Amal Sony(asony) - 200261398
Tanmaya Nanda(tnanda) - 200261608
Prayani Singh(psingh25) - 200251108

Graph Data Mining - Virus Propagation (Option 1)

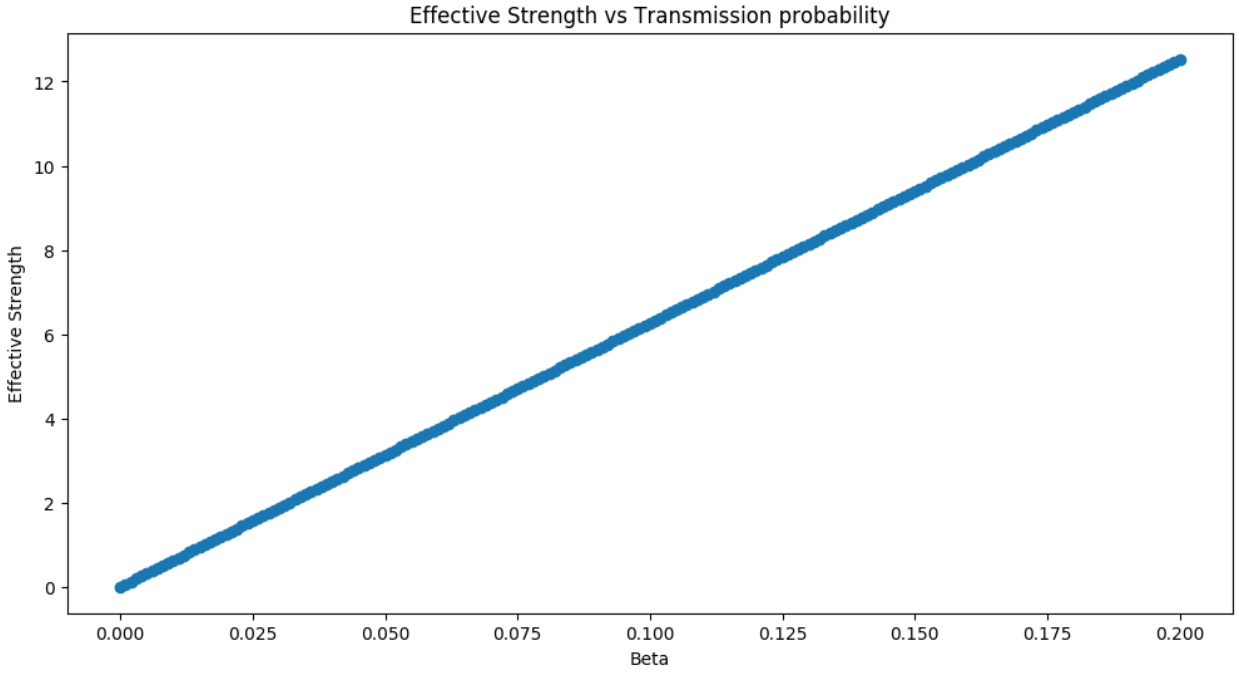
1.

a) Will the infection spread across the network (i.e., result in an epidemic), or will it die quickly?

Solution:
When β = β1 = 0.2 and δ = δ1 = 0.01,
Effective Strength = 12.52991307
Since Effective Strength > 1, infection will spread across the network and it will result in an epidemic.

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 network wide epidemic?

Solution:
The effective virus strength increases linearly with the value of β if δ is fixed, as shown below:

