28462000 - Bourez Nicolas
38502000 - Demaret Dimitri
77102000 - Lhernould Benoit

LINMA2472 - Homework 1 - Group 8

1 Introduction

In our increasingly interconnected society, network analysis is more important than ever. From social interactions on social media platforms to the distribution of goods and services through supply chains, the dynamics of networks profoundly influence our decisions, relationships, and overall well-being. To represent this problem, we built a graph from the script of a famous movie: Star Wars IV. We justified this choice by the relevant amount of distinct communities present in this movie. The number of nodes (56), representing the characters, and the number of edges (154), linking characters who talk to each other, are big enough to perform a good network analysis, but not too high which allows us to perform clear visualizations.

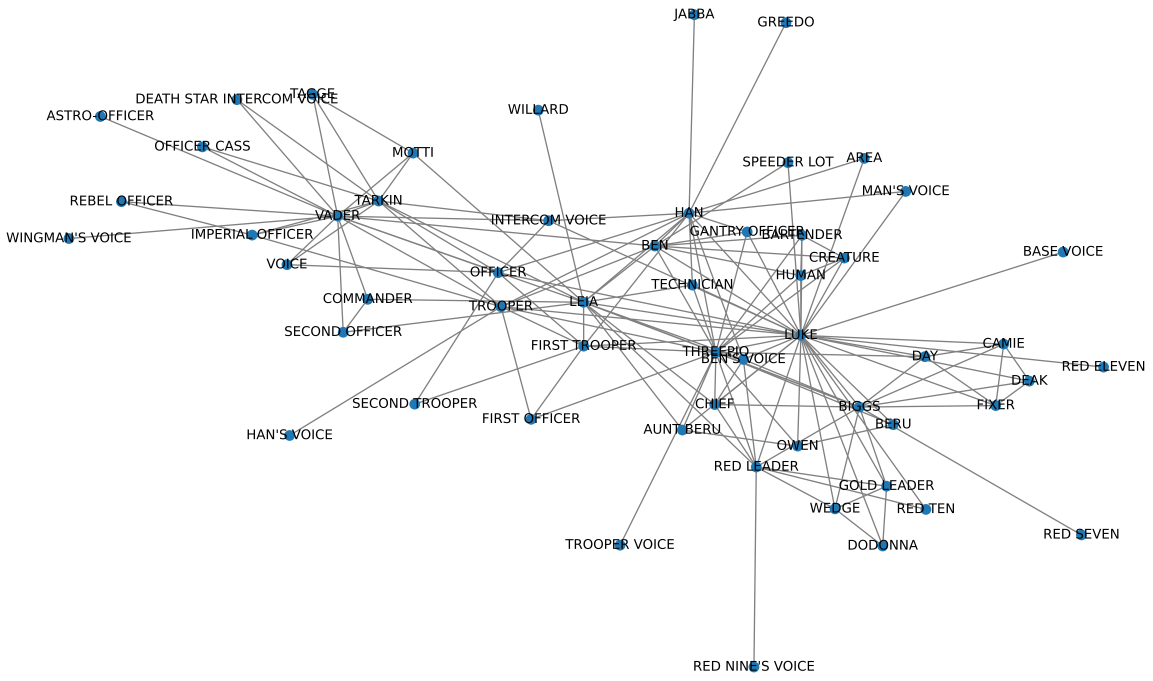


Figure 1: Visualization of the network

The position of the nodes in this graph is totally random. However, if we look at the degree of the nodes, some of them stand out from the others like **Luke, Vader** and **Leia**. At this moment of the analysis, we can only guess that these characters are the main ones and that they have an important role in their community. We can also notice some nodes have only one edge, which suggests that these characters are not important and that they appear only once in the story. Nevertheless, just by looking at the graph, it is hard to guess the exact number and who is part of each community.

To gain a better understanding of the 'Star Wars IV' network and the exact communities, we will perform a more comprehensive analysis. Our approach includes examining degree assortativity to unveil the relationships between characters based on their degree centrality. We will apply the Louvain algorithm, a famous community detection method, and spectral clustering.

