Graph Project: Communities detection

ABDELLATIF MAMOUN
WALID MALEK

December 4, 2021

Contents

1 Introduction **2**

2 Girvan Newman algorithm **2**
2.1 Question 2.1 2
2.2 Question 2.2 3
2.3 Question 2.3 4

3 Neighborhood Overlap (NOVER) algorithm **5**
3.1 Question 3.1 6
3.2 Question 3.2 3.3 7
3.3 Question 3.4 8

4 Graph convolution **9**
4.1 Question 4.1 9
4.2 Question 4.2 10

5 Conclusion **10**

1 Introduction

In this project, we've had the opportunity to work on the famous Zachary's karate club problem, the work was done in three steps, in the first step, we've classified each individual using the Girvan Newman algorithm, however since this method always presume a previous knowledge on the final number of communities, we used in the second part the neighbor overlap algorithm. Finally we tackled a bigger problem which is the process of gigantic graphs from the social networks combine with complex information gathered on each individuals, in this part we got to explore what is the kernel operator and how to calculate it using a functional programming.

2 Girvan Newman algorithm

The Girvan–Newman algorithm detects communities by progressively removing edges from the original network. The connected components of the remaining network are the communities. Instead of trying to construct a measure that tells us which edges are the most central to communities, the Girvan–Newman algorithm focuses on edges that are most likely "between" communities. hence we define a major tool called the betweeness of an edge, it is the number of shortest path between all the vertices of the graph which pass through it. in the betweeness_score we've started to update all the scores of the edges to 0 in order to avoid an accumulation of all the scores in the Girvan Newman algorithm.

2.1 Question 2.1

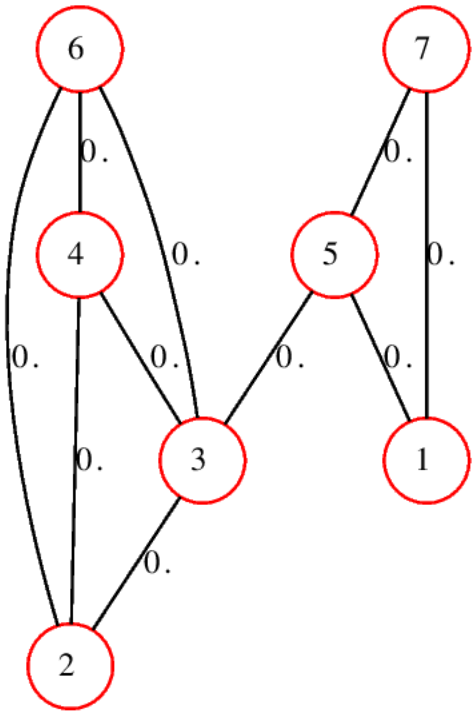

FIGURE 1 – the graphe g1.

In the Girvan Newman algorithm we calculate the betweeness of all edges and then remove the edges with the smallest betweeness, since we would like to separate the graph g1 into 2 communities, it only takes one iteration to get the job done.

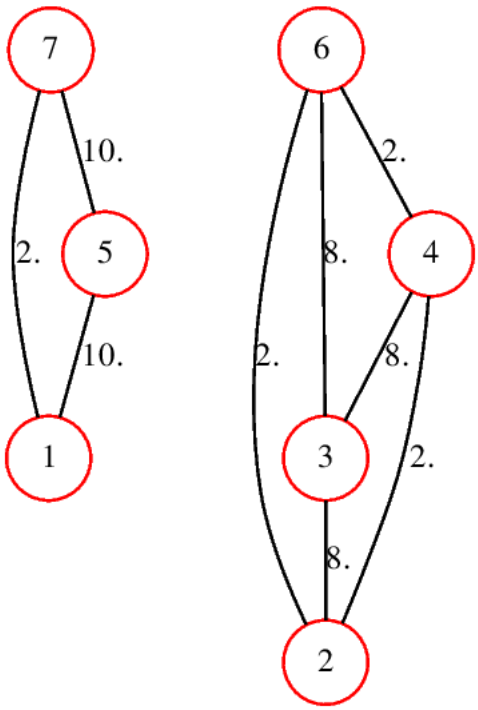

FIGURE 2 – the graph g1 after the Girvan Newman algorithm.