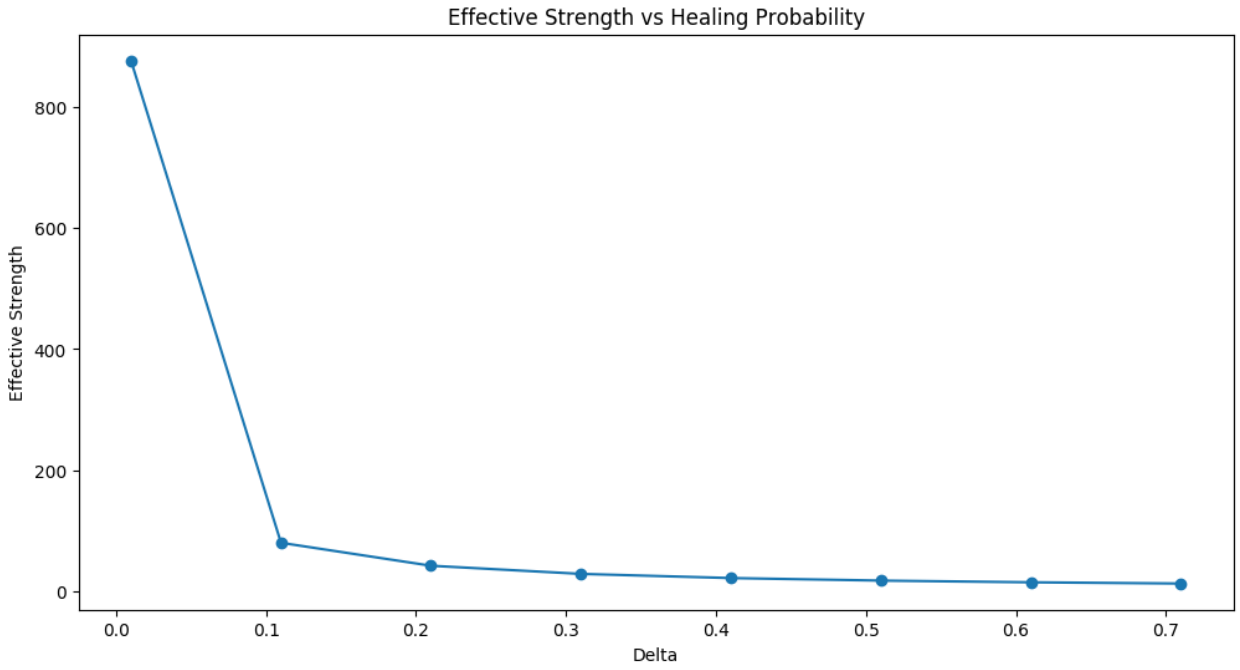


The minimum transmission probability β that results in a Networkwide epidemic = 0.0159618 for β = β1 = 0.2 and δ = δ1 = 0.01

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?

Solution:
The relation between δ and the effective strength of the virus while keeping β fixed is depicted in the graph below:



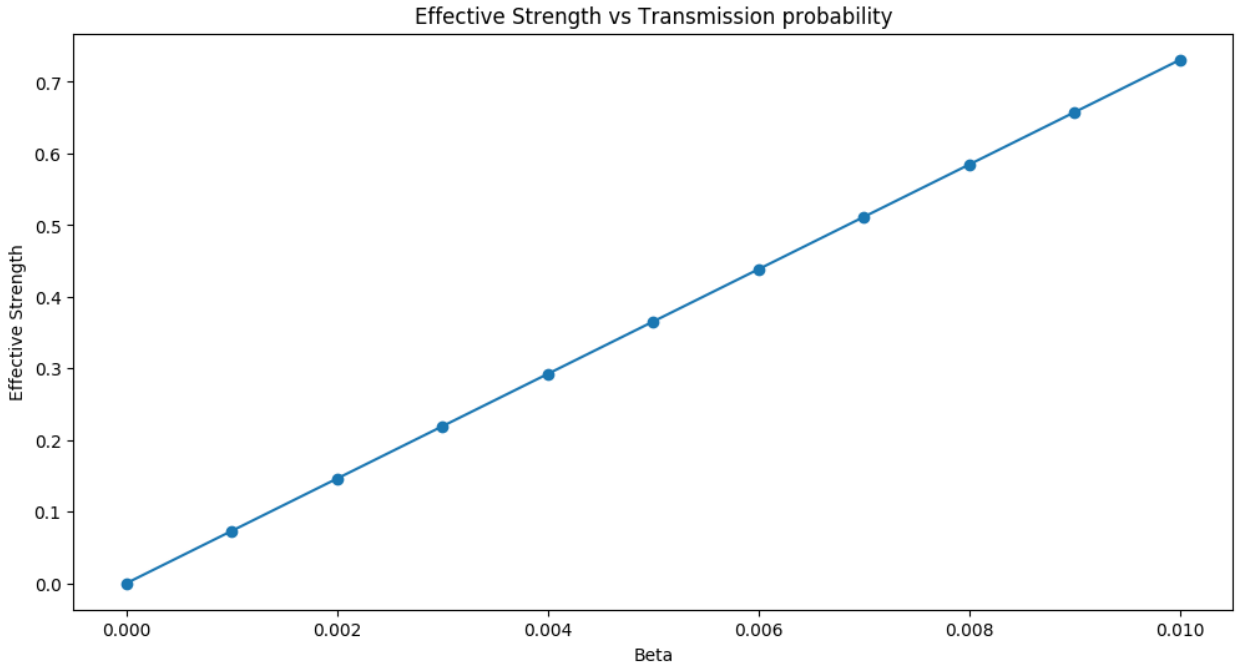
When β = β1 = 0.2 and δ = δ1 = 0.01,
The maximum healing probability for epidemic = 8.77093915

d) Repeat (1), (1a), (1b) and (1c) with β = β_2, and δ = δ_2

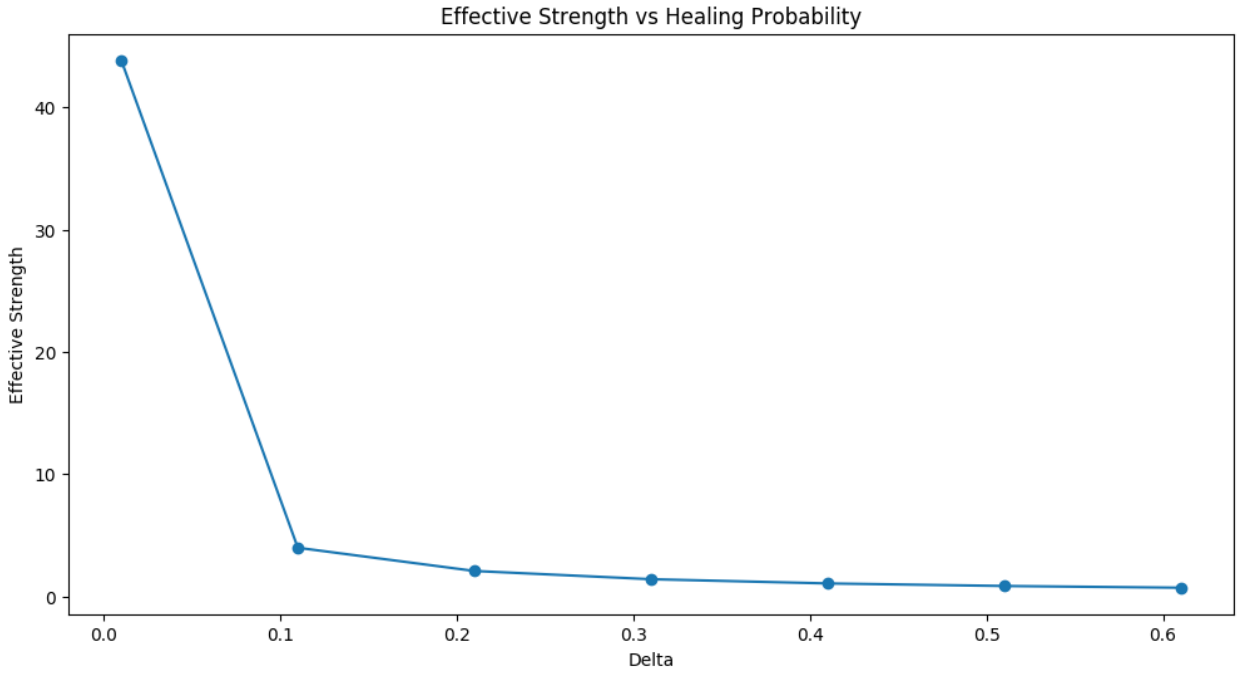
Solution:
For β = β2 = 0.01 and δ = δ2 = 0.60, (a), (b) and (c) are as follows:

a) Effective Strength = 0.7309116
Since Effective Strength < 1, the infection will die quickly.

b) Minimum transmission probability for epidemic = 0.01368155



c) Maximum healing probability for epidemic = 0.43854696

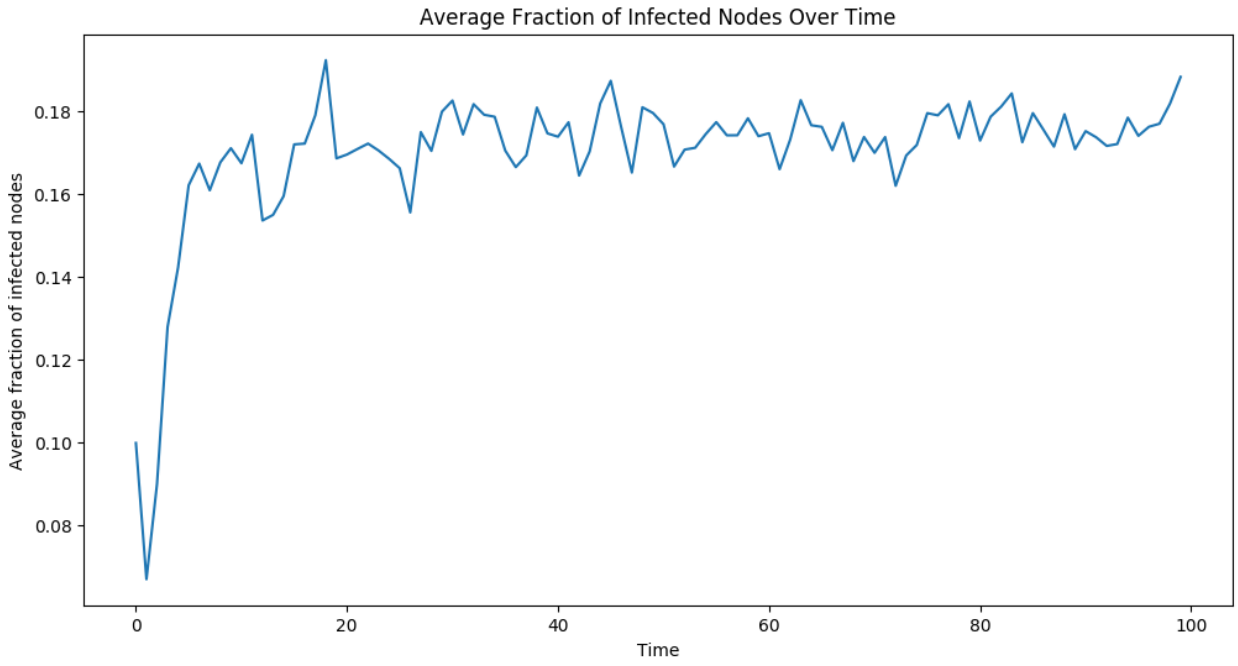


2. Write a program that simulates the propagation of a virus with the SIS VPM, given a static contact network, a transmission probability (β), a healing probability (δ), a number of initially infected nodes (*c*), and a number of time steps to run the simulation(*t*).The initially infected nodes should be chosen from a random uniform probability distribution. At each time step, every susceptible (i.e., noninfected) 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.

1. 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 $t = 100$.
2. 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)?
3. Repeat (2a) and (2b) with $\beta = \beta_2$, and $\delta = \delta_2$.