In the subsequent stages of the report, we will explore the dynamics of information propagation within the network. We will employ algorithms such as the Independent Cascade Model to simulate how information and influence flow through the network. Finally, we will compare it to a Barabasi-Albert network.

2 Degree assortativity of the network

Definition

In the context of networks, assortativity is generally measured as the correlation coefficient between the degrees of nodes at each end of an edge. Mathematically, for an undirected graph, the assortativity coefficient r is given by:

$$r = \frac{\sum_i j_i k_i - \frac{1}{2m}\left(\sum_i (j_i + k_i)\right)^2}{\sum_i (j_i^2 + k_i^2) - \frac{1}{2m}\left(\sum_i (j_i + k_i)\right)^2}$$

where j_i and k_i are the degrees of the nodes at each end of the i^{th} edge, and m is the total number of edges.

Interpretation

- $r > 0$: The network is assortative. This means that high-degree nodes tend to be connected to other high-degree nodes.
- $r = 0$: There is no assortative correlation in the network.
- $r < 0$: The network is disassortative. High-degree nodes tend to be connected to low-degree nodes.

Result for our graph

In our case we have a degree assortativity coefficient of -0.351, which indicates that the network is disassortative. In a disassortative network, nodes with high degrees tend to be connected to nodes with low degrees. A disassortative structure indicates that influential individuals (high-degree nodes) tend to connect with less influential individuals (low-degree nodes).

3 Louvain method

The Louvain method is an algorithm which aims to find the best partition of a graph by optimizing the **modularity**.

Modularity assesses the quality of a graph partitioned into communities, emphasizing strong internal connections and weak links between communities. It quantifies the difference between actual internal edges and those expected by chance. Modularity values range from -1/2 to 1, with a value exceeding 0.3 indicating a significant community structure within the graph. The Louvain algorithm has a bit of randomness, the communities depend on the order in which it parcours the nodes. But by repeatedly using the Louvain algorithm, we can detect similar characteristics of communities.

After performing the Louvain algorithm more than a dozen times, we clearly saw some patterns in the repartitions of communities. We now have a better idea of the communities present in Star Wars IV. We can spot 5 communities:

- The community of Darth Vader. We can see that he has a lot of interactions with officers and Stormtroopers, which is the case in the narrative.
- The community of Luke and 3-CPO, which contains people from the resistance, according to the story
- The community of the Woman and the Death Stars Intercom, which has a whole scene in the film where they all talk to each other.
- The community of Biggs. It's a pilot in the rebel army. He has also people from the resistance in his community, as well as some enemies.
- The community of the bartender and Leia. His community consists of characters related to the Mos Eisley Cantina, including the bartender, other cantina patrons, or characters who have interactions in this place.

We compute a modularity of 0.392 which is higher than 0.3. We are still far from 1 because there are a lot of interactions between the communities as shown by the high number of edges between each group.

