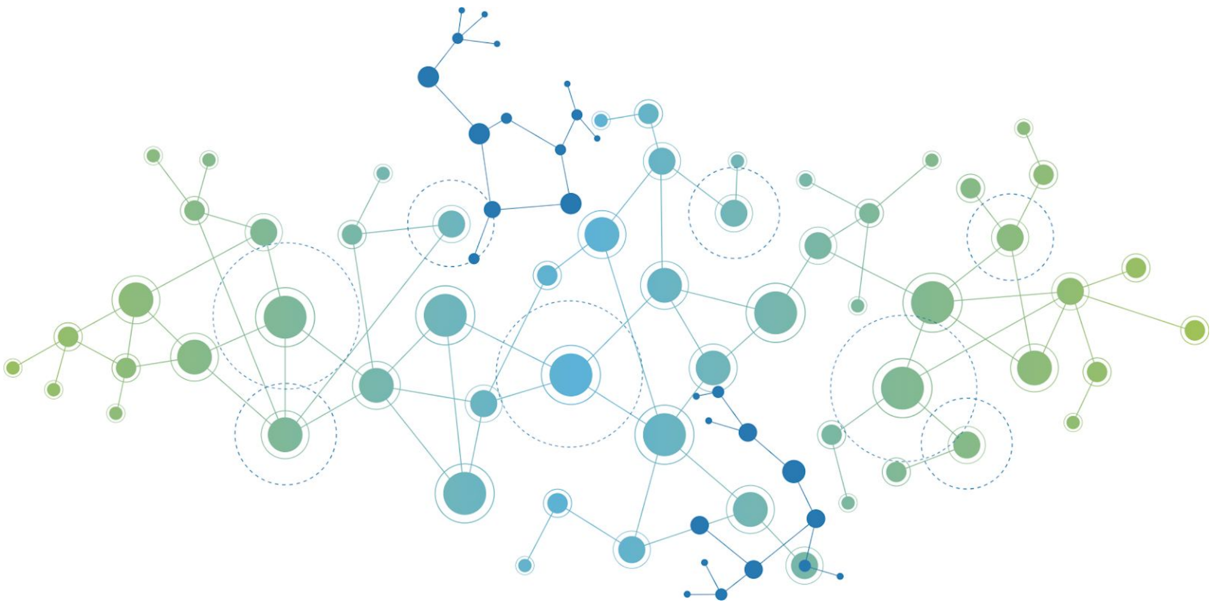
U.PORTO

FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO



Homework #2

Network Science

Henrique Branquinho **201804341**

April, 2020

Link Analysis and PageRank

1. **Draw a network in which one node has a very high value of PageRank, although the same node has low closeness and betweenness centrality.**

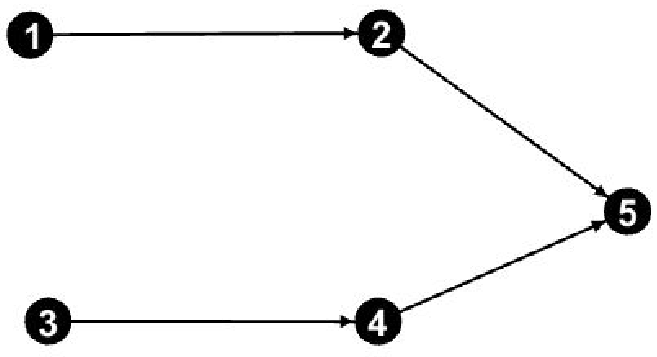
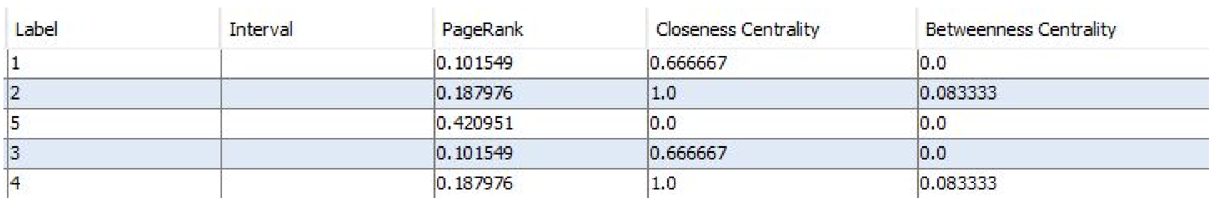


Figure 1 - Example network for exercise 1

PageRank parameters in Gephi:

β = 0.85

Epsilon = 0.001



Label	Interval	PageRank	Closeness Centrality	Betweenness Centrality
1		0.101549	0.666667	0.0
2		0.187976	1.0	0.083333
5		0.420951	0.0	0.0
3		0.101549	0.666667	0.0
4		0.187976	1.0	0.083333

Table 1 - PageRank and centrality measures for graph in Figure 1 (Beta = 0.85, Epsilon = 0.001)

Node 5 has the highest PageRank value, but has the lowest values for centrality measures (0 for both Closeness Centrality and Betweenness Centrality).

2. The damping factor in PageRank controls how of often we follow one of the links of the current node vs going to an arbitrary node on the network.

a. What does β = 0 mean? What would happen to the PageRank values in that case? Why?

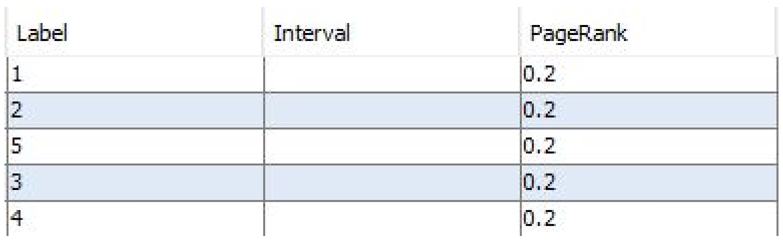
The PageRank values would be equal to 1/N for every node. This means that the surfer would just randomly jump from one page to another, without considering the way pages are connected. If we look at the formula for calculating the PageRank value **r_j** for a node j at iteration **t**:

$$r_j = \sum_{i \to j} \beta \frac{r_i}{d_i} + (1-\beta)\frac{1}{n}$$

d_i ... out-degree of node i

Formula 1 - Formula for PageRank value for a node j

With β=0, **r_j** = 1/N. The algorithm would converge on the first iteration.
If we use the network from exercise 1 as an example:



Label	Interval	PageRank
1		0.2
2		0.2
5		0.2
3		0.2
4		0.2

Table 2 - PageRank values for graph in Figure 1 (Beta = 0)

b. What does β = 1 mean? Can you explain a possible problem with using that value?

With Beta = 1, the surfer never randomly jumps between pages that aren't connected, he only follows page links. This causes a problem known as "spider traps", where groups of pages have no out-links out of those groups and end up accumulating all the importance. If the user only follows page links, he will stay stuck in that group, making those pages seemingly important.

3. **Implement a program (in any programming language) for manually computing the (normalized) PageRank values of a small network using power iterations (the ”flow” mode). Attach the program to your homework submission with a very short description on how it works.**

The file is **pageRank.py**. To run it, use:

python3 pageRank.py filename beta epsilon CSV

Where filename is the name of the file where the graph is, in edge list format (similar to last homework); beta is the value for beta; CSV should be “Y” if the number of iterations is too big, since the program will output the ranks per iteration in CSV fashion. To store in CSV, run as

pageRank.py filename beta epsilon CSV > outfile

There is also a file called **plotter.py**. To run it use:

python3 plotter.py filename minbeta maxbeta step

This will run pageRank for the graph in filename with beta values in [minbeta,maxbeta] with increments of “step”. It will plot the graphics required in exercise 5 in “filename_plot.png”. Epsilon will be set to 0.00001 by default.

