

Network Analysis

Lab assignment 2: Social network analysis and modelling

Group 2

Aline D'Intino Gonçalves (5132185)

Amaru Sinchico Arias (5022096)

Priyanka Kumar (7319533)

Sybren Bouwman (5098386)

Table of contents

Exercise 1: Build and analyse a small network from Facebook	4
Question 1 (2 points)	4
Question 2 (3 points)	4
Question 3 (5 points)	4
Question 4 (3 points)	6
Question 5 (3 points)	6
Question 6 (3 points)	7
Question 7 (3 points)	8
Question 8 (4 points)	8
Question 9 (8 points)	9
Question 10 (5 points)	9
Exercise 2: Formulate a social network for certain architectures	11
Question 11 (3 points)	11
Question 12 (2 points)	11
Question 13 (3 points)	12
Question 14 (2 points)	13
Question 15 (3 points)	14
Question 16 (8 points)	14
Question 17 (2 points)	15
Question 18 (4 points)	16
Question 19 (3 points)	16
Question 20 (5 points)	17
Question 21 (8 points)	18
Exercise 3: Build the social network of this class and simulate the contagion process	19
Question 22 (5 points)	19
Question 23 (2 points)	19
Question 24 (5 points)	20
Question 25 (10 points)	21

Question 26 (15 points)	22
Question 27 (3 points)	23
Question 28 (2 points)	23
Question 29 (3 points)	24
Question 30 (10 points)	24
Question 31 (10 points)	26
Question 32 (2 points)	28
Question 33 (20 points)	28
Reference	29

Exercise 1: Build and analyse a small network from Facebook

Question 1 (2 points)

Check out the Summary and Plot, how many friends do Douglas have on Facebook? Is this a directed or undirected graph and why? What is the meaning of the link between nodes in the plot?

The summary shows that Douglas has 93 friends (vertices) on Facebook divided into 8 groups (BookClub, Highschool, College, Music, Family, Spiel, GraduateSchool and Work). The Basic plot of Douglas's Facebook friends show the network that is built. This graph is undirected, because the edges have no directions (no arrows that indicate the directions of the edge). The undirected graph indicates that there is a two-way relationship between nodes, so each edge can be directed in both the directions. These links show which nodes are directly connected. The meaning of the undirected link in the graph is that there is a two-way relationship between nodes that have edges that have a direction both ways. A directed graph could indicate a one-way friendship and the attention/focus could be in one direction. This would be in the case of users following another user on Twitter, for example.

Question 2 (3 points)

Compare the degree, closeness and betweenness of Vertex 1 to the values of other nodes in the network. How will you evaluate the role of Vertex 1 in this network?

- **Degree** is the number of edges and nodes connected to the given node. Vertex 1 has a degree of 32, indicating that there are 32 edges attached to vertex 1. Vertex 1 has the highest degree in the network. A node with a high degree has a lot of edges attached to it, meaning that there are multiple connections with other nodes. This shows that this vertex is connected to the most number of other nodes in this network. In this case there are 32 nodes attached to vertex 1. The minimum degree in this network is 0.
- **Closeness** is the mean shortest distance from a given node to all the other nodes in the network. Closeness for vertex 1 has a value of 0.0212, indicating that the mean shortest distance from vertex 1 to all the other nodes is 0.0212. This is the highest value of closeness compared to all other vertices. The minimum closeness in this network is 0.0107.
- **Betweenness** represents how often the node is a bridge between other nodes, giving the number of times the given node lies on the shortest path between two other nodes. Vertex 1 has the highest betweenness with a value of 0.14111. This means that Vertex 1 serves the most often as a bridge between other nodes. The minimum betweenness in this network is 0. giving that vertex 1 appears 0.14111 times between two other nodes.

Looking at the total network, Vertex 1 has the most connection to other nodes, has the highest mean shortest distance in the network, and serves the most as a bridge between nodes. So, based on these metrics, it looks like Vertex 1 is the most central node.

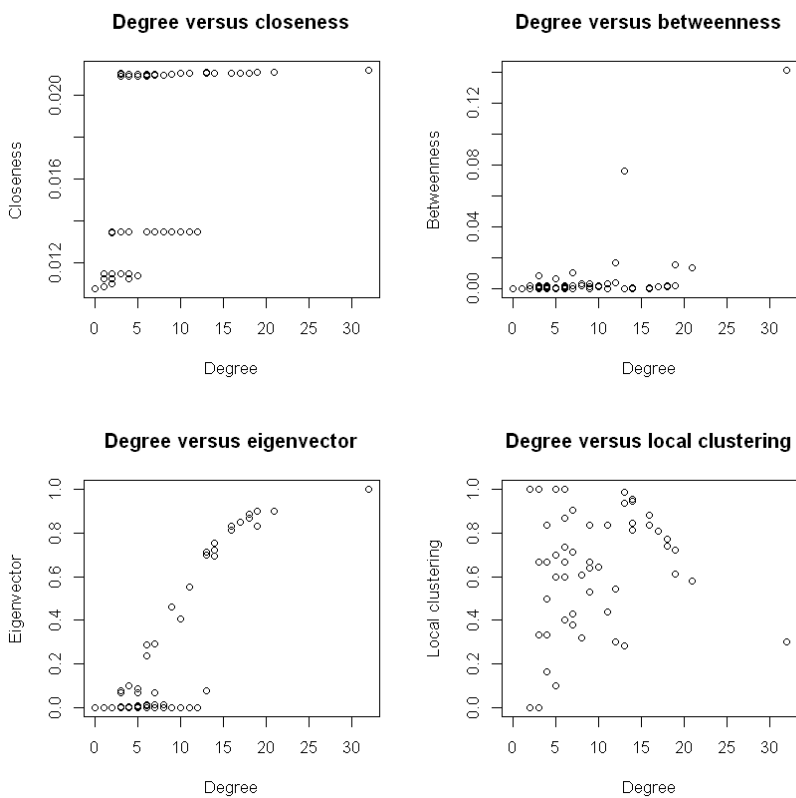
Question 3 (5 points)

In the lecture, we introduced a few measures of centrality: degree, betweenness, eigenvector. Try to find top 5 nodes according to a) degree, b) betweenness, c) closeness and d) eigenvector. And develop the scatter plot between different metrics, you can refer to the code below. Discuss with your group and describe to your teacher, 1) how well does the top 5 nodes by different metrics overlap with each other? 2) why we need more than one metric to define centrality?

Top 5 nodes according to a) degree, b) betweenness, c) closeness and d) eigenvector:

Metrics	Top 5 nodes according to the metrics
Degree	<pre>print(tail(sort(deg), 5))</pre> <p>30 23 31 12 1 18 19 19 21 32</p>
Closeness	<pre>print(tail(sort(cls), 5))</pre> <p>31 12 23 16 1 0.02108641 0.02110576 0.02110576 0.02111545 0.02120793</p>
Betweenness	<pre>print(tail(sort(btw), 5))</pre> <p>12 23 76 16 1 0.01367066 0.01585297 0.01698026 0.07635683 0.14111307</p>
Eigenvector	<pre>print(tail(sort(eig\$vector), 5))</pre> <p>26 30 12 31 1 0.8678475 0.8838761 0.9012656 0.9016029 1.0000000</p>

The scatter plot between different metrics:

Scatter plots	Code:
 <p>The figure contains four scatter plots arranged in a 2x2 grid. Each plot has 'Degree' on the x-axis (ranging from 0 to 30). 1. Degree versus closeness: The y-axis is 'Closeness' (0.012 to 0.020). Data points are clustered at low degree values with varying closeness, and a few points at higher degrees with high closeness. 2. Degree versus betweenness: The y-axis is 'Betweenness' (0.00 to 0.12). Most points are near zero, with a significant outlier at degree ~12 and betweenness ~0.08. 3. Degree versus eigenvector: The y-axis is 'Eigenvector' (0.0 to 1.0). Points are mostly at low degree and low eigenvector values, with a clear upward trend for higher degree nodes. 4. Degree versus local clustering: The y-axis is 'Local clustering' (0.0 to 1.0). Points are scattered, with a general downward trend as degree increases.</p>	<pre>par(mfrow = c(2,2)) plot(deg, cls, main = "Degree versus closeness", xlab = "Degree", ylab = "Closeness") plot(deg, btw, main = "Degree versus betweenness", xlab = "Degree", ylab = "Betweenness") plot(deg, eig\$vector, main = "Degree versus eigenvector", xlab = "Degree", ylab = "Eigenvector") plot(deg, lcl, main = "Degree versus local clustering", xlab = "Degree", ylab = "Local clustering")</pre>

As seen in the table and scatter plots above, some nodes seem to be overlapping for different metrics. From the top 5 nodes by the different metrics, the nodes 1 and 12 are overlapping. These two nodes are in the top 5 nodes of each metrics. The comparison between metrics give the following overlappings:

- **Degree versus Closeness:** the nodes 1, 12, 23 and 31 are overlapping.
- **Degree versus Betweenness:** the nodes 1, 12 and 23 are overlapping.
- **Degree versus Eigenvector:** the nodes 1, 12, 30 and 31 are overlapping.
- **Closeness versus Betweenness:** the nodes 1, 12, 16 and 23 are overlapping.
- **Closeness versus Eigenvector:** the nodes 1, 12 and 31 are overlapping.
- **Betweenness versus Eigenvector:** the nodes 1 and 12 are overlapping.

The node scores are different for the different metrics, we need multiple metrics to define centrality. As seen in the lecture, all metrics say something different about centrality. Degree is the most obvious metric of centrality, but it shouldn't be the only one. Centralization defines how much the network revolves around a single node. Besides the centralization, we should take, closeness centrality, degree centrality, betweenness centrality, and the measured eigenvector into account in order to include not just the number of links and adjacent nodes but also the closeness to all other nodes including connections through other nodes (betweenness) and the how central the neighbors are. If one would only look at one metric, a wrong conclusion can be made about the centrality of the network. So the metrics you take depend on the scenario/case, as the different metrics say something different about centrality.

Question 4 (3 points)

Discuss within your group on how you understand each of these measures. And describe to your teacher, 1) why diameter should be larger than 1, and other metrics such as edge density and transitivity are smaller than 1? 2) is this a tightly knitted network?