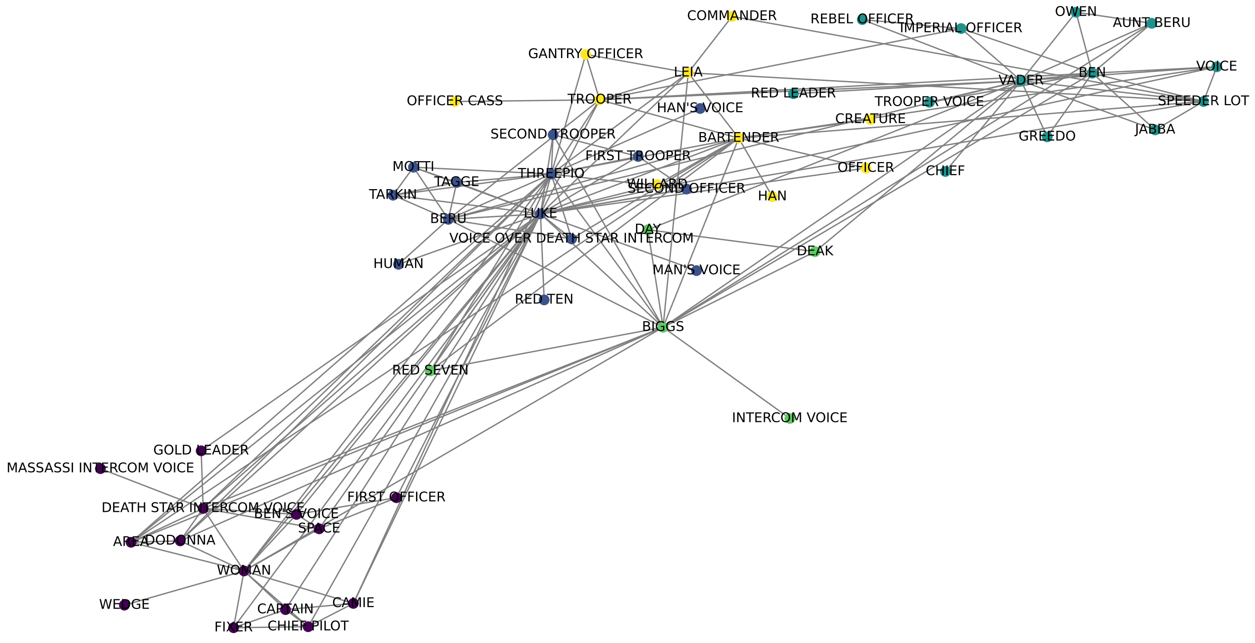


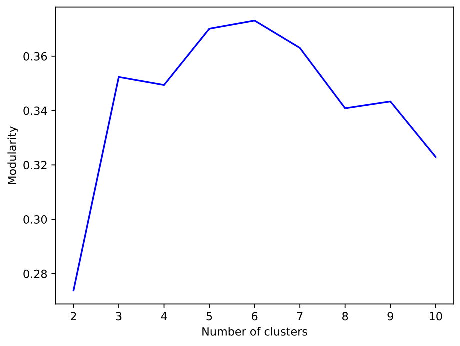
Figure 2: Visualization of the network

4 Laplacian spectral clustering

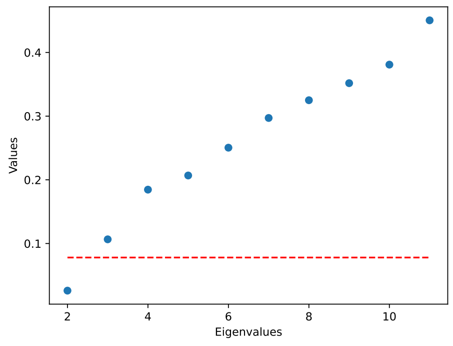
Another way to find the communities in our movie is to perform *spectral clustering.* It is a clustering method based on graph partitioning, which is exactly what we are looking for.

To have a significant result, we need to find the right number of clusters. We will use 2 different methods to accomplish this job :

1. Perform the modularity for each k (k is the number of clusters)
2. Eigengap heuristic for finding the optimal number of clusters



(a) Modularity for each number of clusters



(b) 10 first eigenvalues

(a) With the first method, it seems that 6 clusters will give us the best result, but 5 and 7 are also not far away.

(b) With the second one, it's more complicated to choose the best parameter. The line in red represents the biggest gap between the two eigenvalues. But actually, there is no well-defined gap, the differences between all eigenvalues are approximately the same. The method of eigengap heuristic is performing well when we can make a clear distinction between the eigenvalues but in our case, it is too ambiguous to make a choice.

In order to have a relevant comparison of the two methods, we choose to perform the spectral clustering with 5 clusters, as shown on Figure 4.

We can spot 5 communities :

- The community of Darth Vader is almost the same as for the Louvain Algorithm.
- The community of Luke contains people from the resistance and Treepio (C-3PO), according to the story and Treepio.
- The community of The woman who leads the resistance.
- The community of The Death Stars intercom voice, which is all the people who receive orders in the Death Stars.
- The community of the bartender and Leia. His community consists of characters related to the Mos Eisley Cantina and some people of the resistance like Leia.