4. **Use your program to compute the PageRank values of the following network (with $\beta = 0.85$). Show the values of all nodes for each iteration until the computation converges.**

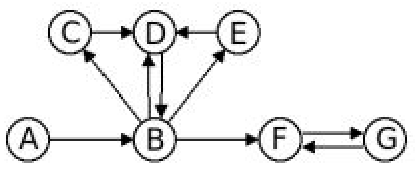
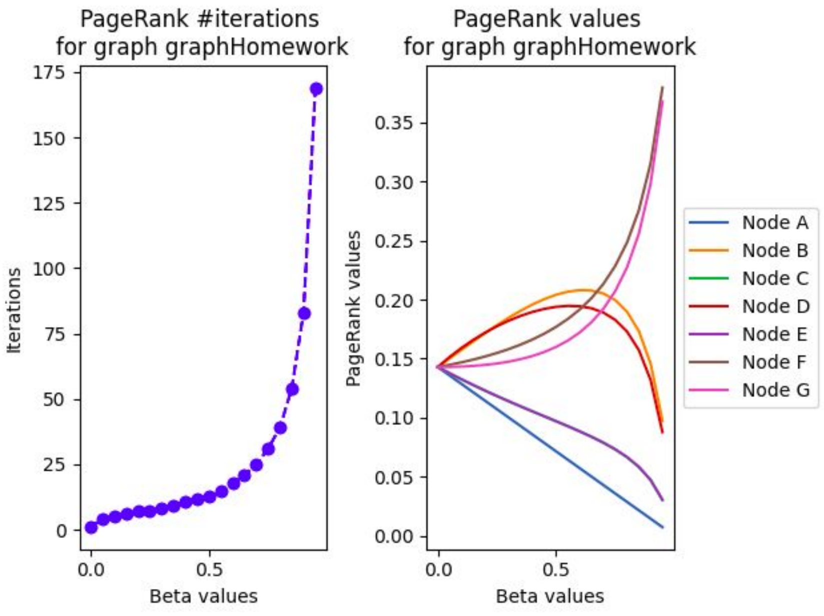


Figure 2 - Example network for exercise 4

The answer for this question is in the file “outHomework.csv” in the PAGE_RANK folder. (Epsilon was 0.00001)

5. **Use the program to do computations varying the β parameter from 0.0 to 1.0 in steps of 0.05 and:**

a. **Show in a plot the number of iterations needed until convergence is reached as you change β. Can you explain what is happening?**
b. **Show in a plot the different PageRank values of all nodes as your change β. Can you divide the nodes into different curve behaviors? Can you explain what is happening?**



Plot 1 - Evolution of number of iterations and PageRank values with Beta values

The graph used has a "spider trap" in nodes F and G. With low values of Beta, the algorithm converges rapidly because there is not much weight on the importance of pages connected by in-links when calculating the PageRank value for a node. When Beta starts to get close to 1, there is a lot of weight on the importance of these pages, so the convergence is more susceptible to small changes of values between iterations, which translates into a higher number of iterations.

Regarding the PageRank values for nodes, there are 3 explicit curve behaviours in my opinion:

- Nodes F and G - This is the "spider trap". As we can see, the higher the Beta value, the less likely we are to get out of this spider trap, so the PageRank values of these nodes grow in an apparent exponential way.
- Nodes E and C - Nodes E and C overlap in the graph as they have the same PageRank values. These nodes act more like gateways between highly connected nodes, having just one in-link (B, who has out-degree of 3), so they

are bound to have small PageRank values, that decrease with the increase of Beta because we diminish the chances of random teleportation of the surfer to these pages.

- Node A - Has no in-links, can only be reached by random teleportation so it's value will always be proportional to Beta.
- Nodes B and D - They are relatively important nodes with PageRank values increasing with Beta until the spider-trap starts to become more noticeable.

Community Discovery

6.

a. Run Louvain Algorithm to find the best possible communities and create a table showing: name of the movie, number of nodes and edges, number of communities found and modularity for those communities. Give a brief comment on which networks seem to present community structure, and why.

Movie	Nº nodes / Nº edges	Nº communities	Modularity
Blade Runner	25 / 44	3	0.466
The Godfather II	78 / 219	6	0.372
LOTR: Return of the King	35 / 98	4	0.472
Pulp Fiction	38 / 102	6	0.328
Star Wars V	37 / 82	5	0.356

Table 3 - Movies and their communities and modularity

Theoretically, all of these networks seem to present community structure, since their modularity values are between 0.3 and 0.7. However, I would say that LOTR and Blade Runner are the ones that seem to preset a stronger community structure due to their higher modularity values.

b. Choose any two of the movies and produce visualizations for the networks, labeling the nodes with their character names, using colors to represent communities and the size of the nodes to represent PageRank values. Try to make the picture as aesthetically pleasing as possible, reinforcing the community structure (and explain how you created the layout). Give a brief informal description on the meaning of the communities in the context of the movie (are they what you were expecting? are they meaningful?)