2.2 Question 2.2

The graph below represents the graph problem we're trying to classify into two communities :

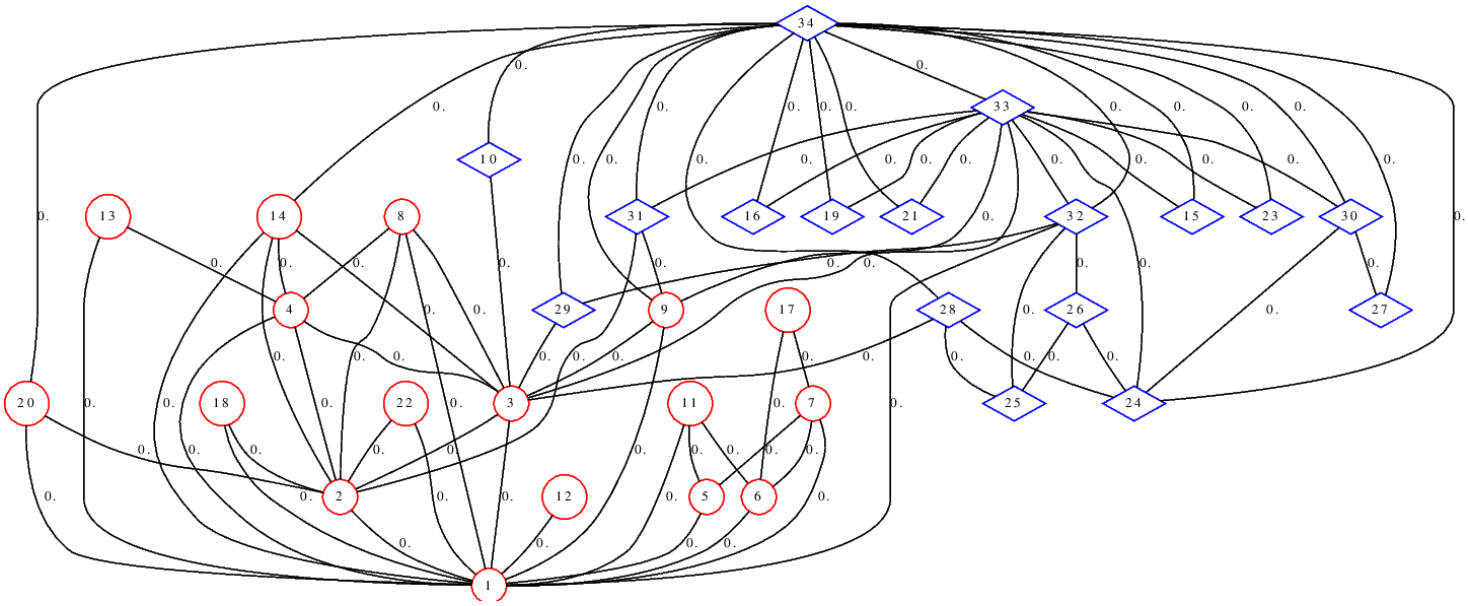

FIGURE 3 – karat7 graph.

After applying the Girvan Newman Algorithm we get :