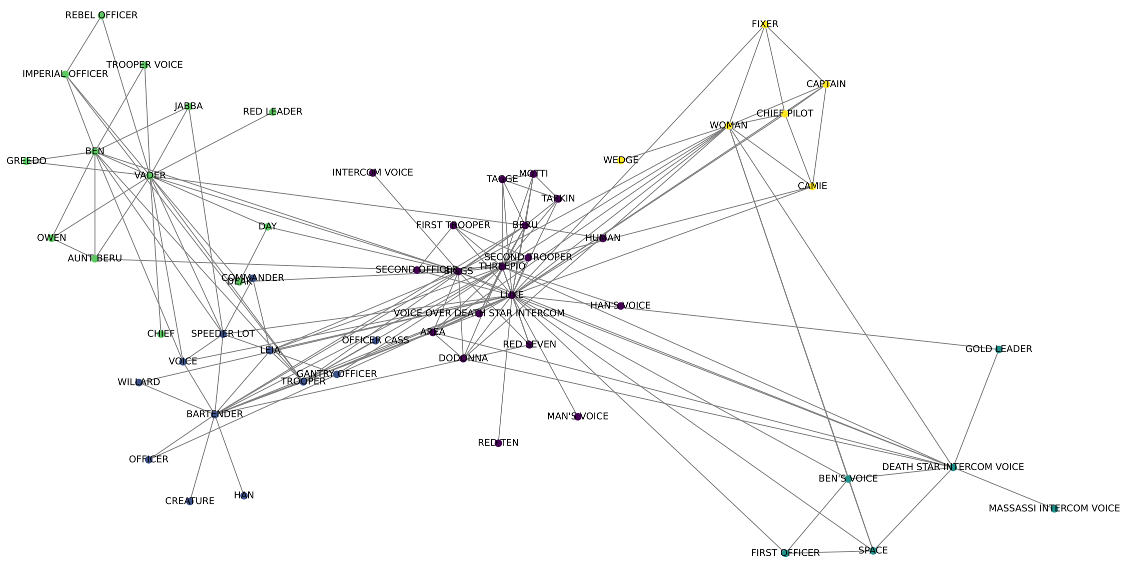


Figure 4: Spectral clustering with 5 clusters

5 Comparison of the 2 methods

5.1 Efficiency

In order to compare the efficiency of these two methods, we will choose two measures of partitions of a graph: *the modularity*[1] and *the conductance*.

We define the volume of a set of vertices S as $a(S) = \sum_{v \in S} d(v)$ and the boundary of S as $\partial(S) = \{(x, y) \in E | x \in S, y \in \bar{S}\}$, the edges from S connected to the other part of the graph.

We can now define **the conductance** as

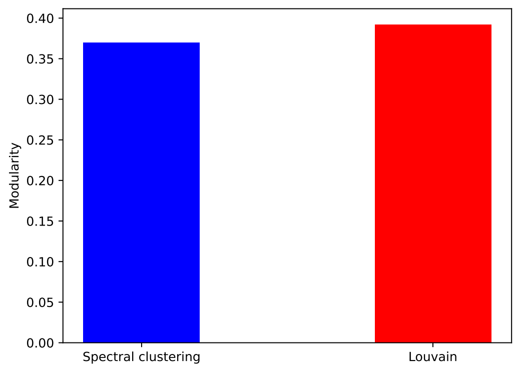
$$\Phi(S) = \frac{|\partial(S)|}{min(a(S), a(\bar{S}))}$$

This is a metric used to measure the quality of a cluster, and so a partition. Low-conductance vertex sets correspond to higher-quality cluster because the sets are larger and have fewer edges to vertices outside of the set. In order to measure the quality of the entire clustering, we decided to take the mean of all the conductance of the clusters, which seems to be a relevant choice.

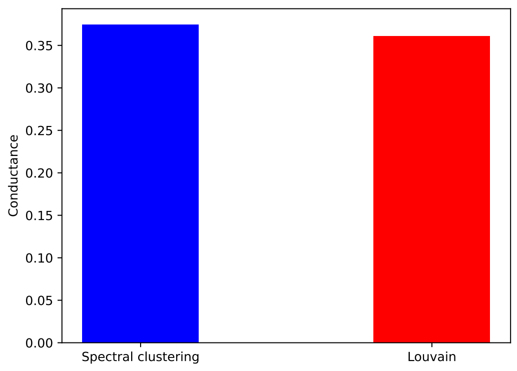
If we compare the modularity and the conductance of the two methods, we get the results in Figure 5.