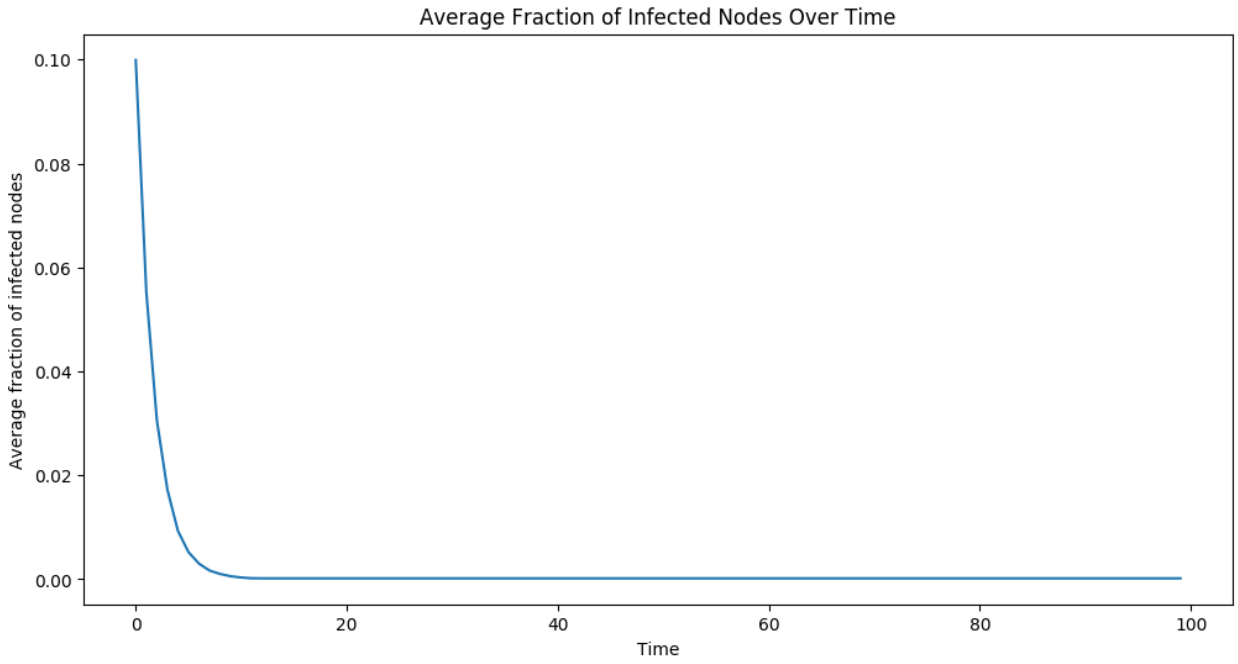
Solution:

For β = β1 (0.20) and δ = δ1 (0.70):



The infection resulted in an epidemic. The effective strength is 12.5. The result of the simulation is in sync with the conclusion in (1a).

For β = β2 (0.01) and δ = δ2 (0.60):



The infection died quickly and there was no epidemic. The effective strength is 0.73. The result of the simulation is in sync with the previous conclusion in (1a).

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.

Solution:

The optimal immunization policy can be finding the subset of k nodes among all n. Then choose k such subsets that give the largest drop in the maximum eigen value. The time complexity of this solution is of polynomial order. Since this solution is computationally expensive and uncontrollable, 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?

Solution:

- *Policy A*: This is a random technique. 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. This doesn't follow an algorithmic approach.
- *Policy B*: This is a better approach than policy A and aims to control the virus propagation by targeting the nodes with the highest degree. No random nodes are selected. 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 highly likely 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 neighbors will change. After removing a node with highest degree, we find more nodes with higher degree and it is more important to immunize them, and this process is repeated.
- *Policy D*: This is the best among all the above 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 as eigen value determines connectivity. 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?

Solution:

- Policy A: Select k random nodes for immunization.
  $k\_random\_nodes = random.sample(range(0, nx.number\_of\_nodes(graph)), k)$
  $graph.remove\_nodes(k\_random\_nodes)$

  Worst Case Time complexity: $O\ (kN^2)$
  Explanation: In policy A, we immunize random nodes and remove k nodes from the network. Removing a node from the graph → $O(N^2)$.
  Hence, removing k nodes and their neighbors → $O(k * (N^2)) \rightarrow O(kN^2)$

- Policy B: Immunize and remove k highest degree nodes from the network
  $k\_highest\_degree\_nodes = [node\ for\ node, degree\ in\ sorted(graph.degree\_iter(), key = itemgetter(1), reverse = True)][:k]$
  $graph.remove\_nodes(k\_highest\_degree\_nodes)$

Worst Case Time complexity: $O(kN^2)$
Explanation: In policy B, we immunize and remove k highest degree nodes from the network. First, we find degree of each node. Then we sort the degrees and remove top k nodes.

Finding degree of each node → $O(N^2)$.
Sorting nodes in decreasing order of 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: Immunize and remove the highest degree node iteratively (k times) from the network

$$for\ i\ in\ range(k):$$
$$highest\_degree\_node = max(graph.degree\_iter(), key = lambda\ node\_degree: node\_degree[1])[0]$$
$$graph.remove\_node(highest\_degree\_node)$$

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 process is repeated k times, the overall complexity for policy C→ $O(k * (N^2 + N + N^2)) \rightarrow O(kN^2)$

- Policy D:

$$largest\_eigen\_value, largest\_eigen\_vector = scipy.sparse.linalg.eigs(nx.to\_numpy\_matrix(graph), k = 1, which = 'LM')$$
$$magnitude\_with\_index = []$$
$$for\ index, value\ in\ enumerate(largest\_eigen\_vector):$$
$$magnitude\_with\_index.append((index, abs(value)))$$
$$k\_nodes = [magnitude\_with\_index[0]\ for\ magnitude\_with\_index\ in\ sorted(magnitude\_with\_index, key = lambda\ magnitude\_with\_index: magnitude\_with\_index[1], reverse = True)][:k]$$
$$graph.remove\_nodes(k\_nodes)$$

Worst Case Time complexity: $O(N^3)\ if\ k << N$
Explanation: In policy D, we remove k nodes corresponding to k absolute largest values in the eigen vector corresponding for 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 → $O(NlogN)$. Finding the nodes at the corresponding positions of eigen vector → $O(N)$.
Removing each of the 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 = β1 , and δ = δ1 , 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 networkwide epidemic?