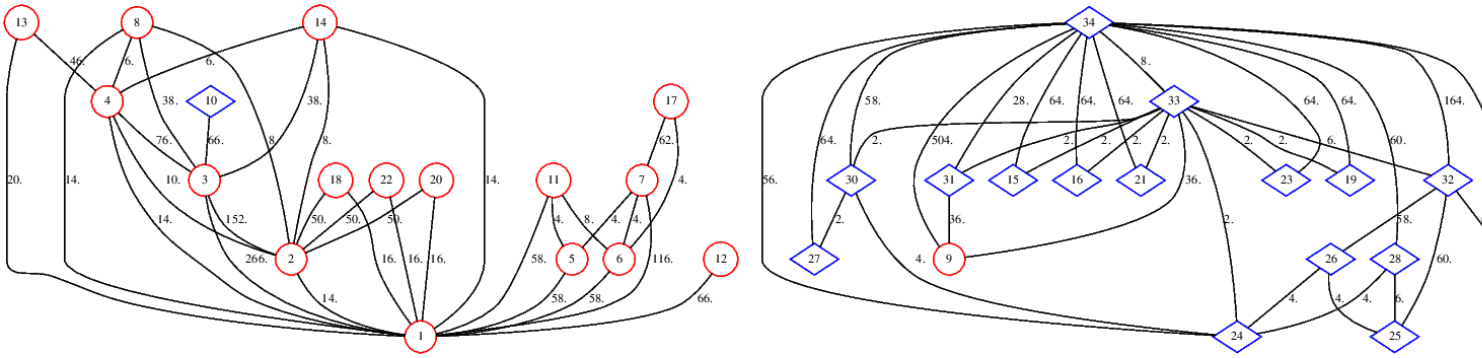

FIGURE 4 – karat7 graph after the GN algorithm.

the complexity of the GN algorithm is in the worst cases cubic O(n^3), hence using the algorithm for large networks such as the twitter's isn't the best thing to do, also we can see that a blue label is inside the red community so the predictions aren't precise however the algorithm remains efficient for small groups in regard of optimization.

2.3 Question 2.3

the result of the Girvan Newman algorithm on the graph g2 is :

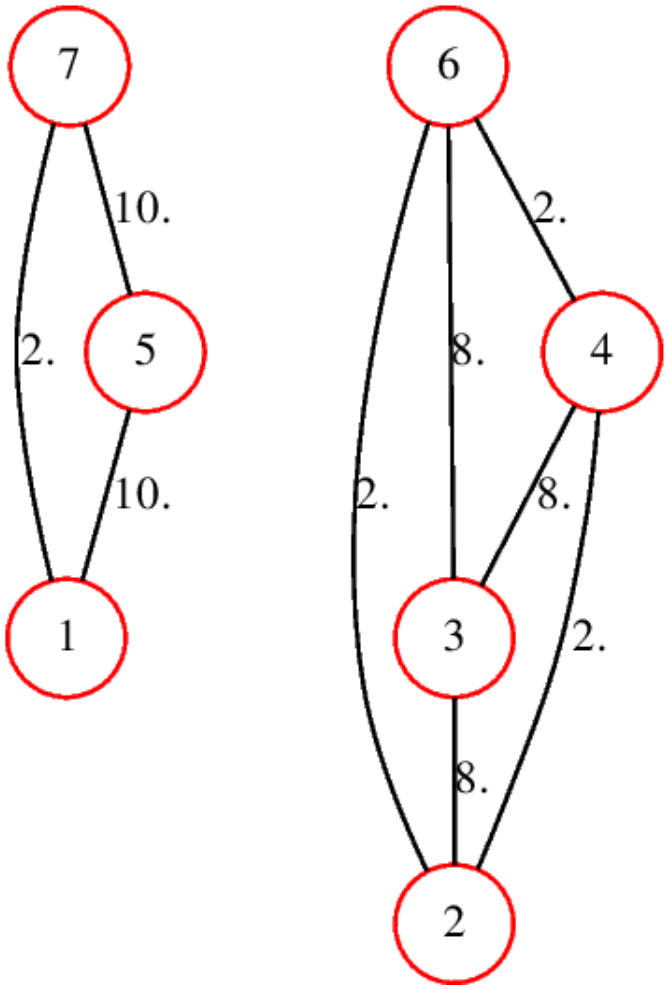

FIGURE 5 – Graph g2 after the GN algorithm.

By Observing the graph, we can expect that if we desire to separate a graph into a large number of communities, one may think of Removing **ALL** the edges with the highest betweenness.