- **Diameter** of network is how big the network is, which gives the length of the longest or the shortest path between any pair of nodes in the network. The diameter should be larger than 1, because this indicates that at least two nodes are connected. If a node is isolated, the length is infinite.
- **Edge density** is the number of edges and the number of possible edges. In a complete network, the network already has the maximum number of edges, and no more edges can be added. The edge density of the network will have a maximum number of 1, indicating the network is fully dense (connected) and thus can not exceed the value of 1.
- **Transitivity** gives the probability that the nodes are connected, revealing the existence of interconnected communities. The transitivity of a node is the ratio of the existing connections with a given node's neighbor, to the maximum number of such connections. The maximum value of transitivity is 1, indicating that all nodes are fully connected to its neighbors, thus transitivity can not exceed 1.

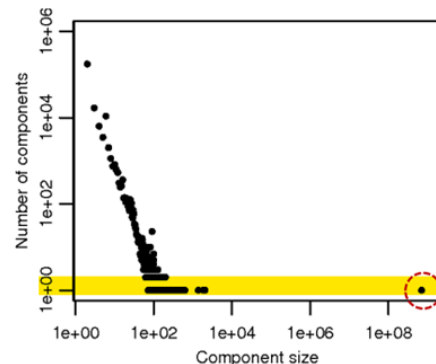
The edge density of this network is 0.075, transitivity is 0.66, and the diameter is 4. These metrics are very low, indicating that this is not a tightly knitted network.

Question 5 (3 points)

Reflect on what we have discussed about the Facebook network on Slide 42 of the lecture. Do you think this small network of Douglas resonates with some general patterns of the entire Facebook network in terms of the components size and number? How can you explain such an observation?

We do see a large number of nodes being connected to a few components, just like in the real facebook network. It is not close to 99.91% though, which may be due to the small network size compared to the real network. The small network of Douglas' friends gives the values 47, 2, 20, 3, 7, 6, 5, 1, 1, 1 for component size and value 10 for number. The component size is the size of the clusters and the number is the number of clusters. The entire Facebook network has many connected components, where most of the components are very small. The graph below shows the values for the entire Facebook network. At a value of 10, the component sizes are between $1e+02$ (100) and $1e+04$ (10000). This does not resonate with the network of Douglas' friends, which have a component size between 1 and 47. This could be

because of the small number of friends of Douglas, where most individuals have a higher amount of friends on Facebook. Douglas has one main friend group, and the other groups are separate smaller groups with few friends.

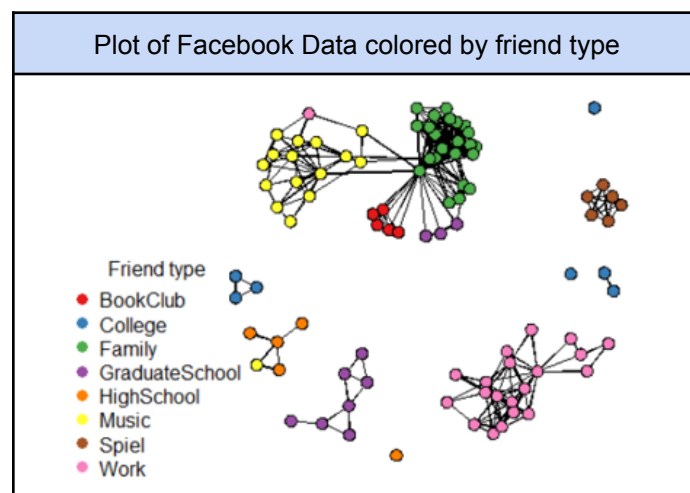


The average value of the diameter was 4.7 for global users and 4.3 for U.S. users. This diameter value is close to the diameter value of Douglas' network, but a bit higher. The local clustering coefficient is also different. For users with 100 friends, the average is 0.14. Douglas' network contains almost 100 users, but the local clustering coefficient is much higher. This coefficient seems to decrease as the number of friends increases. This is near the value of the diameter of Douglas, based on this metric the small network seems to resonate to some general patterns of the entire network. Based on the metrics and how it should be interpreted, one can say that the small network of Douglas, does not really resonate much to some general patterns of the entire Facebook network.

Question 6 (3 points)

From the plot you produced, do you find instances of intermingling of Douglas friends (i.e., belong to different groups but end up in the same component)? Do you find any isolated groups here? What can you conclude about the mixing of Douglas' Facebook friends?

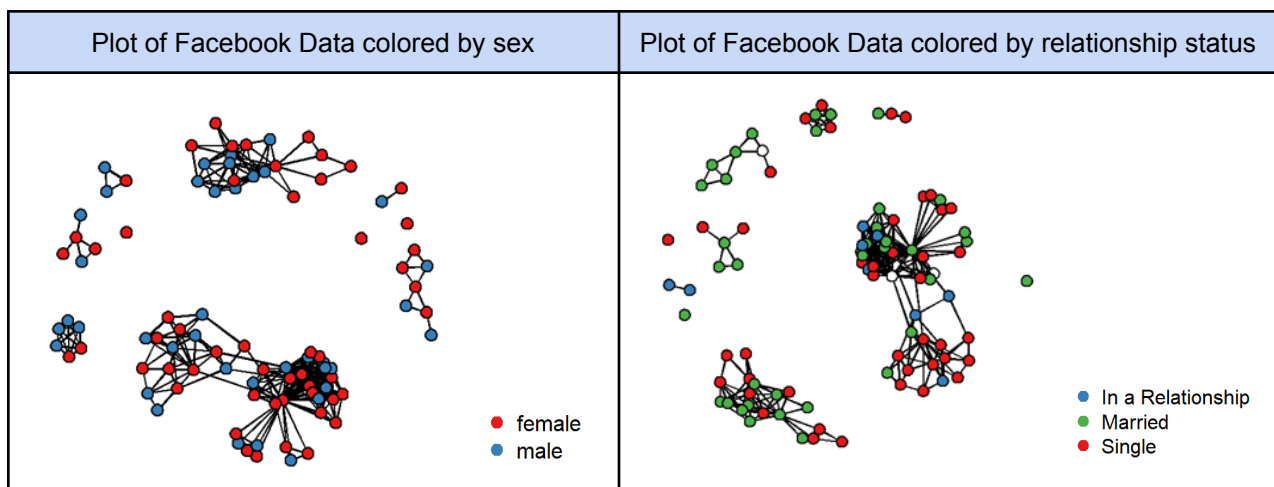
The 93 friends of Douglas belong to 8 groups (BookClub, College, Family, GraduateSchool, HighSchool, Music, Spiel and Work). The vertex of the 93 friends of Douglas are colored according to their group (see graph below). We see that family, graduate school, book club, music and one node of work are connected to each other in the same cluster. A different cluster of High school also contains one node of Music. There are some isolated groups that are not connected to others, such as college, as they are not in the cluster with any other color. Also, the group Spiel is isolated. The whole picture shows that overall there are no mixed groups. This is because of the friends that are isolated in certain groups, that are not connected to other groups.



Question 7 (3 points)

Using the above code as a reference, check out the attributes of other factors (e.g., relationship_status, sex) in terms of people in the same components. Note that you can specify the 'attrname' parameter within the function 'get.vertex.attribute'. Discuss with your group and describe to your teacher whether or not these factors are the keys in determining the formation of components.

The factors sex and relationship status seems not to be the key factors for the formation of the components, because these factors are mixed among the different groups where there are no isolations of certain groups. Sex is likely not a key factor in the formation of groups, it is well distributed over the different groups. Relationship likely isn't either, because it is again well mixed. The connections are not due to the attributes like gender or relationship status. The share of colors inside each component should be more homogenous, like when split by groups, when one split by "sex" we have a very heterogenous definiton.



Question 8 (4 points)

From this analysis, what do you observe from the density values? Are they similar across different groups? What is the minimum and maximum value you observed here and how do you explain that?

```
BookClub
"Density for BookClub friends is 1"
College
"Density for College friends is 0.19047619047619"
Family
"Density for Family friends is 0.624505928853755"
GraduateSchool
"Density for GraduateSchool friends is 0.266666666666667"
HighSchool
"Density for HighSchool friends is 0.3"
Music
"Density for Music friends is 0.316666666666667"
Spiel
"Density for Spiel friends is 1"
Work
"Density for Work friends is 0.3"
```

The density values are different across the groups, with a maximum value of 1 for the group BookClub and Spiel, and a minimum value of 0.19 for the group College. Density gives the proportion of possible connections in the network that are present. Lower values indicate that there are no connections and higher values indicate that there are connections. The network shows that the College (colored blue) has little to no connections, that's why the value is so low. The friends in the groups BookClub (red) and Spiel (purple) are potentially fully connected.

- There are a few high density subnetworks (2) with density 1, but most networks (5) are around 0.2-0.3.
- The larger subgroups have lower densities on average, and the small subgroups BookClub and Spiel are very interconnected.
- The minimum value is 0.19 and the maximum is 1. Spiel is fully interconnected and that's why it is 1, just like the BookClub group.
- College and Graduate school have the lowest densities. This could be because of the many different classes, where he made friends.

Question 9 (8 points)

To answer this question, search and discuss within your group on the theory of 'community detection'. Describe to your teacher what community detection is, and why it is useful to understand complex networks.

The community detection algorithm is used to discover communities/groups that have common interests and/or features/properties. Communities are defined as a subset of nodes that are densely connected to each other and connected to the nodes in the other communities in the same network (Jayawickrama, 2021). A complex or large network has a lot of nodes and edges, detecting communities manually can then be a difficult task. By applying the community detection algorithm the different communities in the complex or large network can be detected.

There are two types of methods for detecting communities in a network: the Agglomerative Methods and Divisive Methods. The agglomerative method starts with an empty network that consists of the nodes of the original network without edges. Then one-by-one the edges are added to the network, where the stronger edges are added first and after that the weaker edges are added. In the divisive method, out of the full network the edges are taken off iteratively. The edges with the highest weight are removed first step-by-step. At every step the edge-weights are calculated iteratively, because the weight of the remaining edges changes after the edge is removed. After some number of steps, the cluster of densely connected nodes are formed (Joshi, 2020).