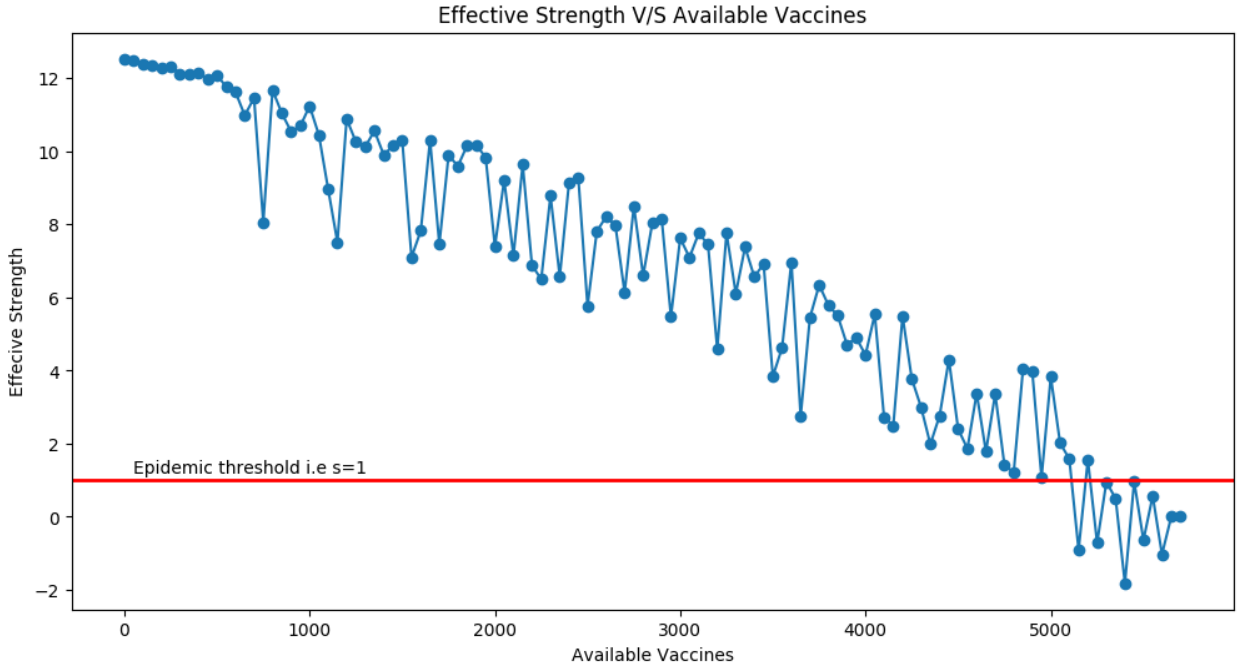
Solution:
Given k = k1 = 200, β = β1 = 0.2, and δ = δ1 = 0.7, we vary k from 0 to the number of nodes in the graph, 5715 in steps of 50 for Policy A, Policy B and Policy and in steps of 200 for Policy D.

i) Policy A: Effective strength = 12.33777405. Since Effective strength>1, this immunization policy would not be able to prevent the virus from spreading in the network.
ii) Policy B: Effective strength = 1.08027548. Since the Effective strength just a little greater than 1, this policy might prevent the virus from spreading in the network.
iii) Policy C: Effective strength = 1.08452054. Since the Effective strength just a little greater than 1, this policy might prevent the virus from spreading in the network.
iv) Policy D: Effective strength = 3.07052788. Since Effective strength>1, this immunization policy would not be able to prevent the virus from spreading in 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 networkwide epidemic.