With these results, we can conclude that the Louvain method is better than spectral clustering for finding communities in a network. Nevertheless, spectral clustering is not quite far away, and it is impressive knowing that it is not maximizing the modularity as the Louvain method does.

[1]already explained in section 3



(a) Modularity for both methods



(b) Conductance for both methods

Figure 5: Comparison of the methods

We are not facing this problem with our network because there are too few nodes but with large networks, spectral clustering would not be as efficient. The disadvantage of spectral clustering methods is that they require to computation and storage of the adjacency matrix of the graph. As the size of the network increases it becomes in-feasible to store the $N \times N$ adjacency matrix where N is the number of nodes. Also, these methods do not have an out-of-sample extension property. Hence, they cannot scale for very large networks (millions of nodes). Some alternatives exist anyway such as KSC-net, developed by researchers from KULeuven.[2]

5.2 Comparison of the communities

There are a lot of common communities between the 2 algorithms which prove that they are both efficient. The main difference between the 2 graphs is that the Louvain method merged two communities of the spectral algorithm (**Woman** and **Death Stars**) and created another instead, the **Biggs's** community.

6 Influence Maximisation Problem

Let us check what is going to happen if there is a rumor spreading in our network. We will make the assumption that a character can only spread the rumor to his neighbors (the neighbors of a character are all the nodes connected by an edge to the node of the character). The spreading has a probability p to happen if it fails, at each iteration, all newly infected nodes have a chance to infect each one of their neighbors. The spreading will not be able to happen even at other iterations (as if we removed the edge of the graph). This model of propagation is called the ICM (Independent Cascade Model).

Given a graph $G(V, E)$, we define the influence of a set of nodes A as $\sigma(A)$ = the expected number of active nodes at the end of the diffusion process given that A is the initial set of active nodes. We define the Influence Maximization Problem as, for a given k, finding the set of size k with maximal influence. For our problem, it is asked to take $k = 5\%$ of the total number of nodes. We decided to use $k = 2$ ($3,57\%$ of the total nodes) because with $k = 3$, the number of iterations will be very small and thus the results are hard to observe, and $k = 3$ represents $5,35\%$ of the number of nodes. Also, it reduces a lot the time of computing the Greedy algorithm, allowing us to compute a lot more

[2]see here

iterations of ICM (see paragraph below).

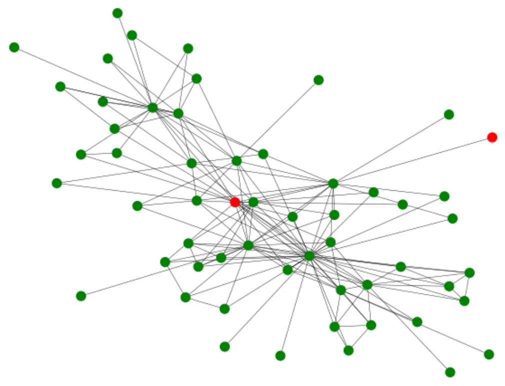
The Greedy-algorithm that we want to solve the IMP is the following :

Algorithm 1: Greedy hill-climbing heuristic for influence maximization

```
Input  : Graph $G(V, E)$ and an ICM model on $G$
Output: Optimal set $A_0$ for influence maximization
Generate a large number of realizations $X_i$ of the ICM on $G$;
Initialize $A_0 = []$;
while $length(A_0) < k$ do
    Find $v$ in $V \backslash A_0$ that maximizes $\sum P(X_i)\sigma_{X_i}(A_0 \cup \{v\})$;
    Add $v$ to $A_0$;
end
```

For the explanation of our ICM (Independent Cascade Model) implementation, see section 7.