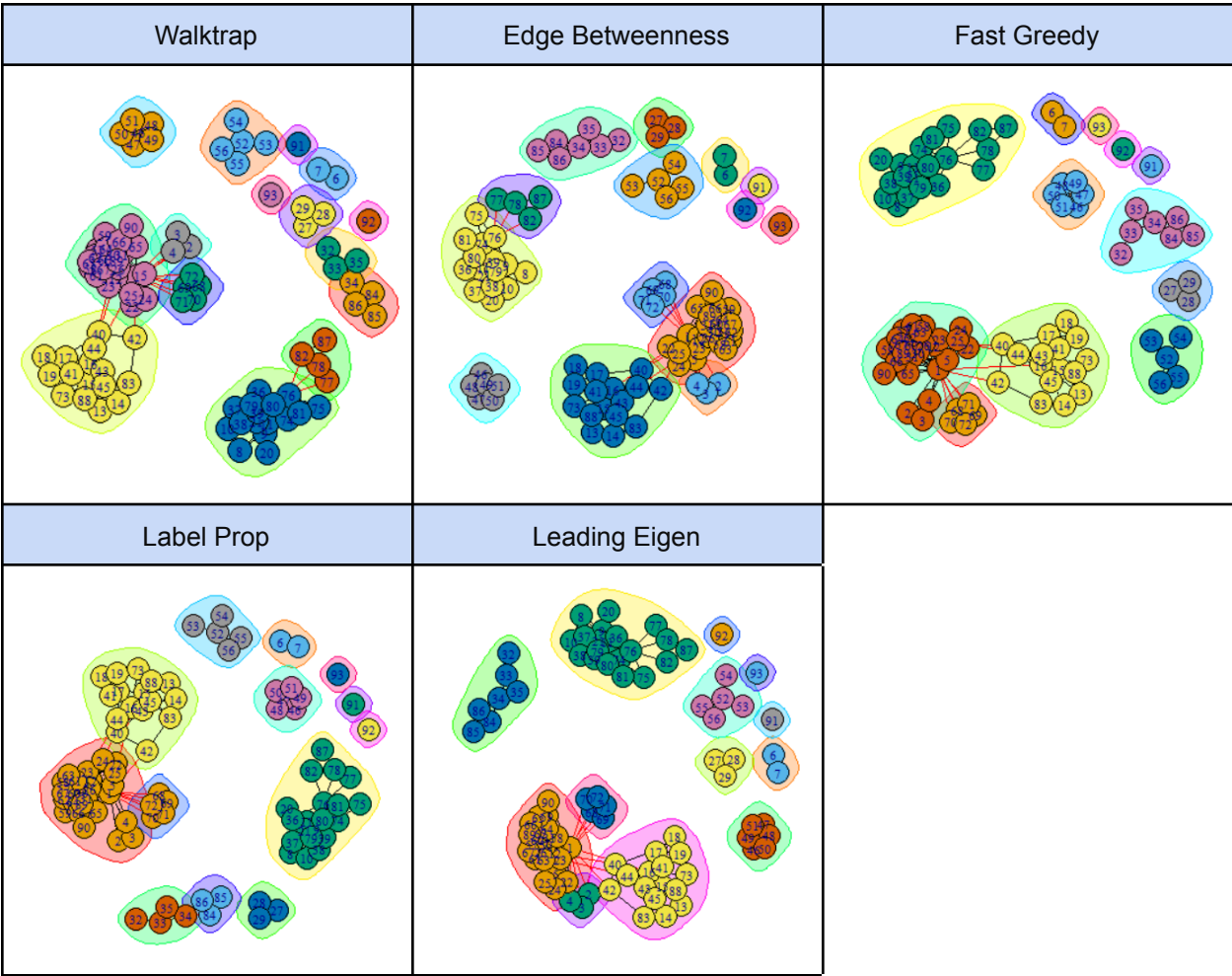
Some examples of implementation of the community detecting algorithm:

- Detecting the social circle of a person on the different social media platforms.
- Detecting people that speak the same language or have other similar characteristics
- Exploring the different stakeholders (customers, competition, partners, etc.) in the market around a specific brand or company.
- A brand or company that wants to know which local communities there are to improve their services and target specific communities.

Question 10 (5 points)

Compare the plots that you generate from the different algorithms; do you find them similar and Why? Do you think this observation (similar or not similar by using different community detection algorithms) holds in more complex networks?

The plots (see below) are largely similar, especially the smaller subgroups such as Spiel and BookClub stay the same in all iterations. The larger subgroups show some variation however, where smaller subgroups are absorbed in larger subgroups. It is expected that in larger networks there will be more variation as well, just like we saw here with smaller subgraphs and larger subgraphs. This is because in larger networks there are more ambivalent nodes due to the increased number of nodes and connections.



Exercise 2: Formulate a social network for certain architectures

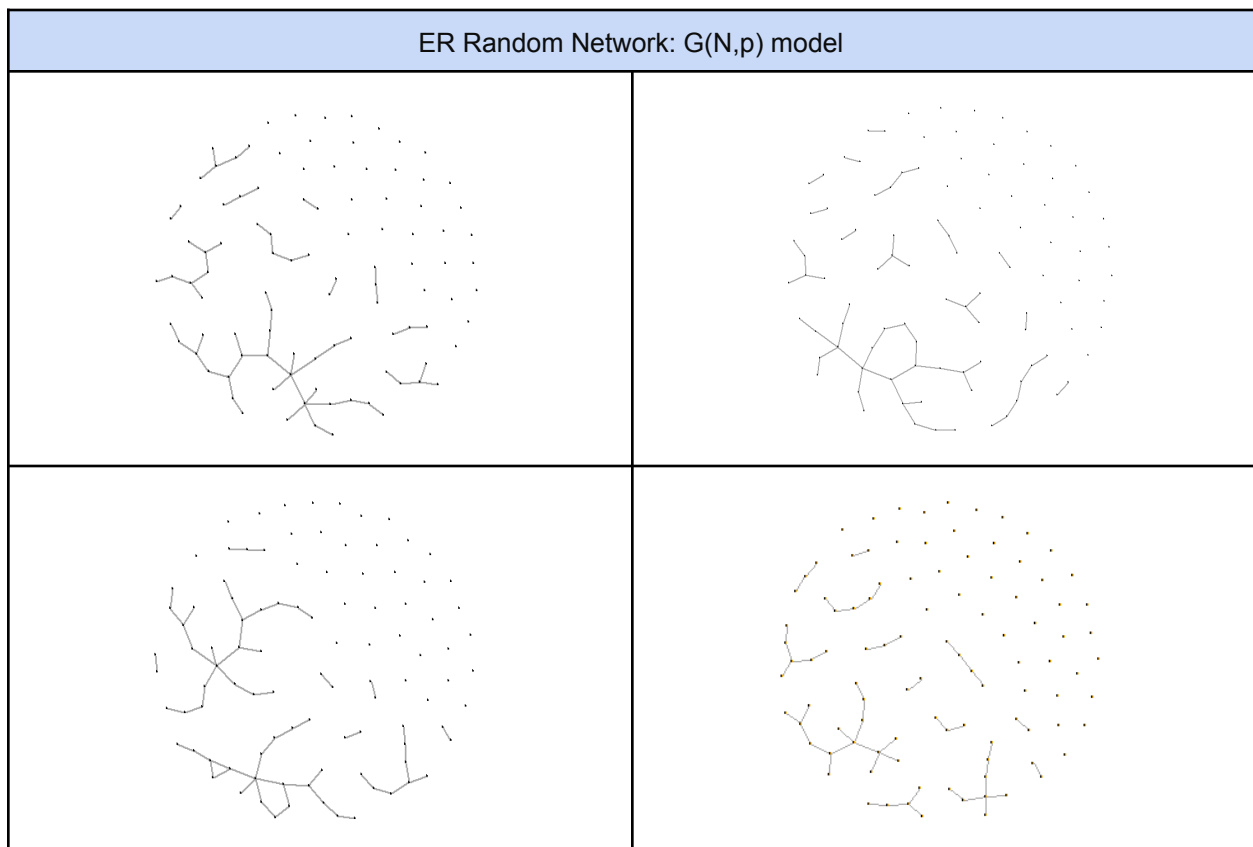
Question 11 (3 points)

Discuss with your group, then describe to your teacher, under which circumstances, we might need to work on a synthetic network.

Synthetic networks may be used to simulate real life networks, when only limited data exists. We can also research how synthetic networks are formed, and use this knowledge to learn more about how real life networks are formed, and what factors influence the formation. For example, we can simulate a network with different communities and then apply community detection algorithms to try to recover the networks. Another example can be when we want to investigate a large network that we cannot access due to privacy reasons. We can then use a synthetic network to substitute for this.

Question 12 (2 points)

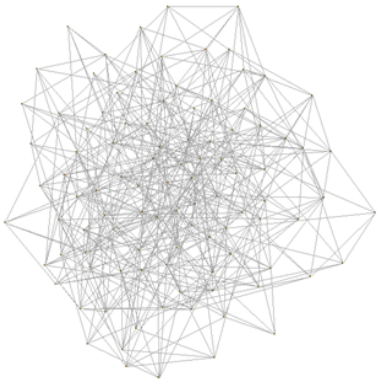
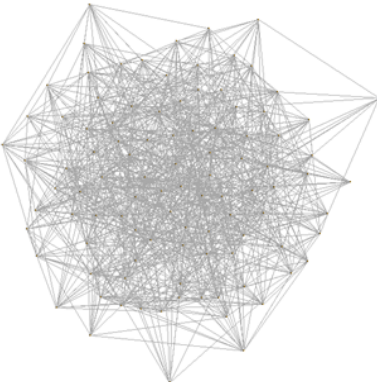
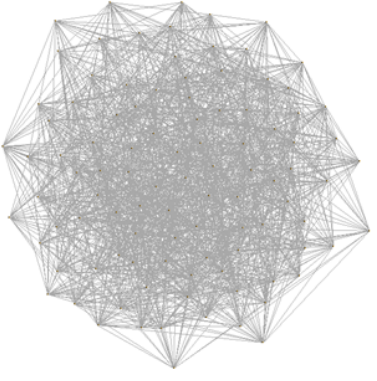
Plot your network (with $n=100$, and $p=1/100$) and compare with those with your group members. Are they identical? Explain why they are/aren't.



They are not identical, because a random graph is generated using the n and p values. The network starts with the nodes (n) and every pair is then connected randomly with probability (p) (Albert & Barabási, 2002). The number of vertices stays the same though, since that part is fixed in the model definition.