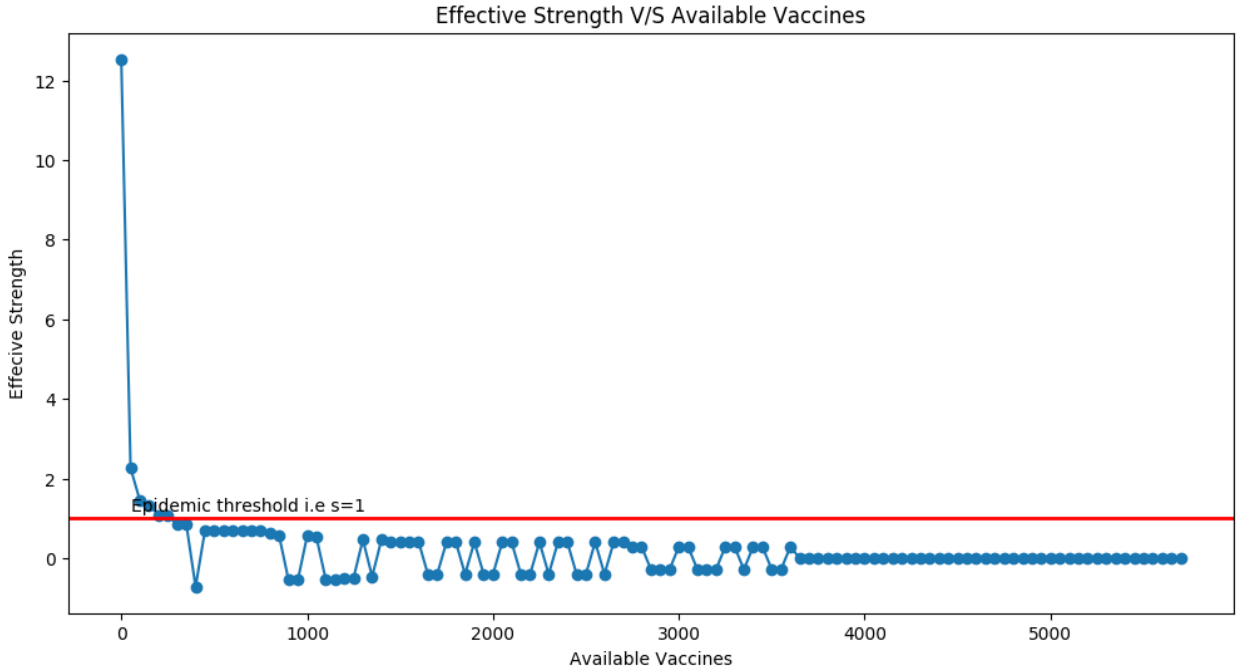
Solution:
For β = β1 = 0.2, and δ = δ1 = 0.7 fixed, varying value of k, we get the following results:

i) Policy A
The minimum number of vaccines required to prevent the epidemic from spreading in the network is 5150.



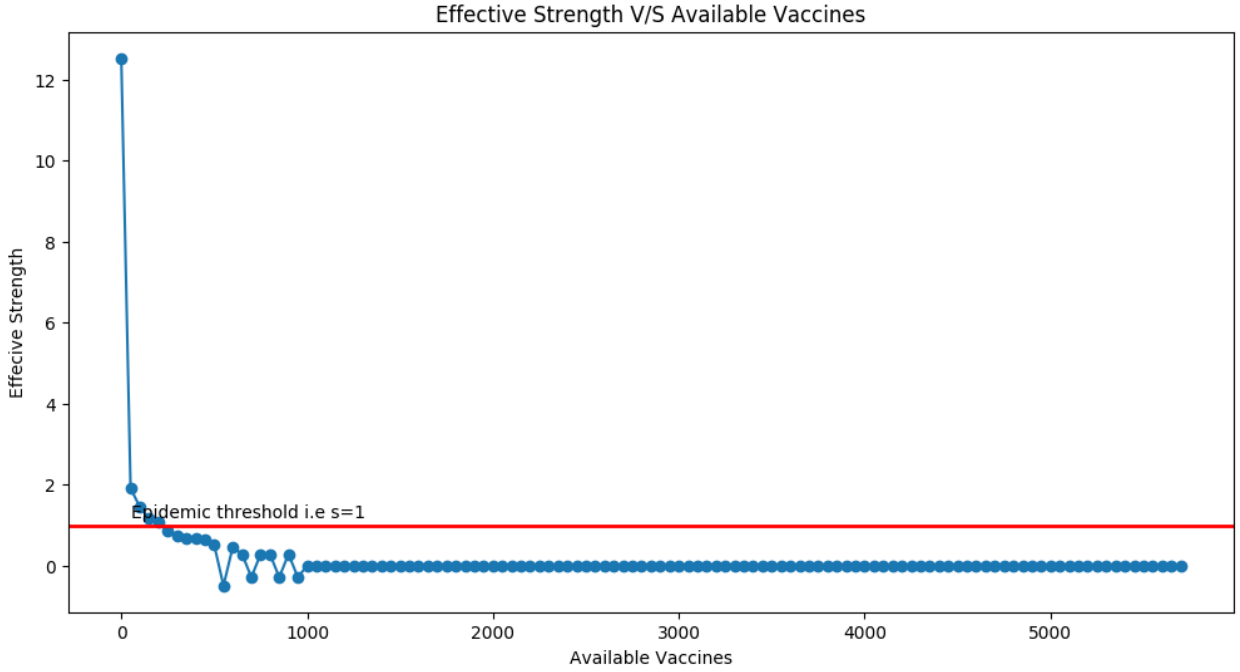
ii) Policy B

The minimum number of vaccines required to prevent the epidemic from spreading in the network is 300.