We know that our graph is connected so experimenting with the algorithm with $p = 1$ would not make any sense because it would always infect all of the nodes, we chose $p = 0.5$ for our experiments. When analyzing the result of this method with $k = 2$ and $p = 0.5$, although the method did not always have the same result, we saw patterns in the results. (see Figure 6)
Indeed, the MI (maximal Influence) set seems to always contain a central node with a high degree and another one of degree 1 in the extremity of the graph. As our graph is very heterogeneous, the degree of the nodes varies a lot. It seems logical to have a node of big influence (high node degree) in the optimal solution because it will create an outburst in the epidemic, also if we only had 2 nodes of degree 1, there is already a probability of $\frac{1}{4}$ that the infection stops immediately at the first iteration. The second node seems weird at first, why is it a node of degree = 1 ? But when we plot the graph of the infected nodes after a full propagation with $p = 0.5$ (see Figure 9, we see that almost every time the only nodes still not infected are extremity nodes, with degree = 1, so having one of these nodes already infected will ensure one of them to be infected. In almost every case, we will not need more than 1 central node to infect all of the nodes with degree ≥ 2.


(a) Example 1 of greedy result

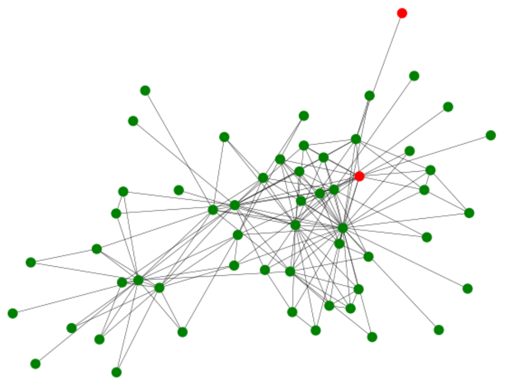
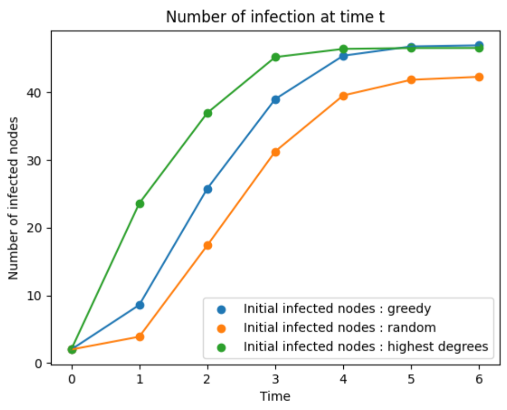

(b) Example 2 of greedy result

Figure 6: Greedy results

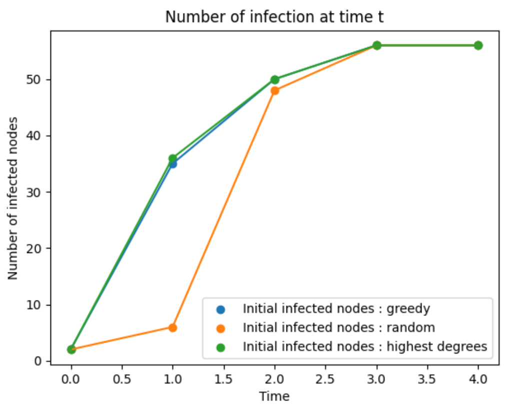
7 Independent Cascade Model (ICM)

7.1 Comparison of initial sets of nodes

We would like to compare our MI set obtained earlier (that we will call "greedy" in the plots) with other sets of nodes such as a random set or a set with the 2 nodes of higher degrees, which we could think would be the best at first sight. So we ran our Python implementation of ICM with those 3 possibilities and we got the average results shown in Figure 7. This is an average of over 300 propagations (we had nearly exactly the same results with more propagations so we kept 300).