3 Neighborhood Overlap (NOVER) algorithm

the inconvenience of the previous algorithm is its previous knowledge of the number of communities, however often we find ourselves in a clustering problems, therefore the GN algorithm doesn't work, as an alternative we discuss the Neighborhood Overlap algorithm, As for the betweenness in the Girvan Newman algorithm [1] we will de-fine a score on each edges : the NOVER score only this time we remove all the edges with the smallest nover score, this value is calculated using the following formula :

$$Nover_{score}(u,v) = \frac{c_{u,v}}{d_u + d_v - c_{u,v}}$$

with $c_{u,v}$ the number of common neighbors of both u and v, and d_u,d_u the degrees of u and v respectively,without counting u or v in each one of them. However, here we don't know the number

of community, so we don't know when to stop. Then, here we introduce the modularity paramater :

$$Mod = \sum_{communities} \sum_{(i,j) \in Vertices} A_{i,j} - \frac{d_i * d_j}{2m}$$

where A is the adjacency matrix, d_i is the is the degree v_i and m the initial number of edge in the graph. In fact, this parameter shows us the strength of a division into communities of a graph. So, the optimal number of communities, corresponds to the highest modularity. In the next sections, we will answer to the questions presented in the subject :

3.1 Question 3.1

Here are the nover scores over multiple iterations on the g3 graph : We mark here the nover scores

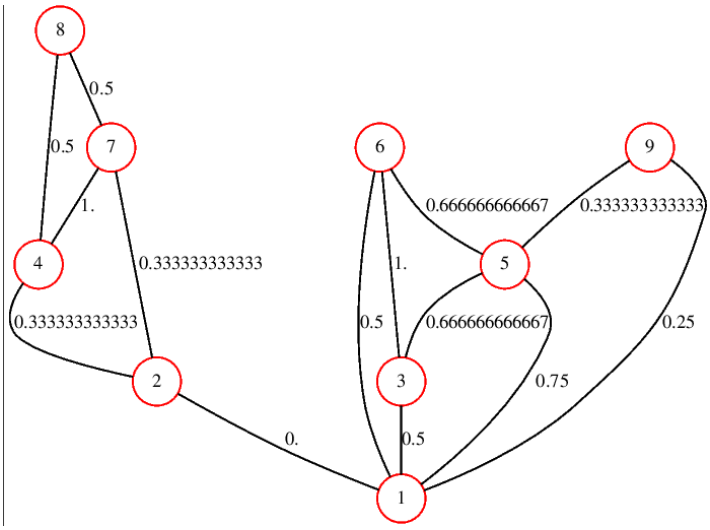

FIGURE 6 – Graph g3 after 0 iteration.

of the algorithm on the graph `g3`, we observe that the edge *(1,2)* has the smallest nover score. Then, we proceed to the next iterations. We will get the following results in the next iterations.

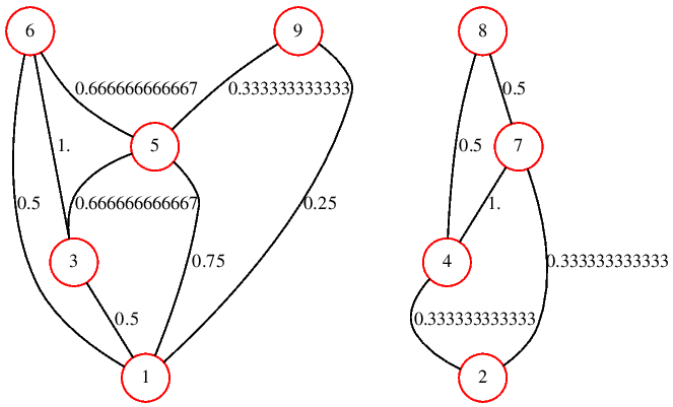

FIGURE 7 – Graph g3 after 1 iteration.