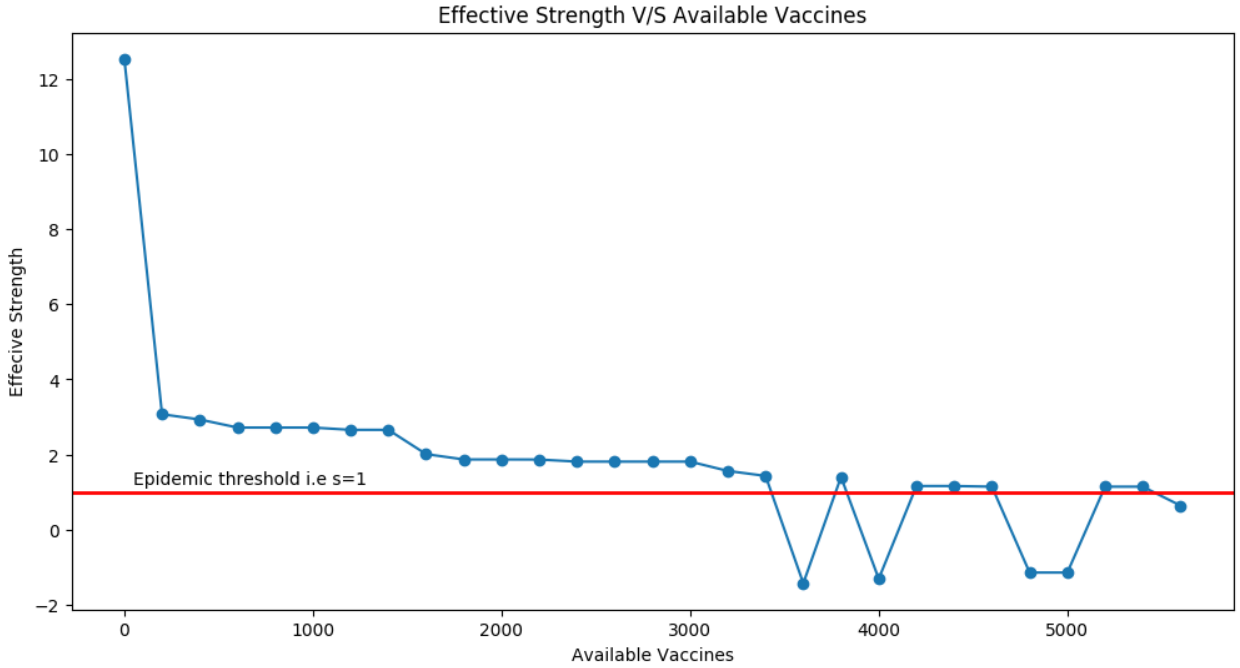
iii) Policy C

The minimum number of vaccines required to prevent the epidemic from spreading in the network is 250.



iv) Policy D

The minimum number of vaccines required to prevent the epidemic from spreading in the network is 3600.



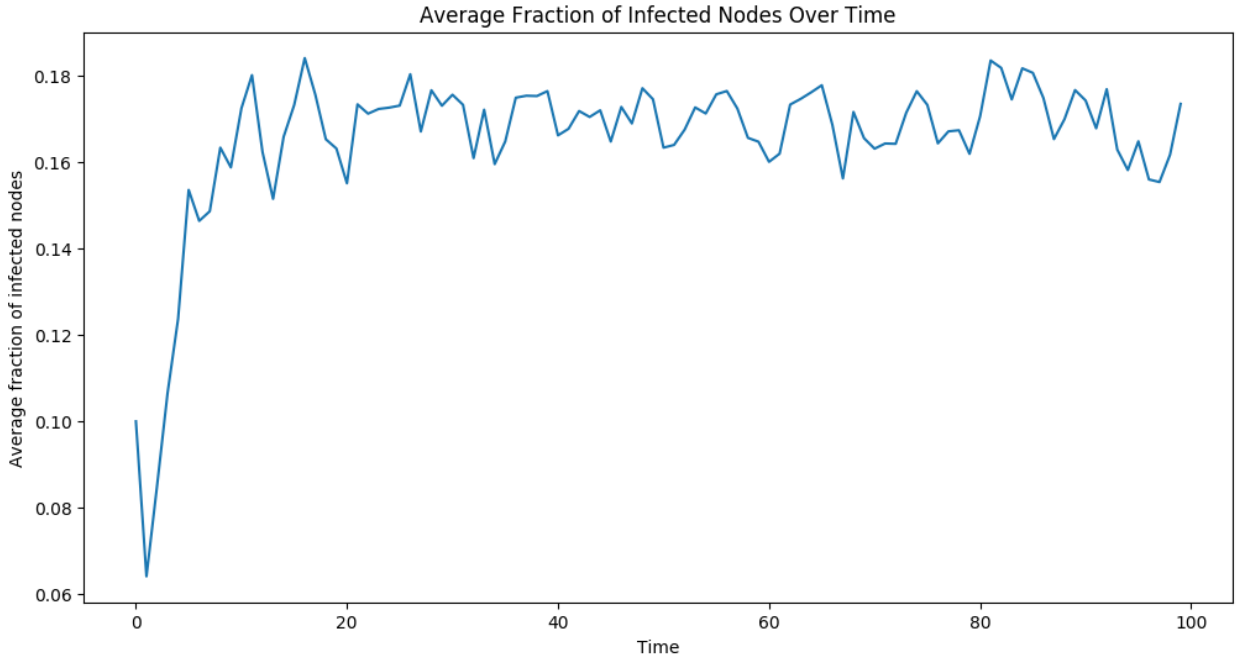
f) Given k = k1 , β = β 1 , δ = δ 1 , c = n/10, and t = 100, run the simulation from problem (2) for the immunized contact network 10 times. Plot the average fraction of infected nodes at each time step. Do the results of the simulation agree with your conclusions in (3d)?