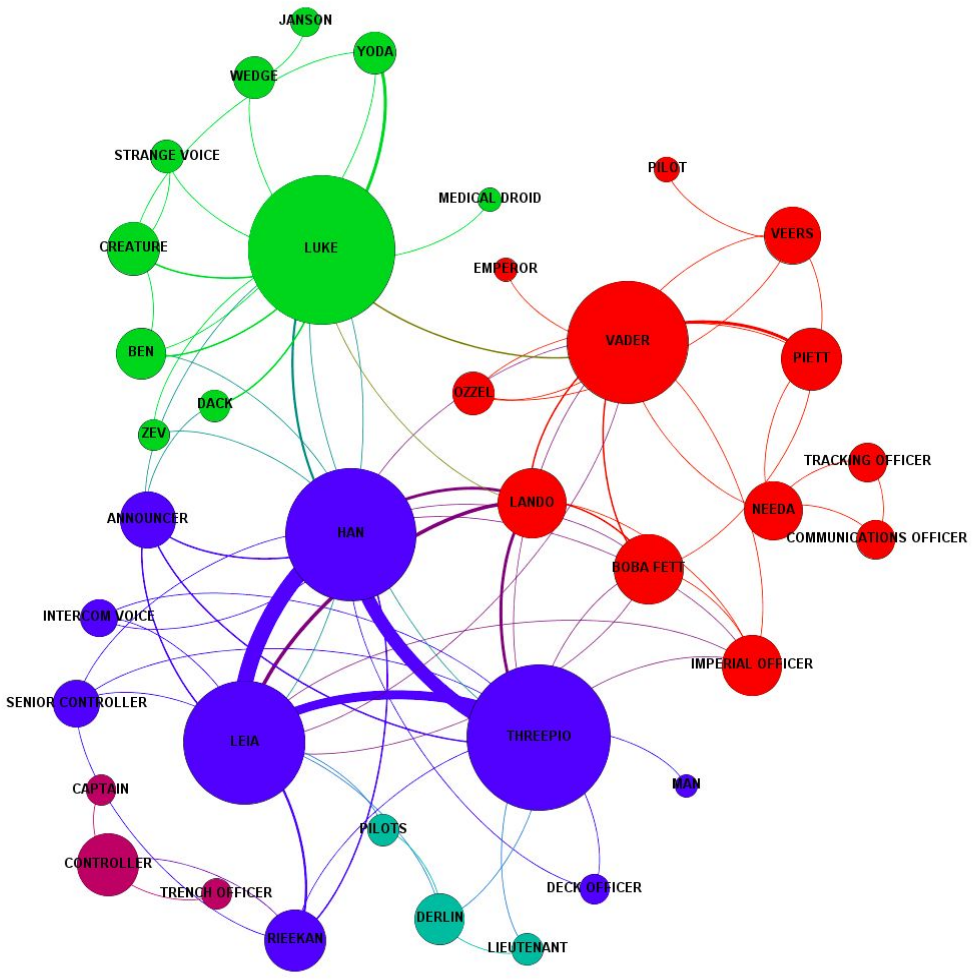


Figure 3 - Star Wars communities

The communities found were significant. In blue, we have the Rebel Alliance, mainly Han, Leia and 3PO who stay together for the entire movie (no Chewbacca in the graph though). The other characters are also members of the Rebel Alliance who make an appearance while they are settled on the ice planet Hoth. There are other two minor communities of soldiers who appear together on some scenes, even though they are part of the Alliance.

In red, we have the Empire community, where Vader is the most representative character, along with other Empire officials. Lando also appears in this community even though he shared scenes with both the Alliance and the Empire members.

In green, we have Luke's adventure in the movie. There are soldiers of the Alliance who fought with him in Hoth, and also Yoda and Ben who were with him while training to becom a Jedi.

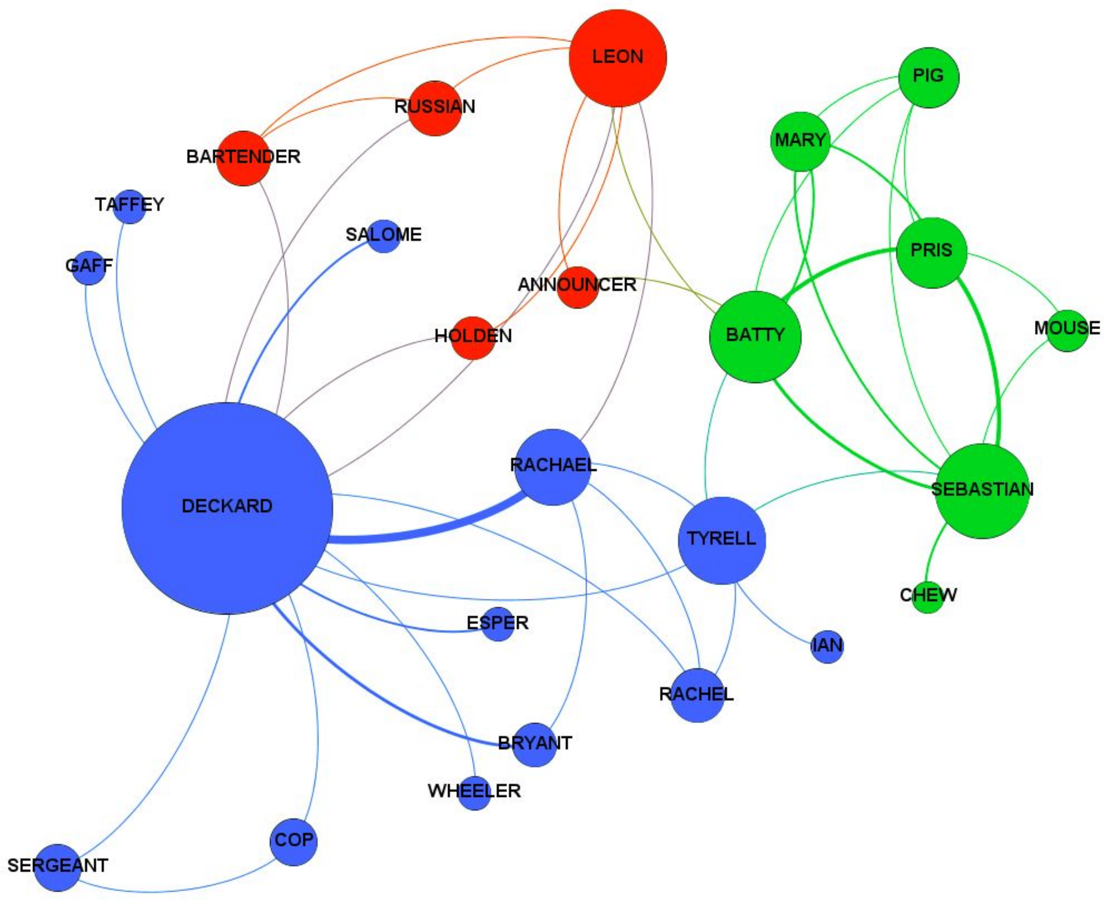


Figure 4 - Blade Runner communities

The blue community is constituted mainly by Deckard and the characters that interact with him over the movie. Most characters in this community only talk to Deckard once in the movie, throughout his investigation, so it's normal they are all in one community. There is a strong connection between Deckard and Rachel as they have a lot of scenes together.

The orange community is constituted mainly by the Replicants, and other characters that only interact with the Replicants like J.F. Sebastian and Chew, who help them get access to Tyrell. Tyrell is one of the main links between the Deckard community and the Replicant community, since he talks to Deckard about the Replicants and the Replicants seek him to extend their lives.

There is a red community which is mainly constituted by Leon I would argue, who also is a replicant but spends most of his time in the movie separated from the other replicants, following Deckard around and trying to kill him. Some other characters in this community are smaller characters who often share scenes with characters from other groups, mostly due to Deckard's investigations.

I found it odd that there is no link between Deckard and Batty or Pris, because Deckard fights them in one of the final scenes of the movie.

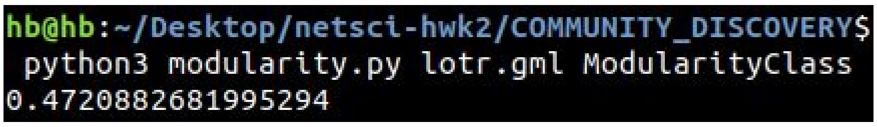
Overall, I found these communities to be meaningful and good representatives of the movie.

c. Implement a program (in any programming language) for manually computing the (normalized) modularity of a network when given a partition. Test it on one movie of your choice and the partitions you produced on the previous questions (and report if the value seems ok)

The file is **modularity.py**. To calculate modularity of a network, run as:
python3 modularity.py graphfile.gml attribute

Graphs should be in GML format. I used the "networkx" module for graph representation. "attribute" is the name of the node attribute used to represent the graph partitioning. Since I used Gephi for community discovery in exercises 6a) and 6b), the attribute will be "ModularityClass".

The values for graph modularity are equal to those calculated in Gephi. Example for LOTR network (Gephi modularity = 0.472):



```
hb@hb:~/Desktop/netsci-hwk2/COMMUNITY_DISCOVERY$
 python3 modularity.py lotr.gml ModularityClass
0.4720882681995294
```

Figure 5 - Example output for modularity calculation

d. Implement (in any programming language) a simple greedy agglomerative algorithm. Make a plot showing the modularity increase as you are making more iterations until you reach you a ”local maximum”, and report the communities you found (as a visualization), comparing them to the communities found previously.