Question 13 (3 points)

Develop three networks with the same number of vertices (n), but different probability (p); Name them as ER1, ER2, and ER3. Develop the plots of ER1, ER2 and ER3, describe how these three graphs look differently as p increase and explain why.

ER1 Random Network: G(N,p) model	ER2 Random Network: G(N,p) model
	
ER1 <- sample_gnp(100, 0.1)	ER2 <- sample_gnp(100, 0.2)
ER3 Random Network: G(N,p) model	
	
ER3 <- sample_gnp(100, 0.3)	

The probability p gives the edge probability between two vertices. The higher the number p , the more edges there are between two vertices, that's why we see the network with a higher p value being more dense than the one with a lower p value. The network gets denser due to the increased number of edges, this will continue with an increasing p until it reaches 1 and all edges possible are drawn.

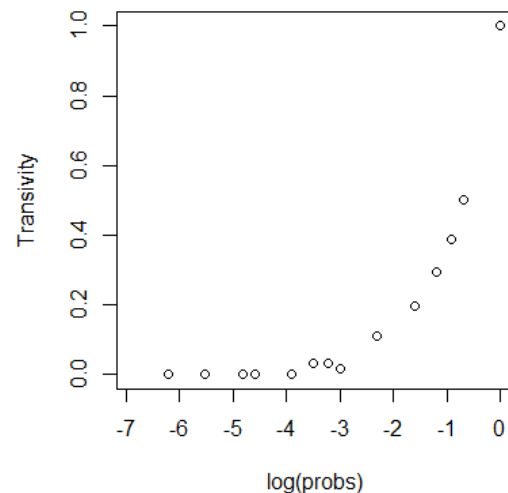
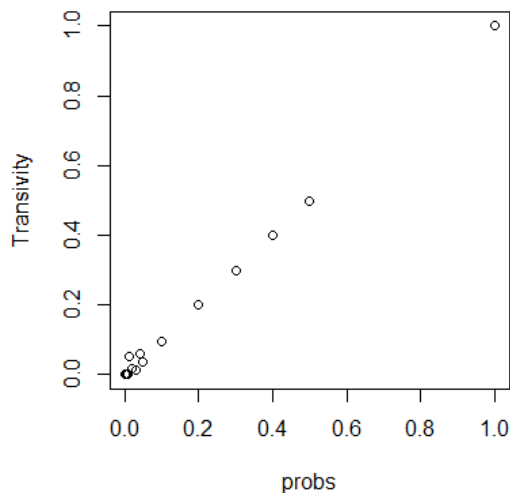
Question 14 (2 points)

If $p < 1$, for n great enough, what happens to the clustering coefficient of an ER random graph and why? (You can use the 'transitivity' function to test your guess). Discuss with your group and describe the answer to your teacher.

The clustering coefficient seems to increase linearly with the probability for edges to be present. The reason for this is that when more edges are present, the nodes will be more interconnected, and thus have a higher transitivity value. When the number of networks n is increasing the transitivity is getting close to p .

Code

```
probs <- c(1, 0.5, 0.4, 0.3, 0.2, 0.1, 0.05, 0.04, 0.03, 0.02, 0.01, 0.008, 0.004, 0.002, 0.001)
tra <- c()
for (i in 1:length(probs)){
  tra[i] <- transitivity(sample_gnp(100,probs[i]), type="global")
}
par(mfrow=c(1,2))
plot(probs, trans_p(probs), ylab="Transitivity")
plot(log(probs), trans_p(probs), ylab="Transitivity")
```

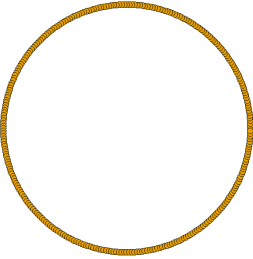
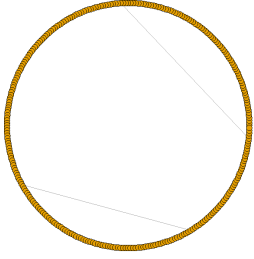
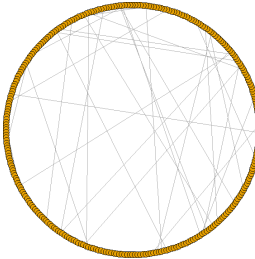
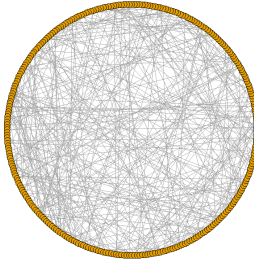


Applied to the ER1 ER2, ER3 networks we get:

Probability	Transitivity
1/25	0.028
1/15	0.056
1/10	0.098

Question 15 (3 points)

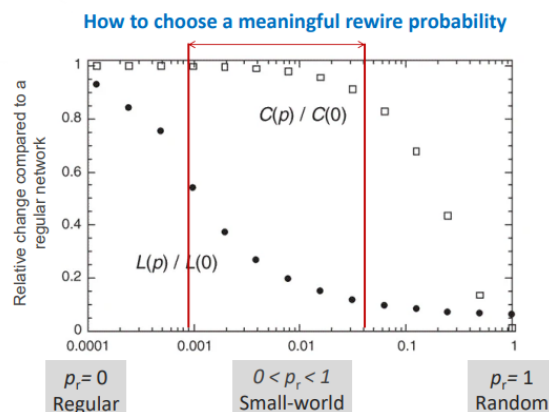
Check the clustering coefficient and average path length of the Regular, SW1, SW2 and SW3. Describe the trend of clustering coefficient and average path length as p increases. Does any of these graphs show the desirable attributes that you are looking for a small world network?

Regular	SW1	SW2	SW3
<p>Network with zero rewiring probability</p> 	<p>Network with 0.001 rewiring probability</p> 	<p>Network with 0.01 rewiring probability</p> 	<p>Network with 0.1 rewiring probability</p> 
<p>Clustering coefficient: transitivity(SW1): 0.68023431976568 transitivity(SW2): 0.640933844292053 transitivity(SW3): 0.3631625967838</p>		<p>Average path length: diameter(SW1): 22 diameter(SW2): 9 diameter(SW3): 5</p>	

The cluster coefficient and average path length decreases when the rewiring probability increases. The graph (SW1) with 0.001 rewiring probability has only two connections, so it is useless. The SW2 network looks better, but it still has not a lot of connections. The SW3 network with 0.1 has a lot of connections, but this might be too much. Small-world networks have a short average path length and a relatively high clustering coefficient (Albert & Barabási, 2002). Graph SW2 has a somewhat short average path length and high cluster coefficient, compared to SW1 and SW3. So SW2 does show somewhat the desirable attributes (low ASL and high transitivity) for a small world network. We see that the diameter is decreasing much faster compared to the transitivity. The ideal network will probably be somewhat in between the SW2 and SW3 probabilities ($0.01 < p < 0.1$).

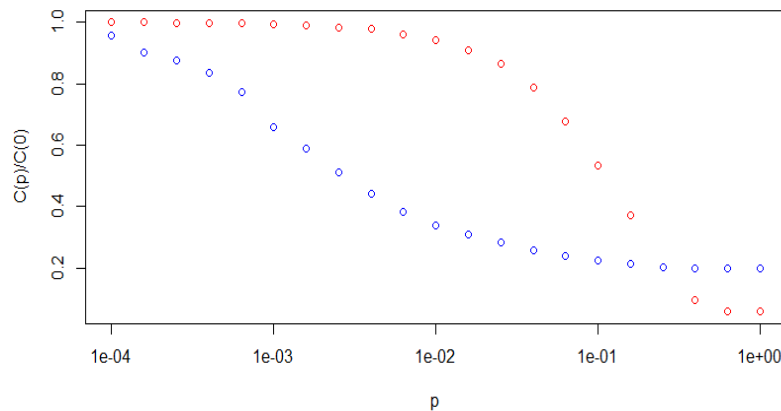
Question 16 (8 points)

For the same setting, i.e., size = 300, $nei=6$, what are the range of p that you will suggest to build a small-world network and why? One solution you can consider is to refer to the Figure 2 in Watts and Strogatz (1998), which explains how small-world networks can be created from randomly rewired graphs (similar figure also appeared in Lecture 5). Reproduce Figure 2 in the current context (i.e., size=300, $nei=6$) and find the right range of p in order to build a small-world network. Discuss with your group and show the answer to your teacher.



The graph (from slide lecture 6) shows that the suggested range of p to build a small-world network is between 0.01 and 0.1. Between these values for p , the path length is relatively low and the cluster coefficient is high.

Rewriting graph probability versus transitivity (red) and average path length (blue)



The randomly rewired graph for the current context (i.e size = 300, nei = 6) is produced see plot above (red = transitivity and blue = average.path.length). The right range of p in order to build a small-world network is where the path length is low and the cluster coefficient is high. The range of for the current context will be between 0.001 (1e-03) and 0.01 (1e-02).

Code

```
translist <- c()
ASLlist <- c()

# create bootstrap function to take average value of 20 random models for a given p
Tbootstrap <- function(p, i){
  transl <- c()
  for (i in 1:i){
    transl[i] <- transitivity(watts.strogatz.game(dim=1,size=300,nei=6, p=p))
  }
  return(mean(transl))
}

Abootstrap <- function(p, i){
  transl <- c()
  for (i in 1:i){
    transl[i] <- average.path.length(watts.strogatz.game(dim=1,size=300,nei=6, p=p))
  }
  return(mean(transl))
}

# find transitivity and ASL for all values of p
for (i in 1:21){
  nullmodel <- watts.strogatz.game(dim=1,size=300,nei=6, p=0)
  translist[i] <- Tbootstrap(p[i],20) / transitivity(nullmodel)
  ASLlist[i] <- Abootstrap(p[i],20) / average.path.length(nullmodel)}

plot(p,translist,type="p",col="red", log="x", ylab="C(p)/C(0)")
points(p,ASLlist,col="blue", log="x")
```

Question 17 (2 points)

As a follow-up question of Q16, do you find the p values of very large or relatively small? Do you need to rewire a significant amount of connections to make a network small-world-like?

As described in the lecture, a small world network has a probability between 0.01 and 0.1. A small world network also has a relatively low average path length and a high cluster coefficient. The graph created in question 16 has the p values (probability) between 0.001 and 0.01 for the small world network of the current context. As the graph of question 16 shows that for these values the average path length is small and the cluster coefficient is high and the p values are on the lower side between the range of 0 and 1, there is no need to rewire the significant amount of connections to make the network small-world-like.