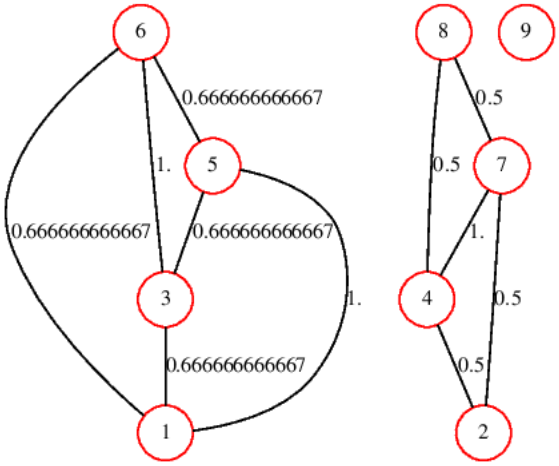

FIGURE 8 – Graph g3 after 2 iteration.

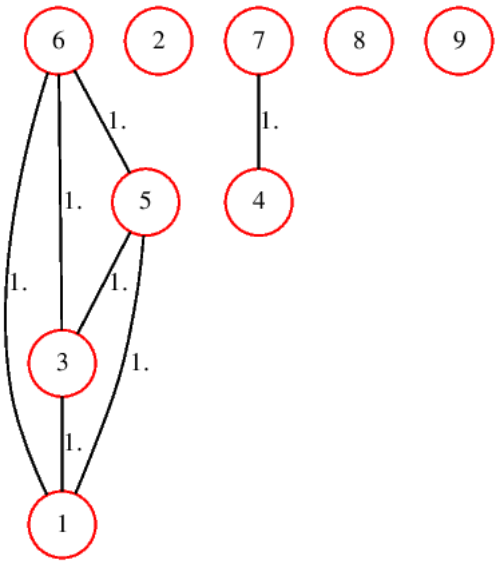

FIGURE 9 – Graph g3 after 3 iteration.

3.2 Question 3.2 3.3

We will apply the *Nover* algorithm on the graph `karate`. We will at the first time, calculate the nover score of the graph, then apply the *step nover* algorithm, until we will get the highest modularity. We will then get the optimal community partition. We will get in iteration 1 the following graph : Then, in the iteration 2, we will get the following outline : In the iteration 3, we will get the following schema :
However, we can't assume that we will get a satisfying separation of communities within few iterations. Therefore, we will compute the modularity parameter in each iteration. Then, for 3 iterations. We will get the highest modularity parameter during the iteration 2, even if we will pass through some extra iterations. Therefore, we conclude that best separation, with the *Nover algorithm* is : two separations.
P.S : in the Girvan Newman Algorithm we remove edges one by one, hence we are sure to get at most one more community in the next iteration, however the NOVER algorithm remove all edges with the lowest nover score so in order to separate a community into multiple communities, the Nover algorithm seems to be faster as regard of time and complexity.

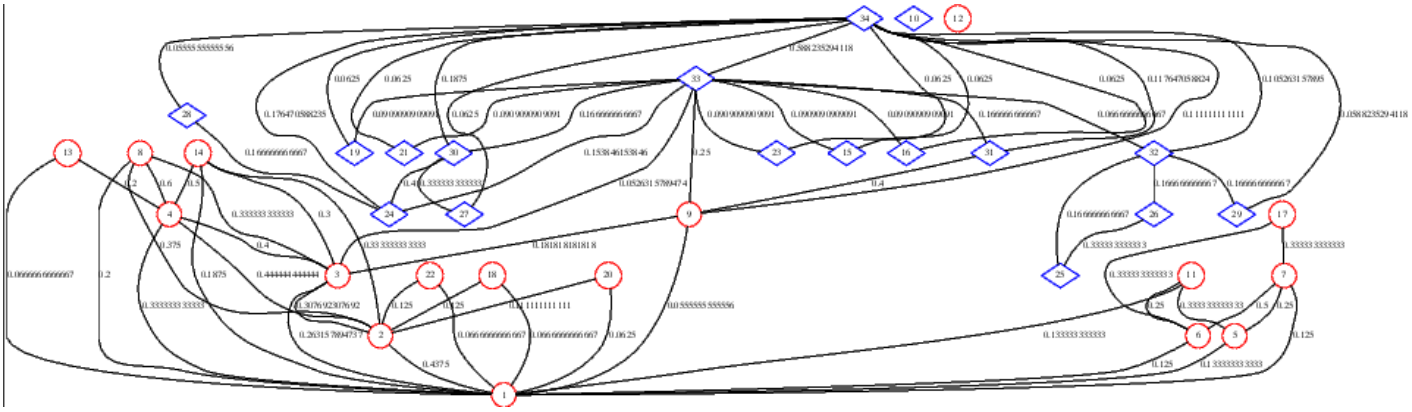

FIGURE 10 – Graph karate in iteration 1.