The file is community_discovery.py. Should be used as:
python3 community_discovery.py graphfile.gml

Initially, a separate community is created for each node.

While there is an increase in modularity, iterates over all nodes (n1). For each of these nodes, iterates over the nodes neighbours (n2) and changes n1 to n2’s community. Uses the modularity function from the previous exercise. If there is an increase in modularity, updates best modularity value and adds n2’s community as a possible change for n1. After iterating over the neighbours, changes n1 to the community that produced the best modularity value.

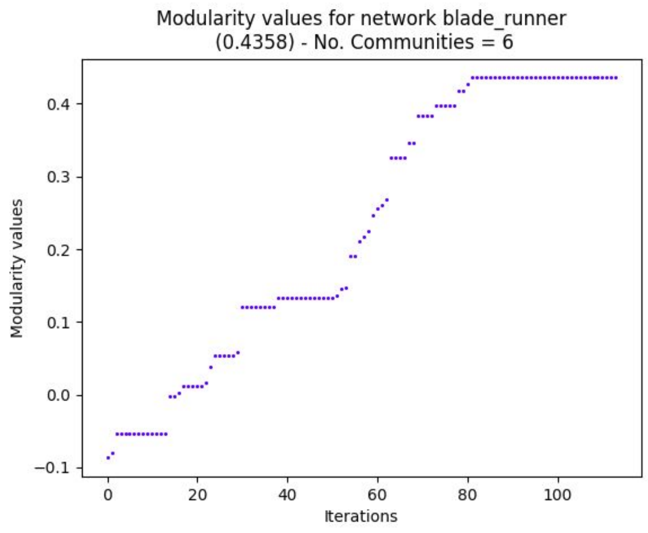
Stores each node’s community as a node attribute called “CommunitiesGreedy”.

Stores the resulting graph as graphfile_sol.gml.

Stores a plot showing the evolution of modularity values as graphfile_plot.png. This plot has the final modularity value and number of communities annotated.

The plot shows the best modularity value for each iteration that tried a community change.

Note: Both these programs take into account **edge weights**.



Plot 2 - Modularity values throughout iterations

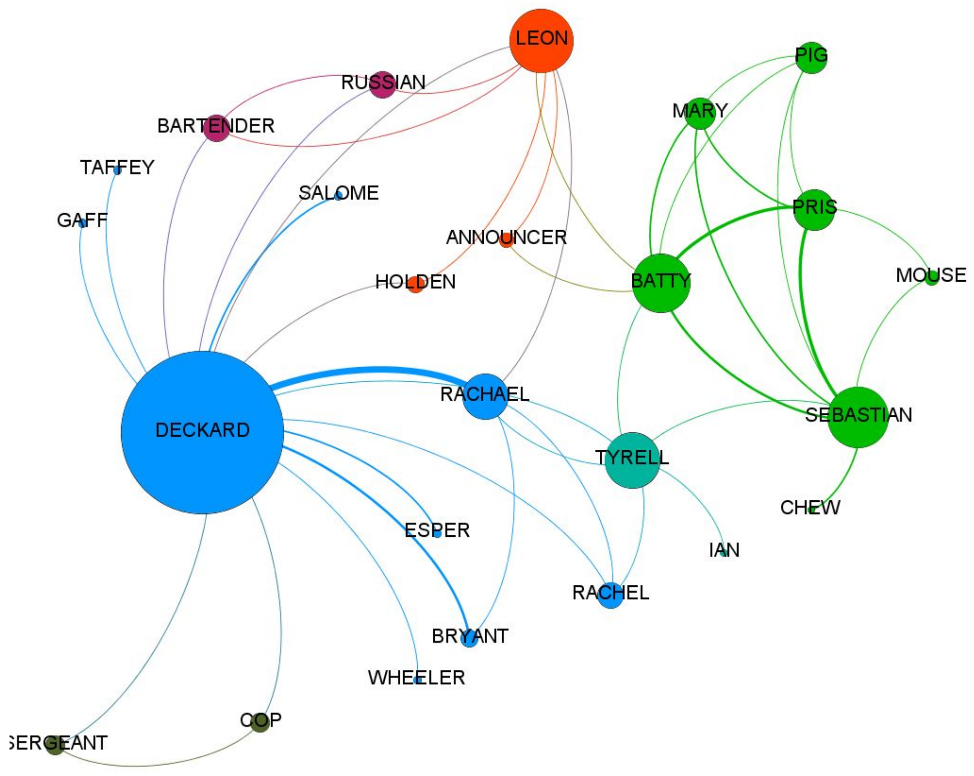
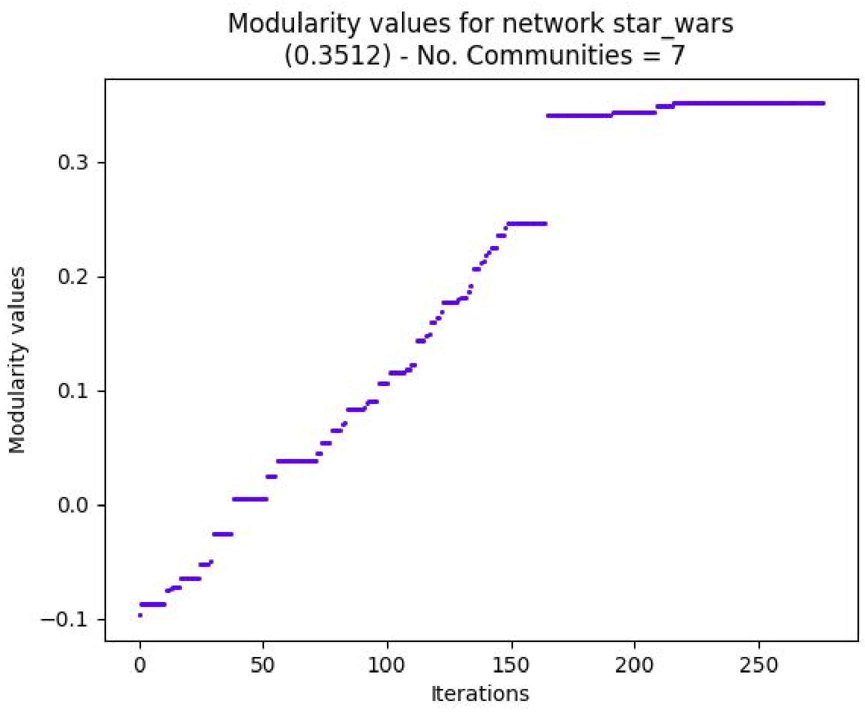


Figure 6 - Blade Runner communities (greedy algorithm)

Modularity for blade_runner was 0.466 with 3 communities with Gephi's Louvain algorithm and 0.4358 with 6 communities with my greedy algorithm.

With my greedy algorithm, three more communities were produced, namely one with Sergeant and cop, who almost always appear together when talking to Deckard, so it still is a meaningful community; one with Tyrell and Ian, which I believe to be significant since Tyrell shares his time in the movie either with Deckard or with the Replicants, which makes him more of an isolated character; one with Russian and Bartender, which I would say is due to the link they share, and that is the least significant community movie-wise.