(a) ICM with $p = 0.5$



(b) ICM with $p = 1$

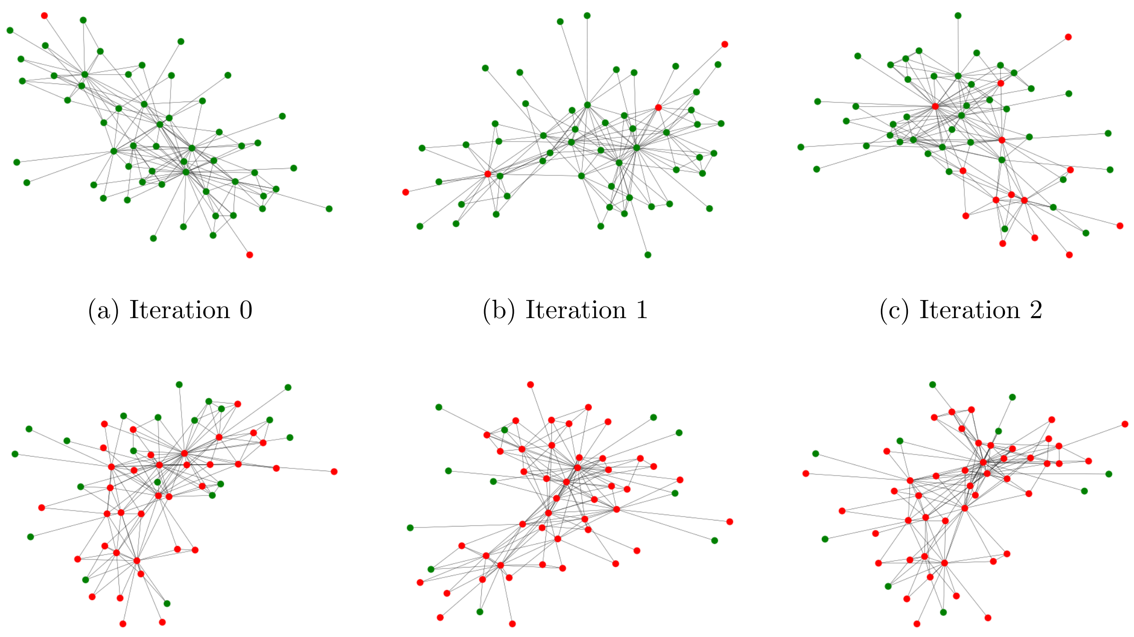
Figure 7: Comparison different initial nodes

We can observe in the first plot that the IM set of nodes obtained with a greedy algorithm has a mean final value of the infected node just a little bit above the "highest degrees" initialization. This is due to what we said earlier about the fact that there is usually one node of degree 1 in the IM set but the 2 nodes of the "highest degrees" are central nodes of the graph. This little optimization makes the difference for the greedy initialization to reach a bit more nodes on average. The random initialization is a bit under the two others, probably because of bad initialization (two nodes of degree 1 for example, which can lead with a probability of 25% to only 2 infected nodes).

In the second plot, we fixed $p = 1$ to have an idea of the spreading speed of the rumor when we know that the spread will not fail. As expected, the start is at the advantage of the "highest degrees" set but it is quickly overtaken by the IM set, the random set takes a bit more time but nearly the same number of iterations to reach the maximum of nodes

7.2 Propagation

Here you can see one example of propagation of the ICM method with $p = 0.5$. We can see that some nodes with degree $= 1$ are not infected at the end, meaning that their neighbors failed to infect them.


(a) Iteration 0 (b) Iteration 1 (c) Iteration 2

(a) Iteration 3 (b) Iteration 4 (c) Iteration 5

Figure 9: Example of propagation with ICM model and $p = 0.5$

8 Barabasi-Albert network

8.1 Methodology

We aimed to construct the BA network with an average degree and size similar to our original character network. In the BA model, early nodes tend to get more links because of preferential attachment, leading to the creation of hub nodes with many connections.

- **Compute the Average Degree**: The average degree k of the original network is given by:

$$k = \frac{\sum_{i=1}^{N} k_i}{N}$$

  where k_i represents the degree of the i^{th} node and N is the total number of nodes.

- **Compute number of links m**:

$$m = \frac{k}{2}$$

8.2 Analyze the results

Degree Assortativity :

- Original graph: The degree assortativity is -0.351
- Barabasi-Albert graph: The degree assortativity is -0.202

Nodes of Maximal Influence (with a greedy algorithm):

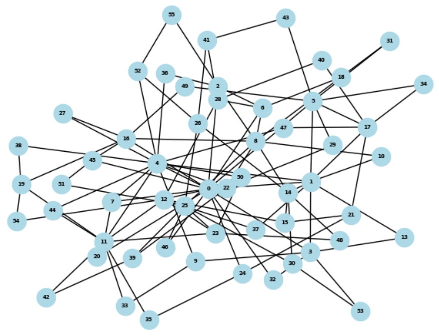


Figure 10: Generated Barabasi-Albert network

- In the Original Network: Nodes 0, 19 are identified as having maximal influence.
- In the Barabasi-Albert (BA) Network: Nodes 20, 47 are identified as having maximal influence.