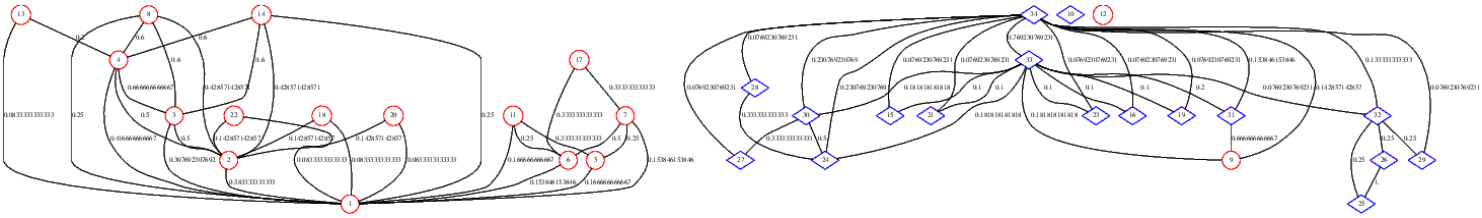

FIGURE 11 – Graph karate in iteration 2.

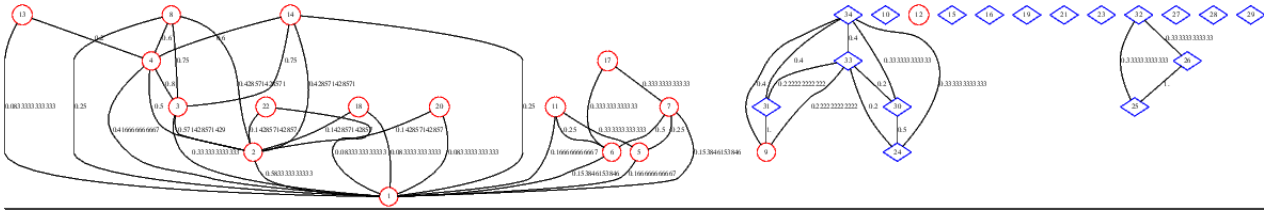

FIGURE 12 – Graph karate in iteration 3.

```
Le nombre de communautés dans l'itération 0 est : 1
Le facteur modularité dans l'itération 0 est : 3.884615
Le nombre de communautés dans l'itération 1 est : 3
Le facteur modularité dans l'itération 1 est : 3.425373
Le nombre de communautés dans l'itération 2 est : 4
Le facteur modularité dans l'itération 2 est : 35.234375
Le nombre de communautés dans l'itération 3 est : 13
Le facteur modularité dans l'itération 3 est : 23.597826
```

FIGURE 13 – Modularity parameter for 3 iterations.

3.3 Question 3.4

- The vertex number 10 is one edge away from the blue community and one edge away from the red community, so being discarded at the first iteration, may imply he's the person who's likely to stop the karate after the separation of the club.
-Our fist observation is that the vertex number 12 got discarded even if he was closely related to the vertex number one, hence we can suppose that the Nover algorithm isn't fair to vertexes with small degrees.

4 Graph convolution

In order to process the gigantic graphs from the social networks, the previous algorithms have their own drawbacks, thus we focus our interest on the Kernel operator defined as :

$$\forall v \in g, c(v) = \frac{1}{|\mathcal{N}(v)|+1} * \left(\sum_{u \in \mathcal{N}(v)} h_u + h_v \right)$$

where h_v is the parameters of the vertex v (basically a vector of $\Re^p$),$\mathcal{N}(v)$ is the neighbours of v.