Question 18 (4 points)

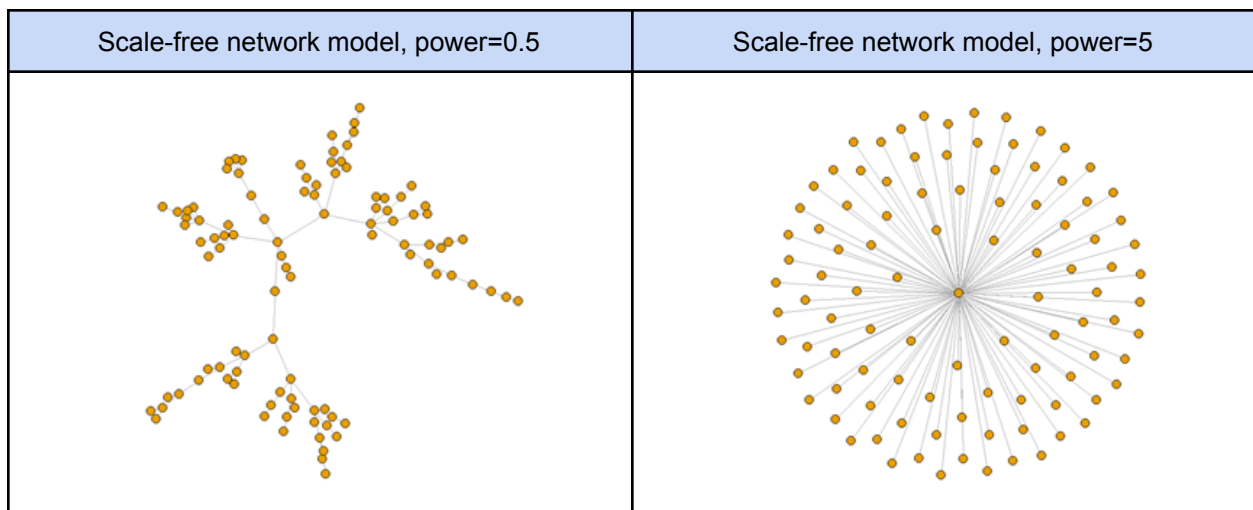
Use your own words to explain these two implications to your teachers (you are suggested to read the paper for a better understanding). For the second implication, try to explain it in the context of COVID.

- **First implication:** Watts and Strogatz found that a small number of shortcuts lead to a large decrease in average path lengths. They looked at three examples found in the real world and found that these networks all exhibited the 'small world' phenomenon. From this, they concluded that in natural networks with many vertices, a small fraction of shortcuts might be sufficient to connect all vertices with a relatively small degree of separation
- **Second implication:** In a small world network there are relatively short paths between nodes. Because of these short paths between the two nodes, infectious disease is predicted to spread much more easily and quickly. At time $t=0$, a single infected individual is introduced into the 'healthy' population. This infected individual can infect its neighbors, which are healthy individuals in the population. The neighbors of the infected individual are infected, and the disease spreads along the edges until the entire population is infected. In a small world network this is possible due to the few shortcuts that make the world small.

Question 19 (3 points)

What does the power in the above function mean? How can it govern the structure of the network (e.g., the formulation of hubs)? (Hint: Change the value of power from 0.05, 0.5, 1, 1.5; See how the plot evolves; if you still fail to see the difference, visualize the vertex size according to the edge number, you can consider the code below.)

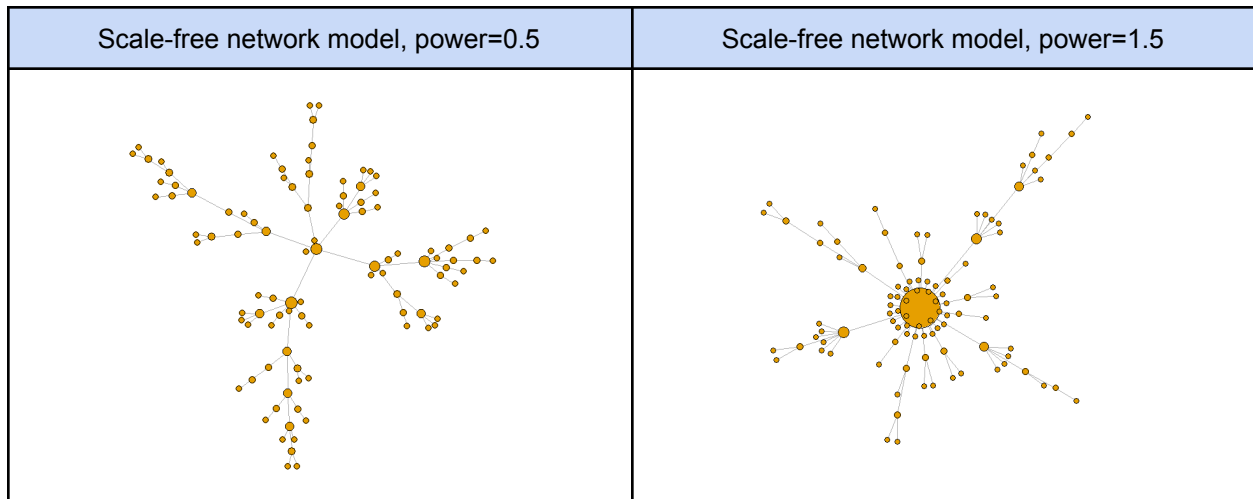
The power function indicates the power of the preferential attachment, where the default is one giving a linear preferential attachment. A higher value for power (power=5) ensures that all the nodes are connected to a single node, so all the nodes are connected to one certain hub. A lower number for power (power = 0.5) ensures that nodes are also connected to other nodes, and not specifically to one certain node. Here the nodes with a higher degree have a stronger ability to grab connections/links that are added to the network. With an increasing power in the function, we see more concentrated hubs. On average, the nodes have more connections with a higher power. It is probably related to the initial number of nodes.



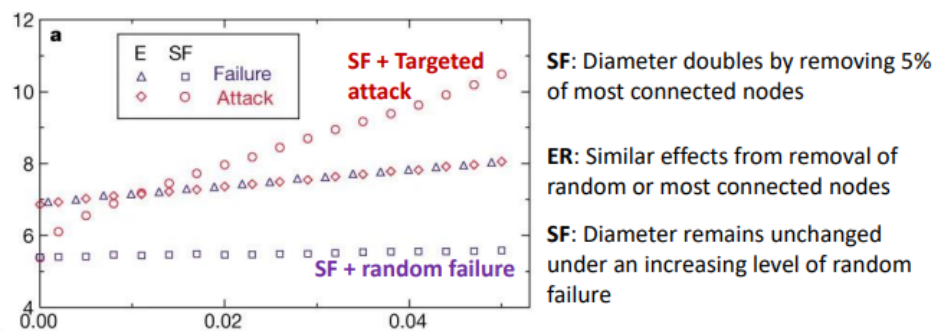
Question 20 (5 points)

Discuss with your groups, if you are maintaining a network with a power of 0.5 and 1.5, respectively, what will be your plans to build up resilience for 1) random attack, and 2) targeted attack? (the meanings of 'random attack' and 'targeted attack' are the same as what is mentioned in Lecture 6, scale-free network).

Below are the networks with a power of 0.5 and 1.5 plotted. As mentioned in question 19, a lower value for power ensures that nodes are not specifically connected to one certain node, and have more connections to other nodes. Where a higher value for power ensures that the nodes are specifically connected to one certain node.



In the lecture we saw how the diameter of the network progresses as the network is either targeted or randomly attacked. The results are shown in the picture below.



We see that for a random network it does not matter whether the attack is targeted or random, as the network diameter does not really change much. The diameter of the scale free network does not change when a random attack is done on the network. However, the diameter of the network does change when a targeted attack is done. For this reason, we think it's best to maintain a network with a power of 0.5 as this is more similar to a random network, and thus more resilient to both random and targeted attacks. The network with a power of 1.5 is more similar to the scale free network from the above picture. In this network the nodes are more connected to a central one, compared to the network with a power of 0.5 where the nodes are not connected to a central one. So for the network with power of 1.5 the diameter will increase when a targeted attack is done. The diameter of this network will not change under increasing level of random failure. This is different for the network with power 0.5, as the diameter for both the random and targeted attacks will not drastically change.

Question 21 (8 points)

Download the csv data of these networks from the BB, import the data to R and build the network. Check out their clustering coefficients, path lengths and degree distribution. Do you find these real networks show some attributes of the synthetic architectures we studied above (e.g., random ER graph, small-world, and scale-free network)? Show your teachers some numbers, plots and how you interpret the results.

Network	Clustering coefficients	Path lengths	Degree distribution
Brightkite	clustering coef: 0.11	APL: 4.92	Smallest distribution
Amazon	clustering coef: 0.21	APL: 4.47	Also small distribution
Collab	clustering coef: 0.63	APL: 6.05	Widest distribution

All distributions are asymmetric with a long right tail. A wide distribution is associated with a wide range of node degrees. The Collab network has the highest variety of node degrees, meaning its nodes have a wide range of connections. The Brightkite network has a small degree distribution, meaning its nodes have very similar degrees, there isn't much variation. The Amazon network has a somewhat larger distribution, but still very small compared to the Collab network. The amazon and brightkite networks have low transitivity, meaning that the networks aren't very well interconnected. The Collab network does have a higher clustering coefficient and thus its clusters are better connected.

These models do share some similarity with the Watts-Strogatz models in terms of APL and transitivity. The degree distribution looks very different in these models though. The barabasi models don't have any transitivity, so they are not like the models we have seen here.