Solution:

Given k = k1 = 200, β = β1 = 0.2 and δ = δ1 = 0.7

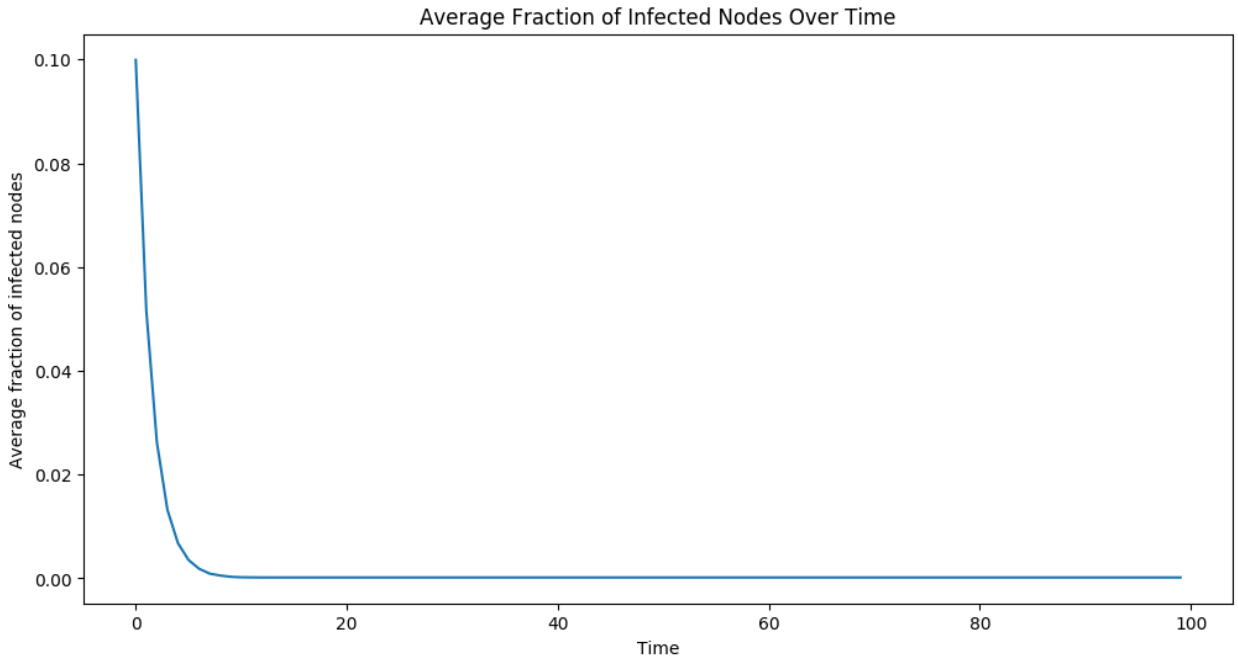
i) Policy A

The infection leads to an epidemic and hence the simulation agrees with the conclusion in 3d.



ii) Policy B

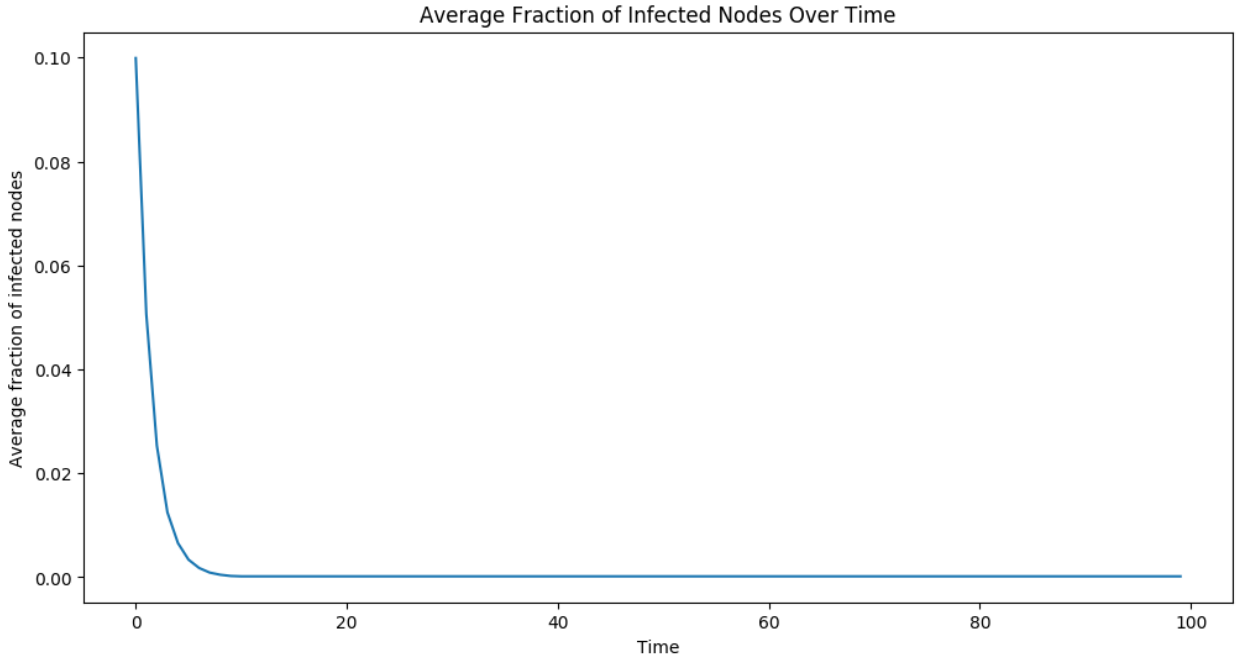
The infection dies quickly and hence the simulation agrees with the conclusion in 3d.



iii) Policy C:

The infection dies quickly and doesn't result in an epidemic and hence the simulation agrees with the

conclusion in 3d.



iv) Policy D:

The infection dies quickly and doesn't result in an epidemic and hence the simulation doesn't agree with the conclusion in 3d.