Plot 3 - Modularity values throughout iterations

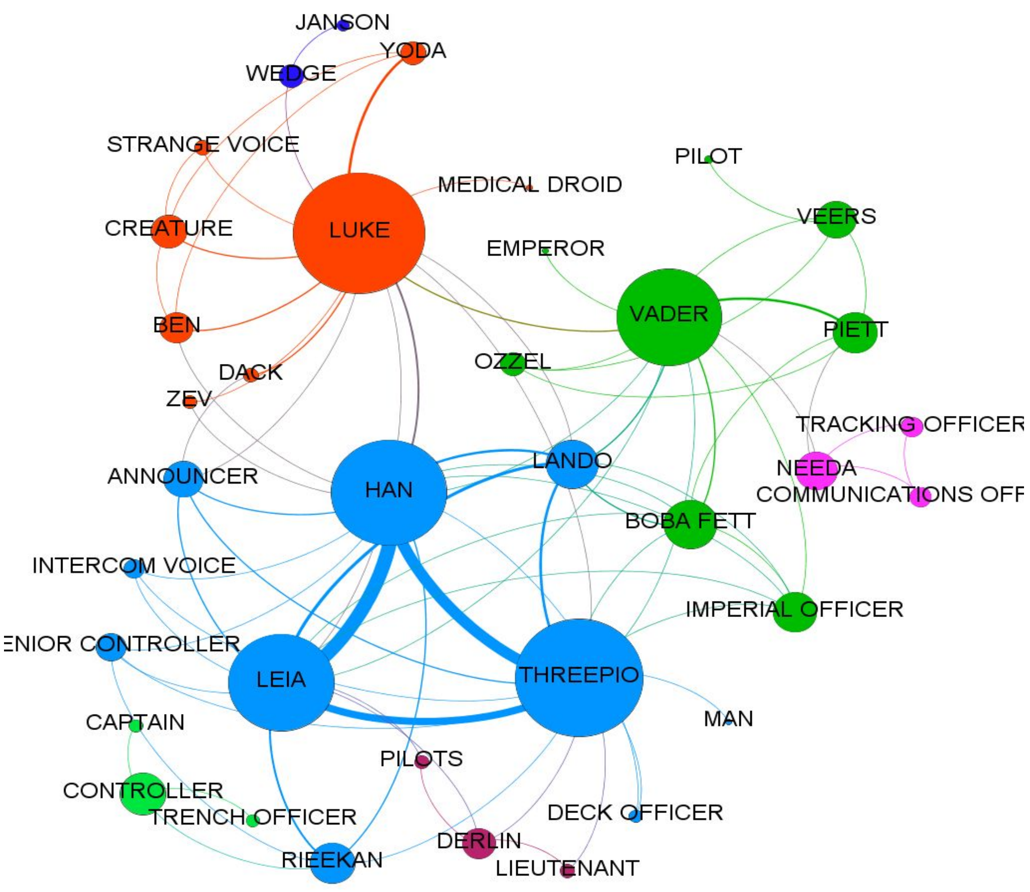


Figure 7 - Star Wars communities (greedy algorithm)

Modularity for star_wars was 0.356 with 5 communities with Gephi’s Louvain algorithm and 0.351 with 7 communities with my greedy algorithm.

Two more small communities were created, constituted of minor characters who interact with each other, that were previously part of Luke’s community and the Empire community. Also, Lando is now on the light side of the force.

On a side note, I want to say I should have used the same colours for the communities that remained the same for better visualization but I only thought of this in the end :-)

e. Using your previous program as a basis, explain how could you obtain a larger quantity of communities? And how could you obtain less communities?

To obtain a larger quantity of communities, I could add a parameter that limits the size of each community. Therefore, by limiting the maximum size of each community, I will obtain more communities (if the limit is 1, each node will be in a different community).

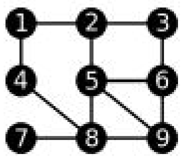
To obtain less communities, I could limit the maximum number of communities. I could start with all nodes in the same community, and try to change nodes between these communities, either because they are empty or because a neighbour node is already in them.

Network Motifs

7.

All outputs and code for this part are in the folder “MOTIFS”.

a. **Your task here is to determine the number of occurrences of all subgraphs of size 4 in this network.**



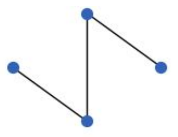
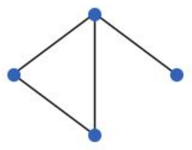
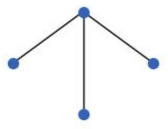
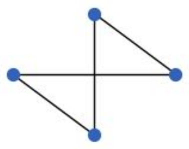
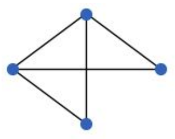
Subgraph	Frequency
	21
	5
	4
	1
	1

Table 4 - Undirected 4-subgraph occurrences

Visualization was made with the graph visualizer I made for my thesis.

b. Imagine you have a Gn,p undirected Erdős–Rényi random network. What is its expected number of triangles ? And what about the expected number of chains ? Justify your answer.

The expected number of triangles is $C_{n,3}$ * p^3.

Let's fix 3 nodes i,j,k E V. The probability that these nodes form a triangle is the probability of having 3 edges, which is p^3, where p is the probability for edge creation. Let $X_{i,j,k}$ be an indicator random variable that takes value 1 if these three nodes form a triangle on the graph G. or 0 otherwise. Let N_Δ be the number of triangles in the graph. Then:

$$N_\Delta = \sum_{i,j,k} \chi_{i,j,k},$$

Where the sum is taken over all unordered triplets of nodes. The number of unordered triples in a graph G is $C_{n,3}$. By linearity of expectation:

$$\mathbb{E}[N_\Delta] = \sum_{i,j,k} \mathbb{E}[\chi_{i,j,k}] = \sum_{i,j,k} \mathbb{P}(\chi_{i,jk,} = 1) = \binom{n}{3} p^3$$ [2]

The same logic can be applied to chains. Taking 3 nodes i,j,k from the graph G, there are 3 ways they can form a chain ((i,j) (j,k) ; (i,j) (i,k); (i,k) (j,k)). Each chain has 2 edges. The probability for each combination is therefore p^2. The expected number of chains is then $C_{n,3}$ * $3p^2$.

c. Your task is now to find some network motifs of the transcriptional regulation directed network of the bacteria Escherichia coli. Check the results and report on what is the more overrepresented subgraph. Check if your very simplistic analysis is consistent with the known literature.

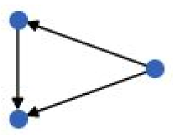
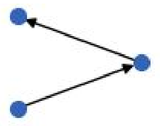
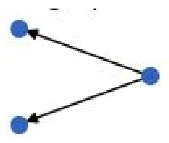
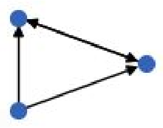
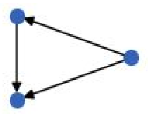
Subgraph	Frequency	Z-score	Rnd_Avg	Rnd_Dev
	130	36.47	11.87	3.24
	250	-6.99	344.88	13.57
	168	-15.63	277.86	7.03
	126	-19.42	237.73	5.75

Table 5 - Motif analysis for size 3 directed subgraphs for ecoli network

The most overrepresented subgraph was  , which is know in the literature as the **feed-forward loop**, a very common subgraph in transcriptional gene regulation networks.

d. Motif fingerprints for 9 networks.