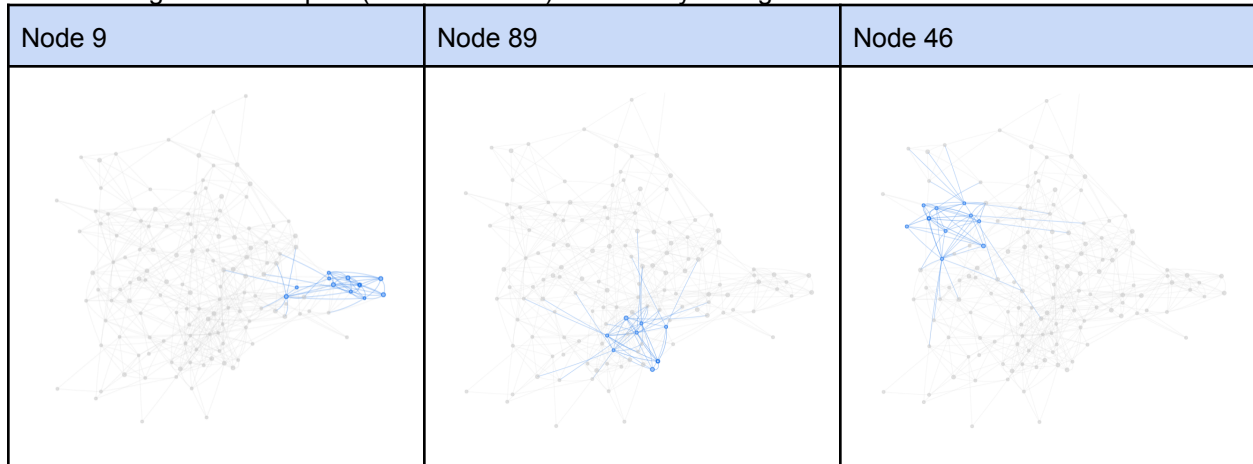
The cluster is low in the Amazon network, while the diameter/distance is high. The diameter increases with the size of the network, which is why we assume a random network due to low clustering. In the Collab network, the cluster is high, while the diameter/distance is low. The diameter increases with the size of the network, so we assume a small world for the Collab network only.

Exercise 3: Build the social network of this class and simulate the contagion process

Question 22 (5 points)

Open the html file to view the network of this class. Do you find a lot of strong ties in this network? Highlight three of them, and show to your teachers. Then check the clustering coefficient of this network, and does it match your observation (i.e., this network has a few/many strong ties)?

The following three examples (of some nodes) have many strong ties in this network:

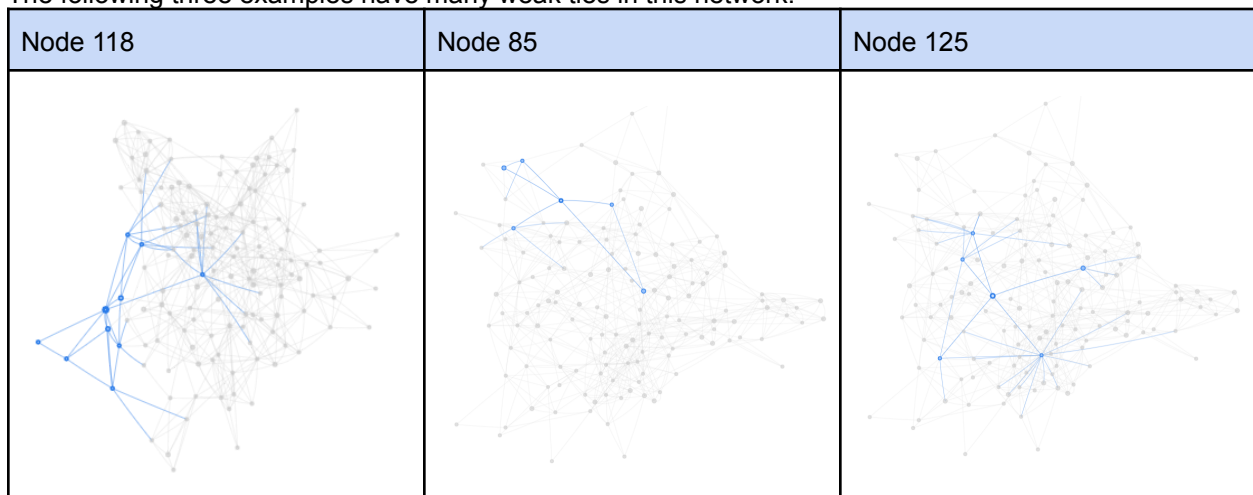


First impression of the neural network is that this network has a lot of strong ties in the network, because a lot of nodes are tied within closed triangles in the network. Some of the strong ties in the network don't have an extremely high number of connections. The following code is used to check the clustering coefficient of the network: `transitivity(class2022)` this gives a value of 0.3919. A perfect small-world network could reach 0.6, while for a random network it would be lower than 0.1. The value of 0.3919 for clustering coefficient indicates that the network has quite strong ties in this network. This matches our first observation, where we mentioned that our first impression was that the network has strong ties. The network is fully connected, indicating that there are many strong ties.

Question 23 (2 points)

Find out the weak ties in this network. Highlight three of them, and show them to your teachers.

The following three examples have many weak ties in this network:






Question 24 (5 points)

Following the above question, can you find out the nodes with many weak ties? One possible way is to find the nodes with high betweenness, using the below code. Find out 3 nodes with highest betweenness. Check their positions in the network again to see if they have many weak ties as predicted. Show your teacher the IDs of these 3 nodes, and explain why betweenness can be used as a proxy to find out nodes with many weak ties?

Code:

```
btw = betweenness(class2022, v=V(class2022), directed = FALSE, weights = NULL)
print(tail(sort(btw), 3))
```

The 3 nodes with the highest value for betweenness:

Node 80: 540.783649183949	Node 71: 546.028622378941	Node 20: 549.743442273848
		

Betweenness can be used as a proxy to find out nodes with many weak ties, because it captures a node's role in allowing it to pass information from one part of the network to the other. Betweenness is a measure of how often the shortest path crosses that node. It is essential to use these nodes with high betweenness, because they are one of the few nodes that are able to get to the nodes with weak ties. In order to spread information from one part of the network to another part, we need to go through the nodes with many weak ties. Weak ties in a network act like this, these nodes act as a bridge across communities.

Question 25 (10 points)

After you build the model, apply it to the network of this class, and develop a plot with x-axis as Day, y-axis as the Cumulative Infected Percentage by Day. (You can use the code below to build the IC model or develop your own codes.)

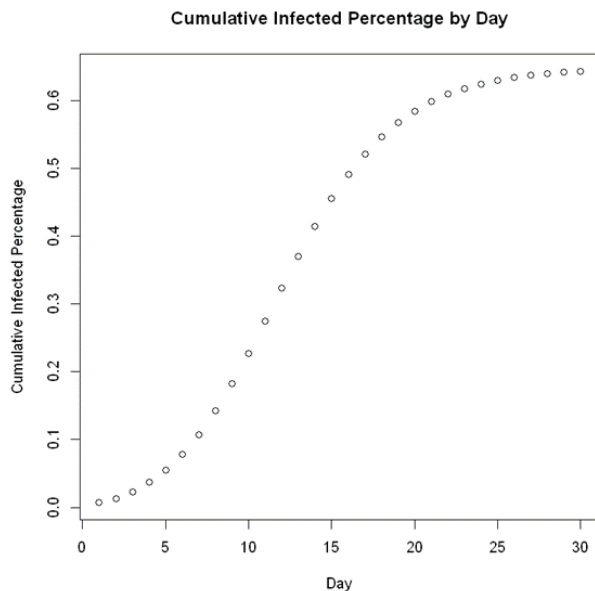
Code:

```
seeds<-c(1:140)
nNode=140 #size of the original network
Pprob=0.1 #the probability for an infected person to pass down the virus to his/her neighbors

# apply the given code on the network
adj_matrix <- igraph::as_adjacency_matrix(class2022, type = 'both')
each_neighbors <- which(adj_matrix > 0, arr.ind = TRUE)
#get the neighbor list of each node
each_neighbors <- split(each_neighbors[, 2], each_neighbors[, 1])
result1<- replicate(100, IC2(140,seeds,nNode,each_neighbors,Pprob), simplify=FALSE)
result1<-do.call(rbind, result1)
# percentage of infected person each day
percentage_per_day <- colMeans(result1)/(140)

# plot x-axis as Day and y-axis as Cumulative Infected Percentage
plot(paste0("Day", 1:30), percentage_per_day, ylab="Cumulative Infected Percentage", main = 'Cumulative Infected Percentage by Day')
```

The following plot of the 'Cumulative Infected Percentage by Day' is developed using the given code. The x-axis describes the days passed, and the y-axis the total % of infected nodes per day.



The values that are plotted in the plot 'Cumulative Infected Percentage by Day' are given below.

```
[1] 0.007142857 0.012811735 0.022190816 0.036653061 0.054493367 0.077645918
[7] 0.106550000 0.141336224 0.181663265 0.226769388 0.274658163 0.323890816
[13] 0.371822959 0.416862245 0.457720918 0.493543367 0.524608163 0.550822959
[19] 0.572237755 0.589593367 0.603548980 0.614573469 0.623220408 0.630036735
[25] 0.635426531 0.639442347 0.642654082 0.645087755 0.646959694 0.648448980
```