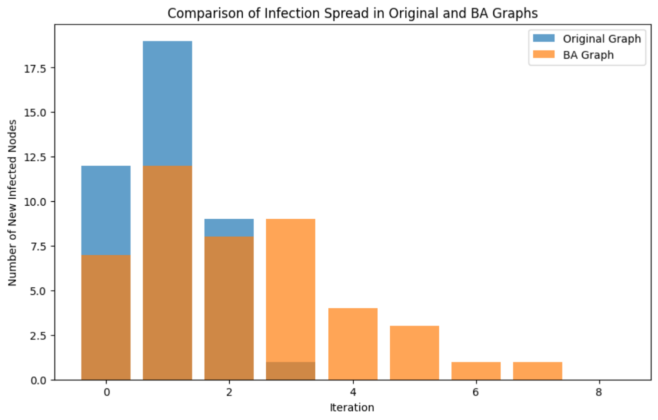


Figure 11: Comparison of Infection Spread in Original and BA Graphs starting with a greedy optimal initialization

We see in Figure 11 that starting with a greedy optimal initialization the original graph infects more people in the first iterations than the BA graph. This is due to the fact that the BA graph is more homogeneous. The nodes tend to have a degree closer to the average degree of the graph. We can see it at the assortativity, which is negative in both graphs, but more pronounced in the original graph. This means that the difference between degrees are stronger in the original graph.

When an infection or information starts in the original graph, spreading in such a heterogeneous network, it initially spreads very rapidly if one of these hubs is infected. However, once these major hubs are infected, the spread slows down because the remaining nodes are not as well connected. This is why we might observe a quick spike in infections in the original graph initially but a slower rate subsequently.

9 Annexe

9.1 Explanation of the spectral clustering

The goal of this clustering is to find a partition of the graph such that the edges between different groups have a very low weight (which means that points in different clusters are dissimilar from each other) and the edges within a group have high weight (which means that points within the same cluster are similar to each other). To solve this problem, we use the Laplacian matrix of the graph:

$$L = D - W$$

which has the important following property:
for all vectors $f_i \in \Re^n$ *:*

$$f'Lf = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$$

Given a graph with adjacency matrix W, the simplest and most direct way to construct a partition of the graph is to solve the min-cut problem. We define $W(A,B) := \sum_{i\in A, j\in B} w_{ij}$ and $\bar{A}$ for the complement of A. For a given number k of subsets, the min-cut approach consists in choosing a partition $A_1, ..., A_k$ which minimizes :

$$cut(A_1, ...,_A k) := \frac{1}{2}\sum_{i=1}^{k} W(A_i, \bar{A}_i)$$

Mostly, the solution of min-cut separates one individual node from the rest of the graph. To avoid that, we need to introduce a condition on the subsets $A_1, ..., A_k$: they have to be sufficiently large. This is why we modify the problem by minimizing the *RatioCut* :

$$RatioCut(A_1, ..., A_k) := \frac{1}{2}\sum_{i=1}^{k} \frac{W(A_i, \bar{A}_i}{|A_i|} = \sum_{i=1}^{k} \frac{cut(A_i, \bar{A}_i)}{|A_i|} = \sum_{i=1}^{k} h_i' L h_i = \sum_{i=1}^{k} (H'LH)_{ii} = Tr(H'LH)$$

this leads to the optimization problem :

$$\underset{H\in\Re^{n\times k}}{min} Tr(H^{-1}LH) \text{ subject to } H^{-1}H = I$$

By the Rayleigh-Ritz theorem, the solution is given by choosing H as the matrix which contains the first k eigenvectors of L as columns.[3]

[3]see here