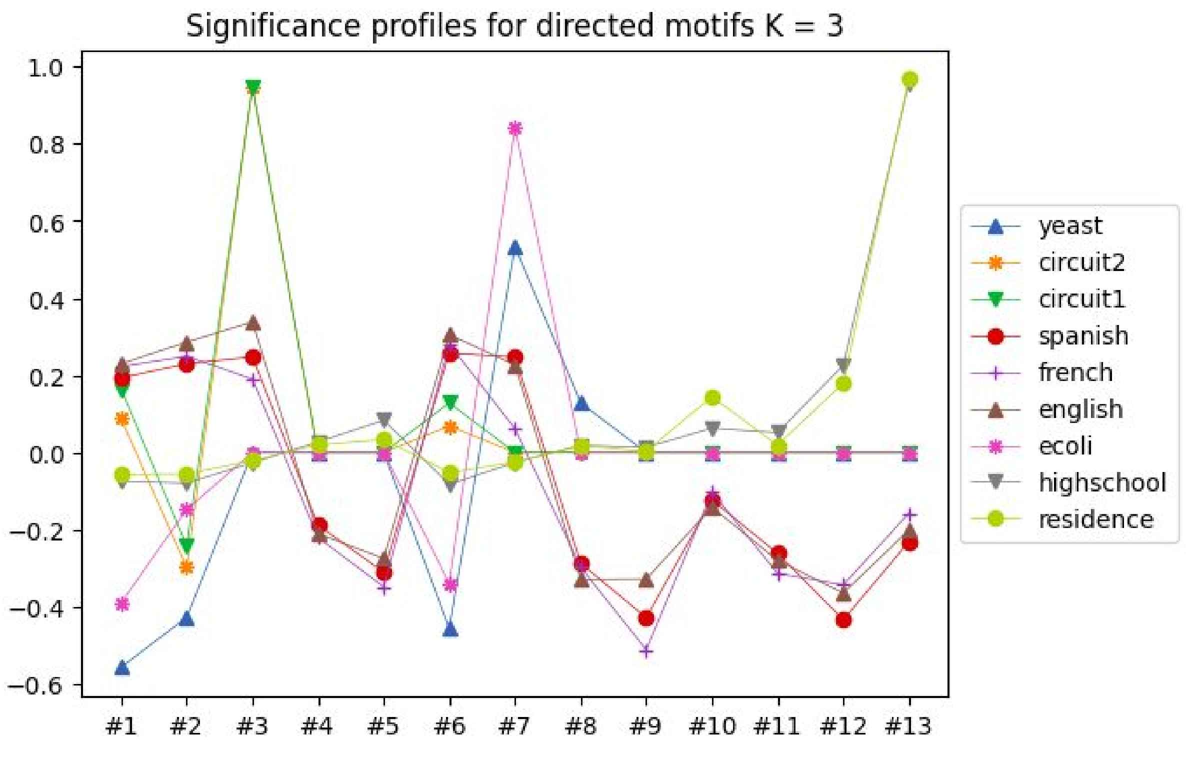
For this exercise, I created a python script (**statistics.py**) that takes all raw output files inside a folder and produces a significance profile plot of all the motifs reported there, as well as a heatmap. From the heatmap, it takes networks with correlation coefficient > 0.85 and groups them in sets. Then, it plots the significance profiles of networks from the same set, with each plot corresponding to each set.

The files should be named as raw_xxx.txt , where xxx is the network name it corresponds to. The script outputs 4 files:

- SP_profiles.png - This is a plot with significance profiles of all networks in the folder. The x-axis corresponds to the ID number attributed to each subgraph that is represented.
- subgraphs.png - This is a subgraph visualization that corresponds each subgraph with its ID.
- heatmap.png - Heatmap for SP correlation between networks
- SP_profiles_classes.png - SP plots for each network set created

I used 1000 random networks for each original network. I ignored subgraphs that occur only once in the original network. All raw outputs are in folder “MOTIFS/exD”. To run the script, simply run “python3 statistics.py”.

For the 9 networks that were presented these were the outputs:



Plot 4 - SPs for all networks

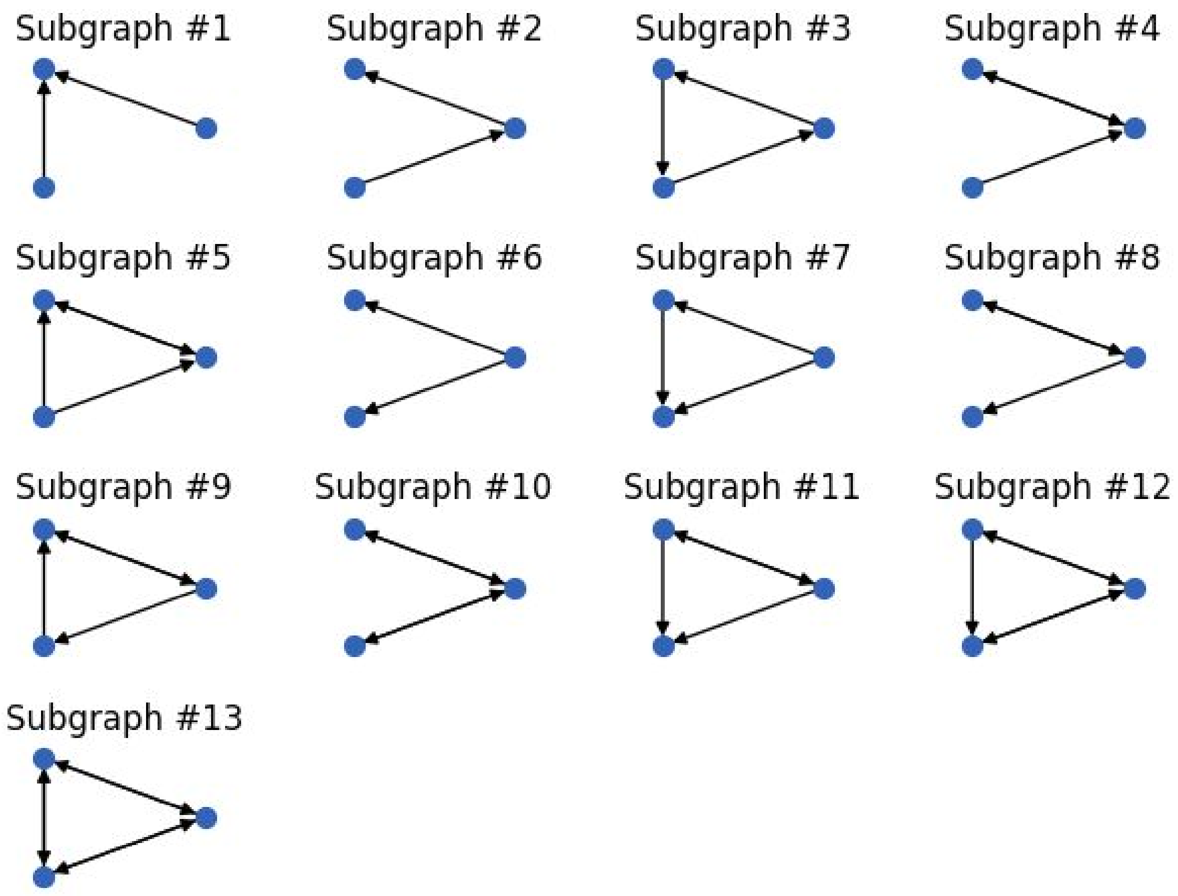
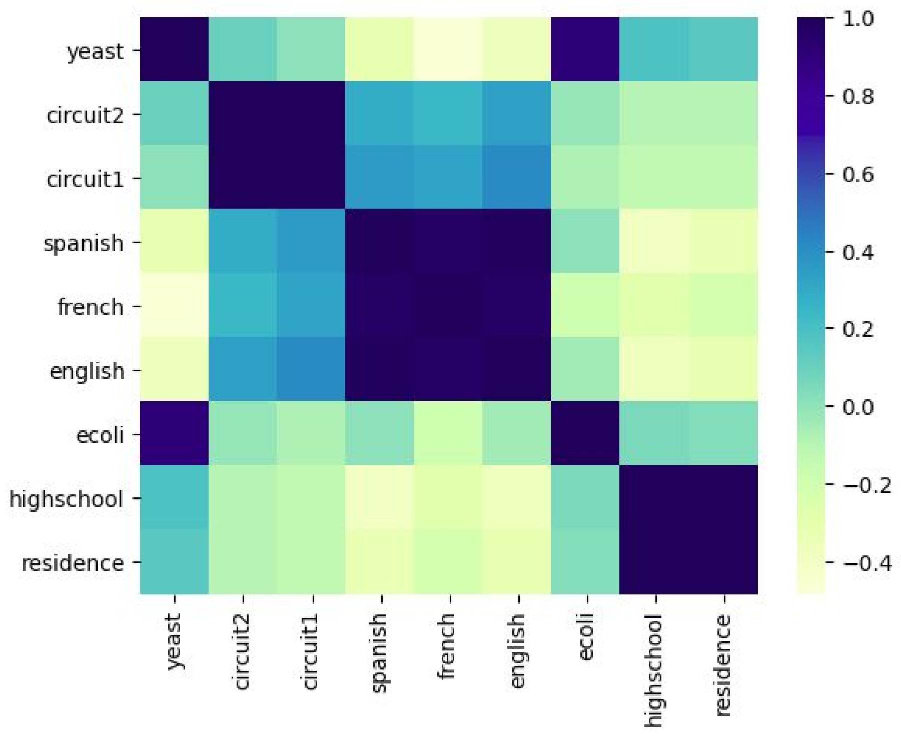
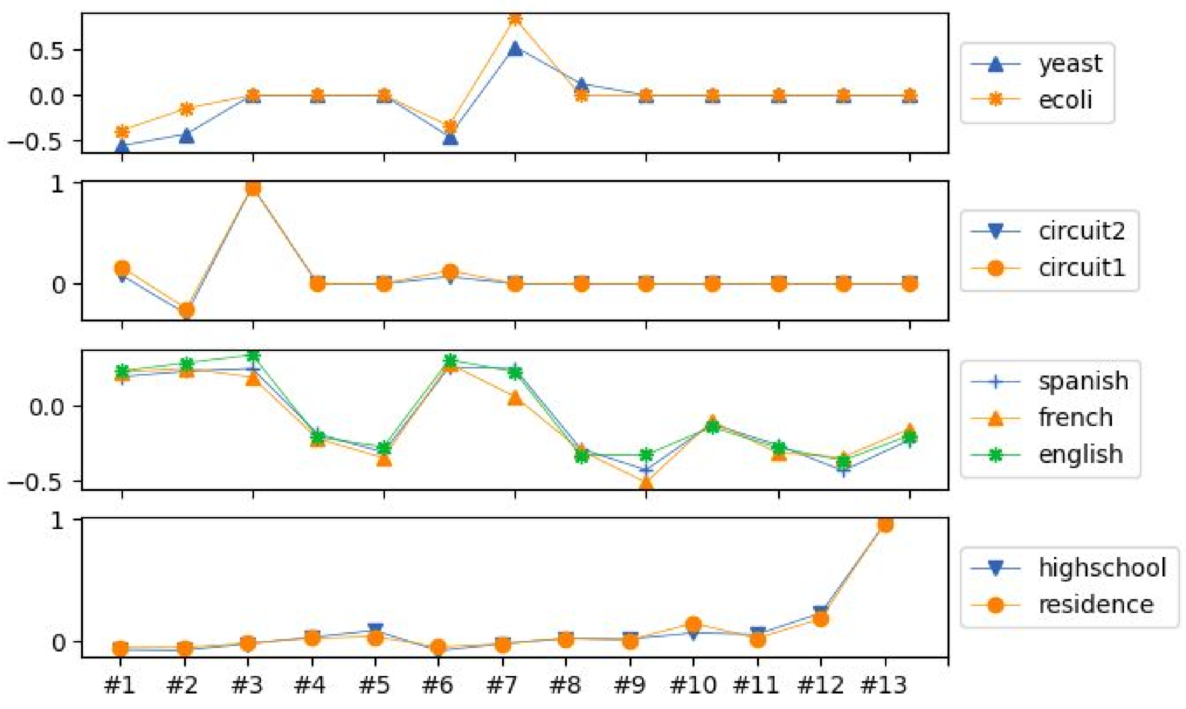


Figure 9 - Directed 3-subgraphs correspondence for exercises 7.e) and 7.d)

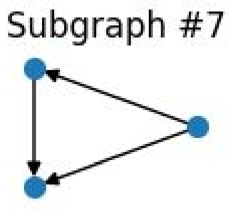


Plot 5 - Heatmap of correlation between network's SPs



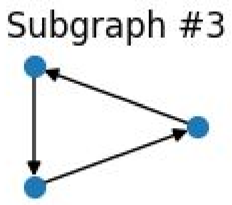
Plot 6 - SPs of networks divided in sets

Transcriptional gene regulation networks



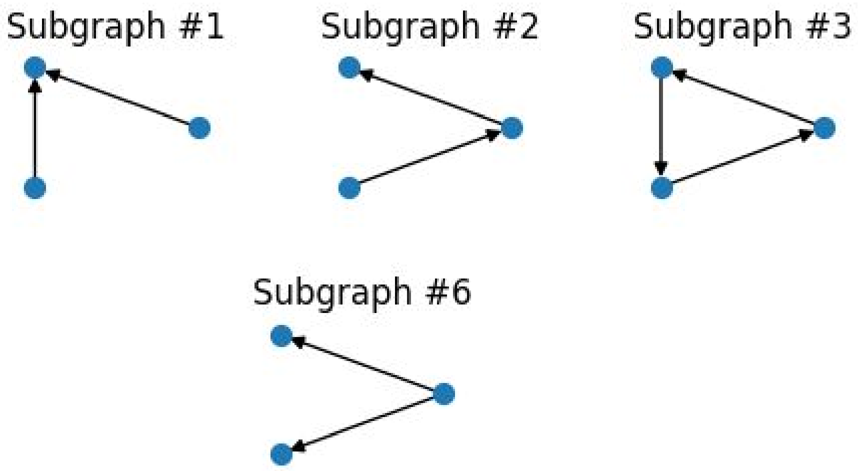
In the yeast and ecoli networks, the motif found was the feed-forward loop (subgraph #7). This type of loop is common in these networks because they are “sensory networks”, meaning they have to respond in minutes to transient signals, so subgraphs like #1, #2 or #6 are not very common because they create extra steps in this process. The feed-forward loop has been shown to be responsible for many signal-processing tasks in these systems.