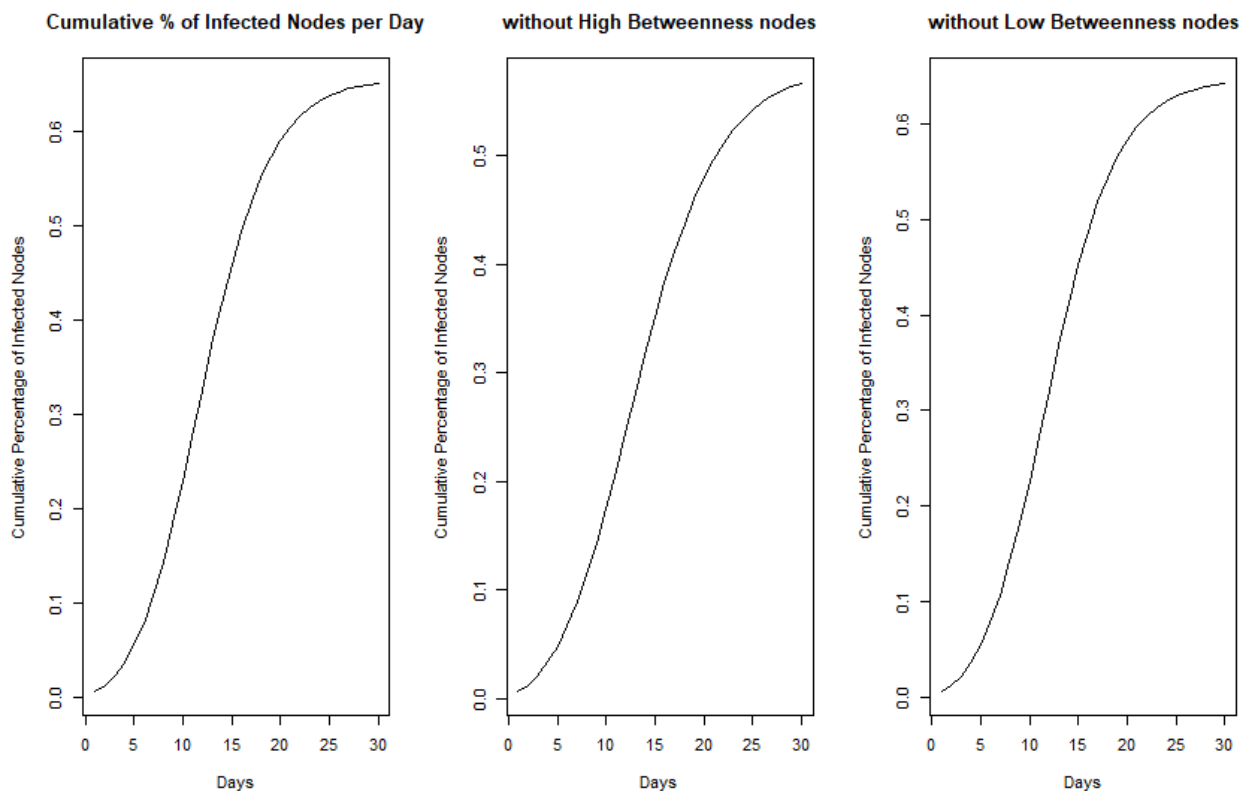
Question 26 (15 points)

Revised the original network of “class2022” by deleting the 3 vertices with highest betweenness (edges directly connecting to these 3 vertices are deleted). Apply the same IC model to this new network, how does the daily cumulative infected percentage change? What if you delete the 3 vertices with lowest betweenness? Show the results to your teacher by plots, and explain the results by “the strength of weak ties”.

Code:

```
del_highBetween <- delete_vertices(class2022, c(20,71,80))
del_lowBetween <- delete_vertices(class2022, c(25, 43, 78))

# plot the different graphs
par(mfrow=c(1,3))
plot(1:length(CP), CP, type="line", xlab="Days", ylab="Cumulative Percentage of Infected Nodes",
     main="Cumulative % of Infected Nodes per Day")
plot(1:length(CP_H), CP_H, type="line", xlab="Days", ylab="Cumulative Percentage of Infected Nodes",
     main="without High Betweenness nodes")
plot(1:length(CP_L), CP_L, type="line", xlab="Days", ylab="Cumulative Percentage of Infected Nodes",
     main="without Low Betweenness nodes")
```



Cumulative % of Infected Nodes per Day

```
[1] 0.007142857 0.012811735 0.022190816 0.036653061 0.054493367 0.077645918
[7] 0.106550000 0.141336224 0.181663265 0.226769388 0.274658163 0.323890816
[13] 0.371822959 0.416862245 0.457720918 0.493543367 0.524608163 0.550822959
[19] 0.572237755 0.589593367 0.603548980 0.614573469 0.623220408 0.630036735
[25] 0.635426531 0.639442347 0.642654082 0.645087755 0.646959694 0.648448980
```

without High Betweenness nodes	[1] 0.00729927 0.01276573 0.02137567 0.03411210 0.04881347 0.06722148 [7] 0.08912675 0.11459961 0.14352869 0.17555331 0.21002344 0.24586446 [13] 0.28203367 0.31770100 0.35154990 0.38336512 0.41222867 0.43817572 [19] 0.46097608 0.48092386 0.49788055 0.51247269 0.52469604 0.53493260 [25] 0.54340988 0.55045394 0.55636528 0.56117960 0.56513773 0.56833289
without Low Betweenness nodes	[1] 0.00729927 0.01314295 0.02279130 0.03766370 0.05605680 0.07971496 [7] 0.10911556 0.14407054 0.18430763 0.22880654 0.27619053 0.32447067 [13] 0.37147211 0.41556876 0.45543929 0.49084927 0.52133998 0.54707283 [19] 0.56837658 0.58594597 0.59993926 0.61103096 0.61983483 0.62665566 [25] 0.63202248 0.63611647 0.63920720 0.64164260 0.64350738 0.64494592

Removing the high betweenness nodes significantly impacts the total % of infected nodes, changing it from 65.1 -> 56.7. Removing the low betweenness nodes on the other hand, changes barely anything (65.1 to 64.2). This simulation demonstrates the strength of weak ties. In these kinds of networks, most new information is shared via the weak ties. Strong ties have many connections to other nodes, and so if the node were to disappear, we would still get the information through the other strong ties. If a weak tie breaks however, there are no other ways to get the information. This is why weak ties are very important in networks.

Question 27 (3 points)

Inspired by your observations from Question 26, can you relate it to some of the COVID measures implemented in the real world?

Family and close friends can be seen as strong ties. While people who you see occasionally are weak ties. When the lockdown started, we were still able to see our close friends and family (the strong ties). These people won't transmit the virus as much as the weak ties will. The lockdown is effective because it makes it harder to come into contact with the weak ties. Friends you see occasionally, distant family members and for example class mates will not come into contact with you as much. This helps slow down the spread of a virus. Another example is when the flights were banned, people were not able to travel to other countries. The people that you meet on vacation are considered weak ties, with whom you don't interact much as you are once in a while in your travel destination. This covid measure ensured that people don't go to new destinations and meet new people (come in contact with weak ties), and that is also effective to slow down the spread of a virus.

Question 28 (2 points)

Recall the fourth and fifth questions in the survey as shown below. They are designed to collect the parameters that we will need to build 1) an independent cascade (IC) model and 2) a threshold model. Can you see which question is for the IC model and which one is for the threshold model?

- Question 4 of the survey is for the independent cascade (IC) model, there is no threshold necessary here, just the chance of transmission. Where you have to answer how likely you will share your exploration of something new to other students in the class, indicating the transmission of information.
- Question 5 of the survey is for the threshold model. There is a certain activation threshold that only gets activated when enough people recommend it to you, so after you hear it from x number of students where x indicates the activation threshold.

Question 29 (3 points)

We asked your response for three types of behaviors (i.e., Share/watch Youtube video, Share/try a vegetarian recipe, Share/read a paper related to the lecture). The distribution of your answers are shown in the above figure. Among these three types of behaviors, which one is the least contagious? Which one is the most contagious? And why?

- Share/watch Youtube video is the least contagious because has a lower threshold compared to the other types of behavior where single contact sufficient for transmission.
- Share/read a paper related to the lecture is the most contagious one, because of the relatively higher threshold for activation where multiple (and credible) sources are required for transmission.

Question 30 (10 points)

Based on the model that you've built, if you can seed only one person, who will you choose? Explain to your teacher 1) how do you find out this node, 2) the ID of this node 3) its attributes (e.g, degree, betweenness, probability to share), and compared such attributes to other nodes in the network to explain why it is a good candidate to start the diffusion.

Model that is built:

Code:

```
seeds<-c(1:140)
nNode=140
class2022_attributes <- readRDS("class2022_attributes.rds")
class2022_pYoutube <- class2022_attributes %>% select(1:2)

#build an independent cascade (IC) model to simulate the spread of a Youtube video in this class:
stopifnot(require(data.table))
stopifnot(require(Matrix))

#search the neighbours of contagious node, and their neighbours have a chance of Pprob being infected
calculate_value <- function(node, each_neighbors,Pprob){
  return(each_neighbors[[node]][ which(runif(length(each_neighbors), 0, 1) <= Pprob[[node]])])
}

#function to model simple contagion, return can be modified if you want to see other outputs
IC <- function(node_seed,each_neighbors,Pprob){
  node_seed2 <- node_seed
  node_value <- rep.int(0, nNode)
  #each node in the network has two status: watched (value=1) or hasn't watched (value =0)
  #at day 0, all the nodes in the network haven't watched the video (value=0)
  watched <- rep.int(0, nNode)
  new_watched <- list()

  #at day 1, some seed nodes (Ns) are introduced to the network (you showed them the video)
  day <- 1
  day_watched <- 1
  new_watched[[day]] <- node_seed2
  node_value[as.numeric(node_seed2)] <- 1
  watched[as.numeric(node_seed2)] <- 1

  #at the following days, students who have watched the video will share it with their neighbours with a probability
  #once watched, the status of this node will remain as 1 till the end

  for (day in c(2:30)){
    old_watched <- which(watched == 1)
    #always contagious
    ContagiousID <-which(watched == 1)
    activeID<-unlist(lapply(ContagiousID,calculate_value,each_neighbors,Pprob))
```



```

newactiveID <- setdiff(activeID, old_watched)
if (length(activeID)!=0){
  new_watched[[day]] <- newactiveID
  watched[as.numeric(newactiveID)] <- 1
} else {new_watched[[day]]<-integer(0)}
day_watched[day] <- sum(watched == 1)
day=day+1
}
return(day_watched)
}

IC2<-function(node_seed,each_neighbors,Pprob,niter){
  result<- replicate(niter, IC(node_seed,each_neighbors,Pprob), simplify=FALSE) #run niter times since each IC model run has it
  own randomness (see the "calculate_value2 function")
  result<-do.call(rbind, result)
  return(round(colMeans(result) ,2))
}

greedy_ICyoutube <- function(each_neighbors,Pprob,niter,k){
  nNode <- length(each_neighbors)
  SeedCan <- c(1:nNode) # nodes that can still be used as seeds
  Seedset <- c() # nodes that will be included in the final seed sets

  for (s in 1:k){
    best_seed <- 1 # initial setting of each round
    best_spread <- -Inf

    N <- length(SeedCan)
    for (i in 1:N){
      current_seed <- SeedCan[i]
      current_seedset <- c(Seedset,current_seed)
      current_spread <- IC2(current_seedset,each_neighbors,Pprob,niter)
      if (sum(current_spread) > sum(best_spread)){
        best_seed <- current_seed
        best_spread <- current_spread
      }
    }
    Seedset <- c(Seedset,best_seed)
    SeedCan <- -setdiff(SeedCan,Seedset)
  }
  return(Seedset)
}

adj_matrix <- igraph::as_adjacency_matrix(class2022, type = 'both')
each_neighbors <- which(adj_matrix > 0, arr.ind = TRUE)
each_neighbors <- split(each_neighbors[, 2], each_neighbors[, 1])

Pprob <- class2022_pYoutube$pYoutube
greedy_seeds1 <- greedy_ICyoutube(each_neighbors,Pprob,100,1)
greedy_seeds1