4.1 Question 4.1

le convolution sur le graphe g4 est défini comme ceci :

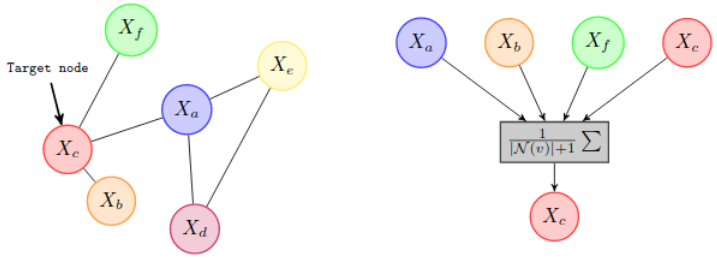

Figure 3: Illustration of a graph convolution on a vertex

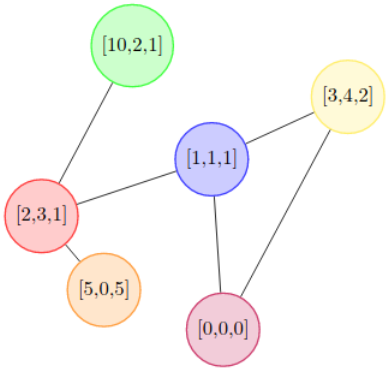

Figure 4: Graph carrying parameters

FIGURE 14 – Illustration of the convolution process.

For the vertex 1 (X_c in the previous graph) $\mathcal{N}(v_1)$=3 and
$c(v_1) = \frac{1}{4}([2 \quad 3 \quad 1] + [10 \quad 2 \quad 1] + [1 \quad 1 \quad 1] + [5 \quad 0 \quad 5]) = \frac{1}{4}([18 \quad 6 \quad 8]) = [4.5 \quad 1.5 \quad 2]$
For the vertex 2 (X_b in the previous graph) $\mathcal{N}(v_2)$=1 and
$c(v_2) = \frac{1}{2}([2 \quad 3 \quad 1] + [5 \quad 0 \quad 5] = \frac{1}{2}([7 \quad 3 \quad 6]) = [3.5 \quad 1.5 \quad 3]$
and so on :
$c(v_3) = [6 \quad 2.5 \quad 1]$
$c(v_4) = [1.5 \quad 2 \quad 1]$
$c(v_5) = [1/3 \quad 5/3 \quad 1]$
$c(v_6) = [1/3 \quad 5/3 \quad 1]$
Thankfully these are the same results we've obtained using the algorithm.

```
************************************************************
************************************************************
************************************************************
1 : [ 2. 3. 1. ]
2 : [ 5. 0. 5. ]
3 : [ 10. 2. 1. ]
4 : [ 1. 1. 1. ]
5 : [ 3. 4. 2. ]
6 : [ 0. 0. 0. ]
aprés convolution
convolution sur 1 : [ 4.5 1.5 2. ]
convolution sur 2 : [ 3.5 1.5 3. ]
convolution sur 3 : [ 6. 2.5 1. ]
convolution sur 4 : [ 1.5 2. 1. ]
convolution sur 5 : [ 1.33333333333 1.66666666667 1. ]
convolution sur 6 : [ 1.33333333333 1.66666666667 1. ]
```

FIGURE 15 – Convolution results on the graph g4.

4.2 Question 4.2

Using the Girvan Newamn or Nover algorithm is not advisable because it it time-demanding. As we've seen before the complexity can get at the worst cases to a cubic complexity So, for a data of millions of users the algorithms fails miserably on regard of optimization and complexity.

5 Conclusion

In this project we've had the opportunity to explore three ways of exploring data, the first one is the Girvan Newman Algorithm where we separate a graph knowing the number of communities wanted . Then we explored the the Nover algorithm, a clusering algorithm where we separate a certain graph to an optimal separation, and finally the convolution graph we explore the basics of the convolution process using the famous kernel operator used in neural networks.