Networks of digital fractional multipliers electronic circuits



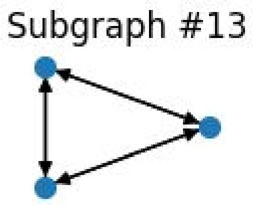
The motif found in these circuit networks was subgraph #3, also known as the 3-node feedback loop

Word-adjacency networks taken out from books



The motifs found were subgraphs #1, #2, #3 and #6. Subgraph #6 is mentioned in [1] as underrepresented, so I cannot find an explanation for it. Since words tend to be followed by words of different classes, closed triads are not very common in these networks, which is noticeable by looking at the underrepresented subgraphs in these networks.

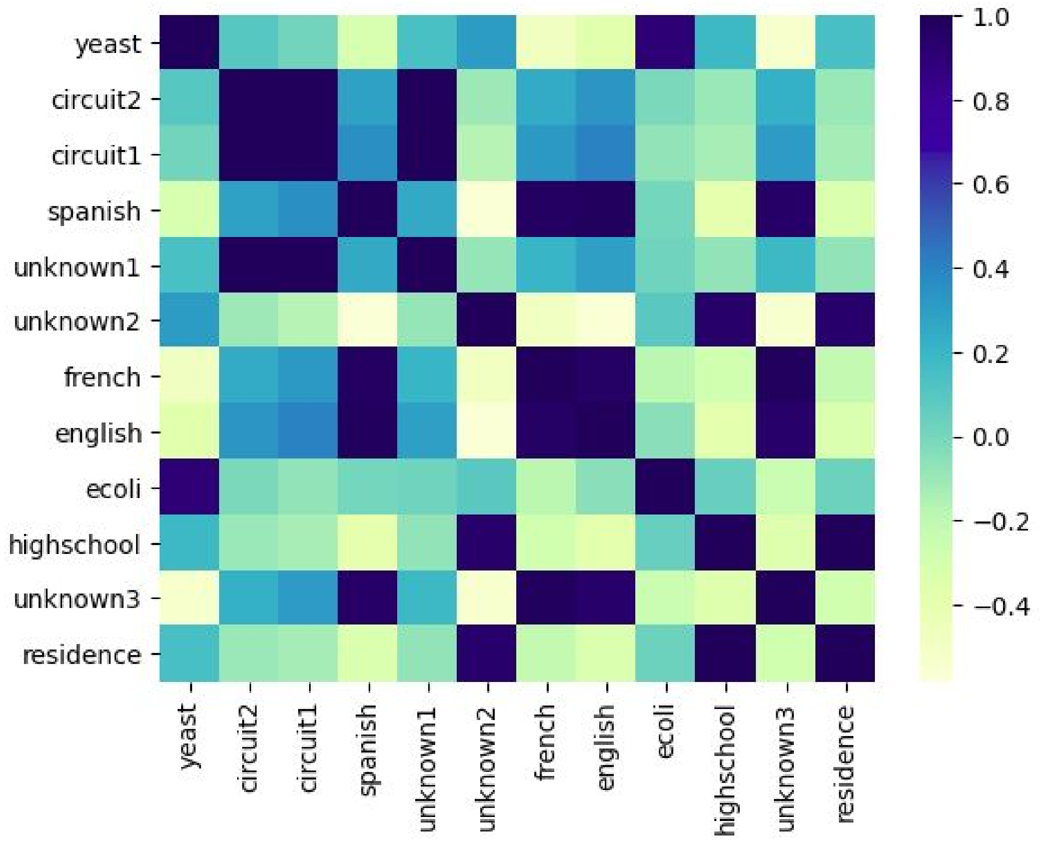
Networks of human friendships



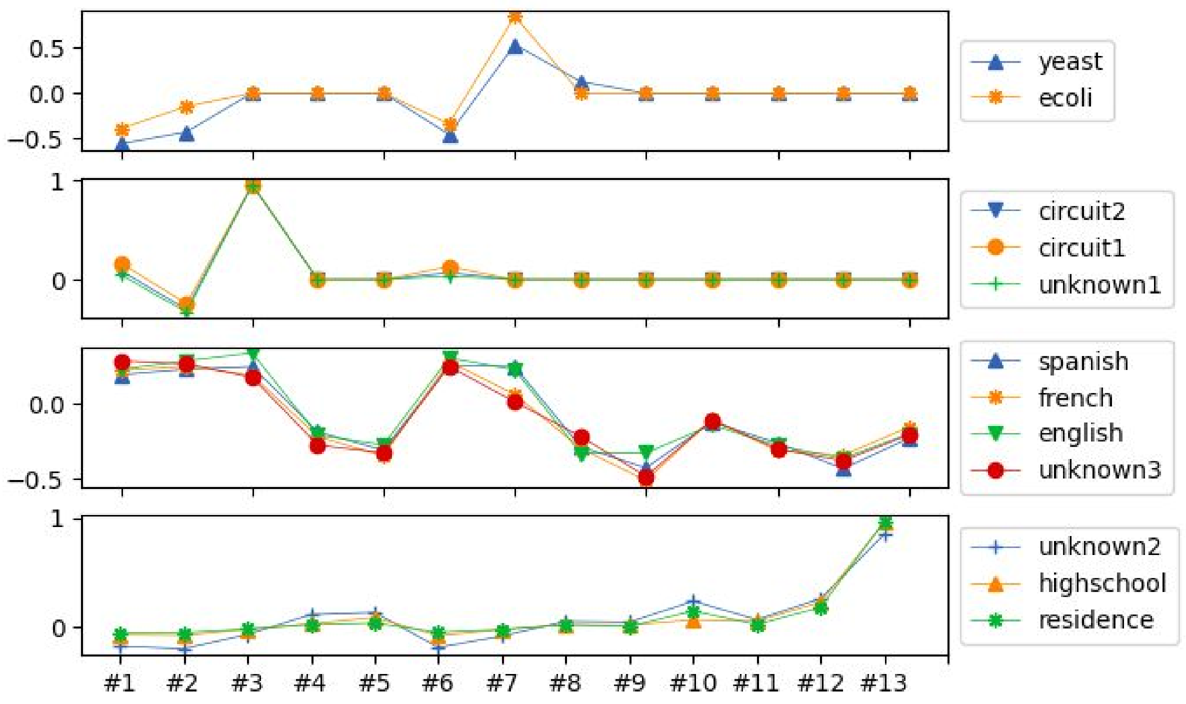
The motif found in these networks was subgraph #13. These networks represent friendship relationships, and this motif shows the “friend of a friend is my friend” relationship.

e. Your task is to find the ”family” of the three ”unknown” networks given in unknown.zip. You should justify your answer by computing and plotting their motif significance profiles and by adding them to the previous heatmap. Each network will clearly belong to one of the groups discovered on the previous question.

For this exercise, I simply placed the “raw_unknownx.txt” outputs in the same folder as the previous raw outputs and used the script I previously mentioned. The outputs were the following:



Plot 7 - Heatmap of correlation of all network’s SP



Plot 8 - SPs for networks divided by sets

- **unknown1** - Network of digital fractional multipliers electronic circuits
- **unknown2** - Network of human friendships
- **unknown3** - Word-adjacency network

References

[1] Milo et al. ”Superfamilies of evolved and designed networks.” Science 303.5663 (2004)

[2]https://people.maths.bris.ac.uk/~maajg/teaching/complexnets/random_graphs.pdf

[3] Milo et al. “Network Motifs: Simple Building Blocks of Complex Networks”. Science 298 (5594), 824-827. [doi:0.1126/science.298.5594.824]