```

By building the independent cascade (IC) model above we assumed that the diffusion of Youtube video follows simple contagion, so a single contact is sufficient enough for transmission. Based on this model, node 65 is chosen to promote the Youtube video to its peers in the class. This person is chosen, because the greedy algorithm show that this person is the best option for spreading information. The node of this person has the following attributes:

ID of this node:	65
degree(class2022, v = 65, mode = "all")	13

betweenness(class2022, v = 65, directed = FALSE, normalized = TRUE)	0.03590310598253
Probability to share = class2022_attributes\$pYoutube[65]	0.425
transitivity(class2022, type = "localundirected", vids = 65)	0.230769230769231
closeness(class2022, v=65, normalized = TRUE)	0.396011396011396
eigen_centrality(class2022)\$vector[65]	0.820521782775765

Comparing these attributes to other nodes in the network give that this node is a good candidate to start the diffusion, because this node is one of the nodes with the highest value for probability (0.425) to share. Beside this, the node also has a high value (0.8205) for the eigenvector centrality giving that this node is pointed to by many other nodes in the network. This gives that node 65 (this person) is relatively popular in the class, that's why the high value for the eigenvector centrality.

Question 31 (10 points)

If you can seed only one person to spread the recipe, who will you choose? Explain to your teacher 1) how do you find out this node, 2) the ID of this node 3) its attributes (e.g, degree, betweenness, threshold), and compared such attributes to other nodes in the network to explain why it is a good candidate to start the diffusion.

Model that is built:

Code:

```
# threshold model to simulate the spread of a vegetarian recipe in this class
calculate_infected <- function(node, infection, each_neighbors){
  return(mean(infection[each_neighbors[[node]]] == 1)) #### percentage with infection
}

LTRecipe<-function(node_seed,each_neighbors,threshold){ #i, the node ID of seed
  nNode<-length(each_neighbors)

  node_value <- rep.int(0, nNode) #### node_value
  infection <- rep.int(0, nNode) #### infection status 0 uninfected 1 infected
  node_value[as.numeric((node_seed))] <- 1 #### assign value for seed nodes
  infection[as.numeric((node_seed))] <- 1 #### assign infection status for seed nodes
  new_infected <- list()

  #####
  day_infected <- rep(0,20) #### only simulate the first 20days
  day_infected[1]=sum(infection == 1)
  #####

  day <- 1
  max_day <- 20
  Diff=1

  while(day <= (max_day - 1) ){
    if (Diff==0){day_infected[(day+1):20]=day_infected[day] #can stop before D20 if no more nodes can be
    activated
    break}
    not_infected <- which(infection == 0)
    old_infected <- which(node_value > threshold)
    node_value[not_infected] <- unlist(lapply(not_infected, calculate_infected,
```

```

        infection, each_neighbors))
new_infected[[day+1]] <- setdiff(which(node_value > threshold), old_infected)
infection[new_infected[[day+1]]] <- 1 ### if exceed threshold, infected
day <- day + 1
day_infected[day] <- sum(infection == 1)
Diff<-day_infected[day]-day_infected[day-1]
}
return(day_infected)
}

# greedy algorithm
greedy_LTRecipe<-function(each_neighbors,threshold,k){#if you want to select 5 seeds, k=5
nNode<-length(each_neighbors)
SeedCan<-c(1:nNode) # nodes that can still be used as seeds
Seedset<-c() # nodes that will be included in the final seed sets

for (s in 1:k){
  best_seed<--1 # initial setting of each round
  best_spread <- -Inf

  N<-length(SeedCan)
  for (i in 1:N){
    current_seed<-SeedCan[i]
    current_seedset<-c(Seedset,current_seed)
    current_spread<-LTRecipe(current_seedset,each_neighbors,threshold)
    if (sum(current_spread)>sum(best_spread)){
      best_seed<-current_seed
      best_spread<-current_spread
    }
  }
  Seedset<-c(Seedset,best_seed)
  SeedCan<-setdiff(SeedCan,Seedset)
}
return(Seedset)
}

class2022<-readRDS("classnetwork_2022.rds")
Att<-readRDS("class2022_attributes.rds")
Att<-as.data.frame(Att)
tRecipe<-Att$tRecipe

adj_matrix <- igraph::as_adjacency_matrix(class2022, type = 'both')
each_neighbors <- which(adj_matrix > 0, arr.ind = TRUE)
each_neighbors <- split(each_neighbors[, 2], each_neighbors[, 1])

greedy_LTRecipe(each_neighbors, tRecipe,1)

n<-106
degree(class2022, v = n, mode = "all")
betweenness(class2022, v = n, directed = FALSE, normalized = TRUE)
Threshold = Att$tRecipe[n]
transitivity(class2022, type = "localundirected", vids = n)
closeness(class2022, v =n, normalized = TRUE)
eigen_centrality(class2022)$vector[n]

```

ID of this node:

106

degree(class2022, v = 106, mode = "all")	15
betweenness(class2022, v = 106, directed = FALSE, normalized = TRUE)	0.052
Threshold = class2022_attributes\$tRecipe[106]	0.67
transitivity(facebook, type = "localundirected", vids = 106)	0.181
closeness(facebook, v = 106, normalized = TRUE)	0.394
eigen_centrality(class2022)\$vector[106]	0.935

This node was selected by using a greedy algorithm to find the highest spread. Every node was selected as a first node, and the node with the highest spread was found to be the best initial node. When we look at the attributes of other nodes, we can begin to understand why. The node has a high degree compared to the other nodes, it belongs to the top 4 highest degrees out of 140. This means that it is connected to many nodes, and so it has a high initial diffusion. It also belongs to the top 4 highest betweenness nodes, while having a relatively low threshold. A high betweenness means that is a vital node for sharing information, since information often has to go through it in order to reach all nodes with the shortest path length. Finally, a high closeness centrality means that it is relatively close to all other nodes in the network.

Question 32 (2 points)

To answer Q31, you might search explicitly to remove the 140 nodes one by one from the network. If you want to find 5 people to seed, can you apply such explicit search again?

To seed only one person from the 140 nodes, we have to run the model 140 times in order to find this one node. The formula for this is: $n! / r! (n - r)!$, with $n = 140$ and $r = 1$. So, in order to find this single person to seed, the model has to run 140 times. So if we want 5 people to seed, where the order in which the person is selected does not matter and the person cannot be seeded twice (no repetition), the model has to run $140! / 5! (140 - 5)! = 140 * 139 * 138 * 137 * 136 / (5 * 4 * 3 * 2) = 416.965.528$ times. So, no this explicit search cannot be applied to find 5 people to seed.

Question 33 (20 points)

Now you have a little more time or budget to seed more than 1 person in this class. To promote the Youtube video, you can seed 3 people. To promote the vegetarian recipe, you can seed 5 people. Use degree heuristics, betweenness heuristics and greedy algorithms to find out 1) 3 seeds to spread Youtube videos and 2) 5 seeds to spread the vegetarian recipe. Show your teachers 1) the ID of these seeds, 2) Among degree heuristics, betweenness heuristics and greedy algorithms, which method provides the best result? 3) Do you find some common properties of the seeds provided by different methods?

3 seeds to spread Youtube videos:

Code:
<pre># degree heuristics tail(sort(degree(class2022)), 3) # betweenness heuristics tail(sort(betweenness(class2022)), 3) # greedy algorithms</pre>

```
# same code as used for question 30
Pprob <- class2022_pYoutube$pYoutube
greedy_seeds3 <- greedy_ICyoutube(each_neighbors,Pprob,100,3)
greedy_seeds3
```

5 seeds to spread the vegetarian recipe:

Code:

```
# degree heuristics
tail(sort(degree(class2022)), 5)

# betweenness heuristics
tail(sort(betweenness(class2022)), 5)

# greedy algorithms
# same code as used for question 31
greedy_LTrecipe(each_neighbors, tRecipe,5)
```

Use degree heuristics, betweenness heuristics and greedy algorithms to find out 1) 3 seeds to spread Youtube videos and 2) 5 seeds to spread the vegetarian recipe. The ID of these seeds are given below:

	3 seeds to spread Youtube videos	5 seeds to spread the vegetarian recipe
degree heuristics	Nodes ID: 20, 80, 71	Nodes ID: 20, 80, 71, 106, 2
betweenness heuristics	Nodes ID: 20, 71, 80	Nodes ID: 20, 71, 80, 106, 2
greedy algorithms	Nodes ID: 65, 122, 107	Nodes ID: 106, 79, 53, 50, 126

From the results, we can see the degree and betweenness have similar behavior as they have shown similar results. The heuristics algorithms do not intend to find the best solution, but it terminates in a reasonable solution. On the other hand, a greedy algorithm tries to make the locally optimal choice at each stage, but sometimes a greedy strategy does not produce an optimal solution. In this case, the greedy algorithm has performed very differently from the heuristics, only overlapping node 106 for the vegetarian recipe.

The greedy algorithm makes choices based on what looks best at the moment. The decision is mainly made on the basis of currently available information without worrying about the effects of the future. Where degree gives you the node that has the number of connections with other nodes. And betweenness gives the node that is most important for the flow of information. Depending on the case/scenario you might find the degree heuristics, betweenness heuristics or greedy algorithms the best. For the sprees of

Reference

Albert, R., & Barabási, A. L. (2002). Statistical mechanics of complex networks. *Reviews of Modern Physics*, 74(1), 47–97. <https://doi.org/10.1103/revmodphys.74.47>

Jayawickrama, T. D. (2021, December 29). Community Detection Algorithms - Towards Data Science. Medium. Retrieved 30 March 2022, from <https://towardsdatascience.com/community-detection-algorithms-9bd8951e7dae>

Joshi, P. (2020, April 13). Getting Started with Community Detection in Graphs and Networks. Analytics Vidhya. Retrieved 30 March 2022, from <https://www.analyticsvidhya.com/blog/2020/04/community-detection-graphs-networks/>