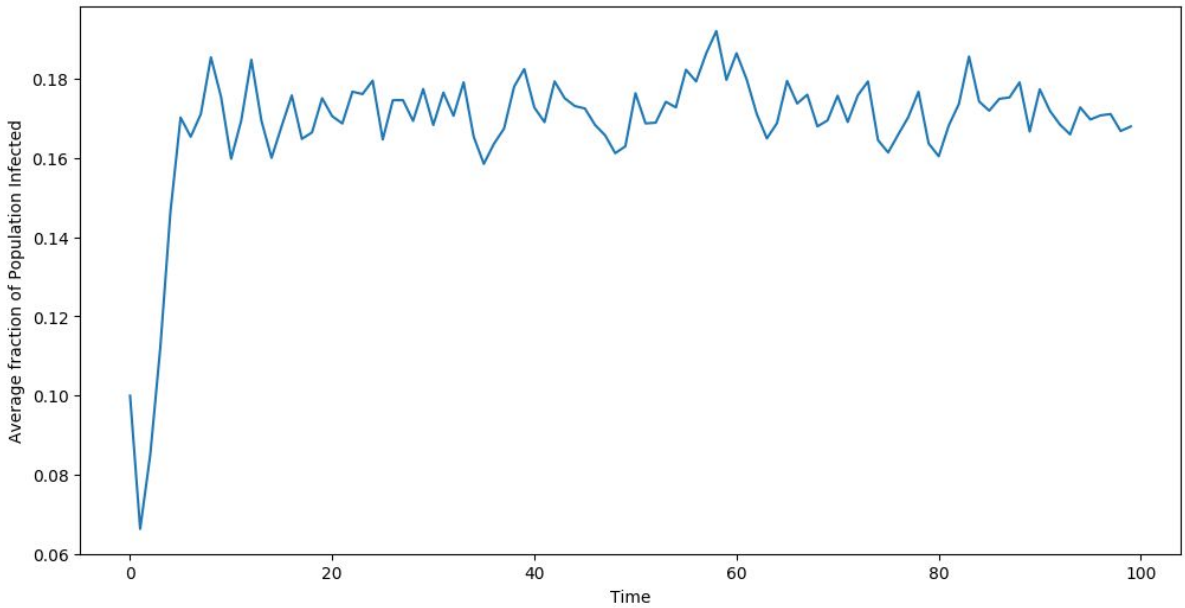


f. Given k = k1 , $\beta = \beta_1$, $\delta = \delta_1$, c = n/10, and t = 100, run the simulation from problem (2) for the immunized contact network 10 times. Plot the average fraction of infected nodes at each time step. Do the results of the simulation agree with your conclusions in (3d)?

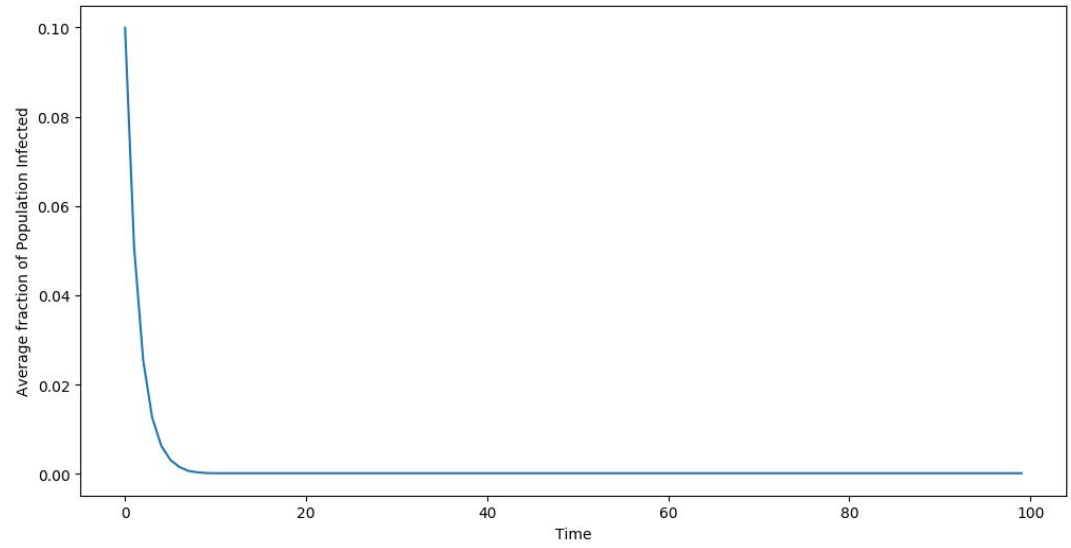
Answer:

For k=200, β = 0.2 and δ = 0.7

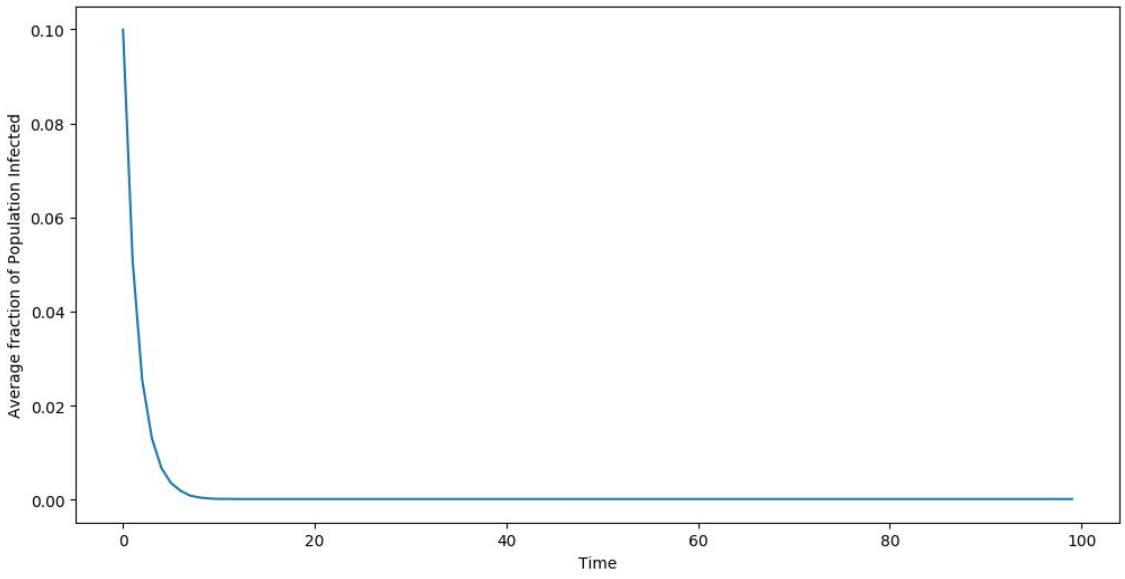
- *Policy A:* The infection results in an epidemic. The result is in sync with the conclusion from (3d).



- *Policy B:* The infection does not result in an epidemic. The result is in sync with the conclusion from (3d).



- *Policy C:* The infection does not result in an epidemic. The result is in sync with the conclusion from (3d).



- *Policy D:* The infection does not result in an epidemic. The result is not in sync with the conclusion from (3d).