Modelling Static Complex Networks

Domonkos Haffner (S24Q2W)

March 8th 2021

Contents

1 Introduction

Whenever one works with networks or graphs, the visualisation of connections is quite useful. Even though the computer itself doesn't benefit from the visual representation, the person working on the project may get some ideas about new modelling strategies after the visualisation, when working with networks. Networks are made up of a collection of two objects:

- Nodes: represents and entities.
- Edge: represents the connections amongst the entities.

The nodes usually have some attributes/properties, for which the whole network gives the connections amongst these. When these networks get more and more complicated, it's crucial under the modelling process to visualise them, so it showcases as much information as possible.

2 NetworkX

In the following research, I'll be analysing and modelling different networks and graphs with the NetworkX Python library. The tasks and exercises for the semester are written in a github repository, which can be found through the following link: [1]. The first task was to choose a couple data sets, which can be represented as networks. The data sets I chose are the following: The first one is about steroid usage amongst baseball players [2]. The second data set contains information about domestic flights in the UK in the year of 2003 .[3]

After getting a little bit familiar with the data sets and the NetowrkX library, I tried out different layout schemes for each of these two networks. The results are interpreted in the Results and discussion section.

2.1 Baseball data set

The following graph 1 shows the 72 different players in relation to having the same steroid supplier. There is another data set in the package I downloaded, which shows all the suppliers and the different players, who bought steroid from them, however, that graph will only be presented in the second progress report. In graph 1 the common steroid suppliers can be seen for each player, where the different players are represented with numbers from 1 till 72. As it's clearly seen, most people used the same steroid supplier, while only a smaller portion of the people used different ones.

2.2 Flights data set

The other data set I chose to visualise was about domestic flights in the UK. The graph is very simple, it shows the connections between the cities, from which and to which flights flew. The visualised network can be seen in graph 2.

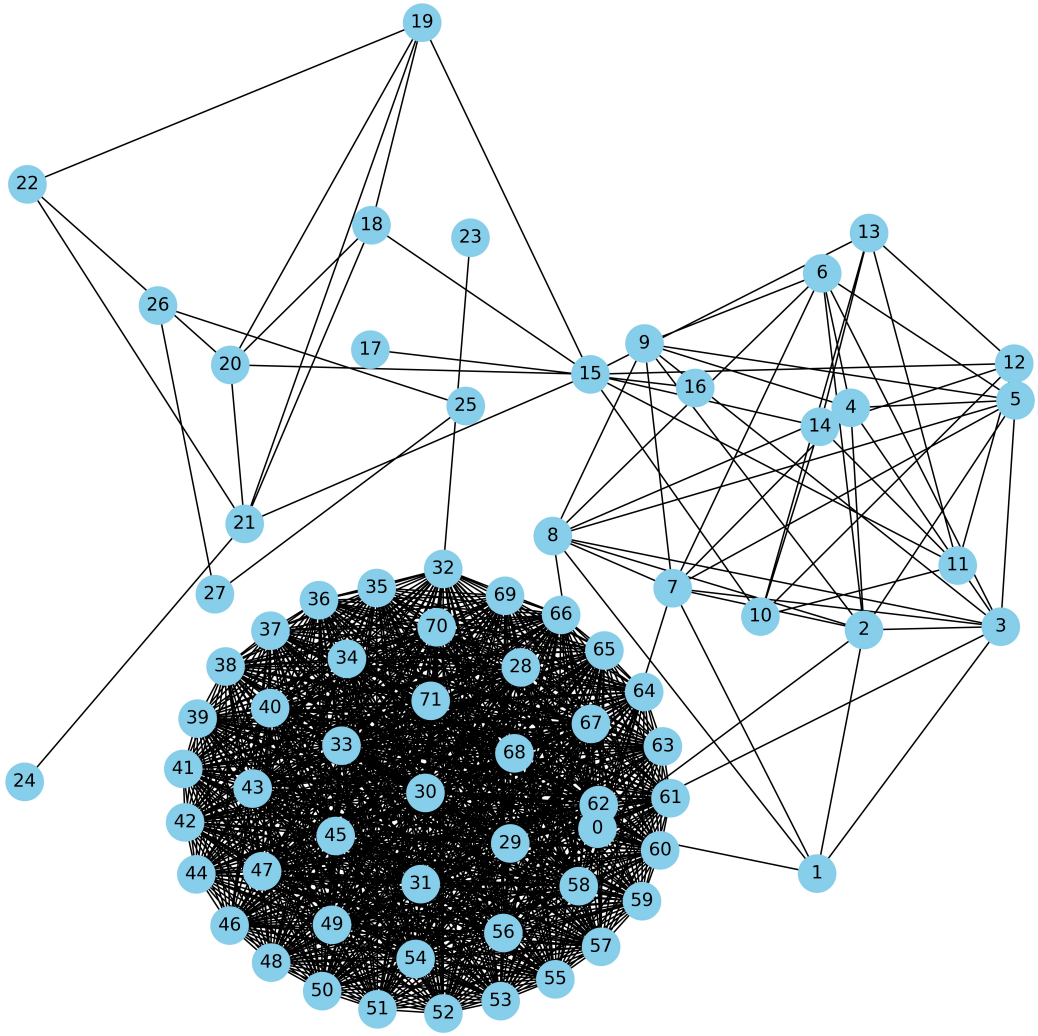

Figure 1: Common steroid suppliers of the different baseball players.

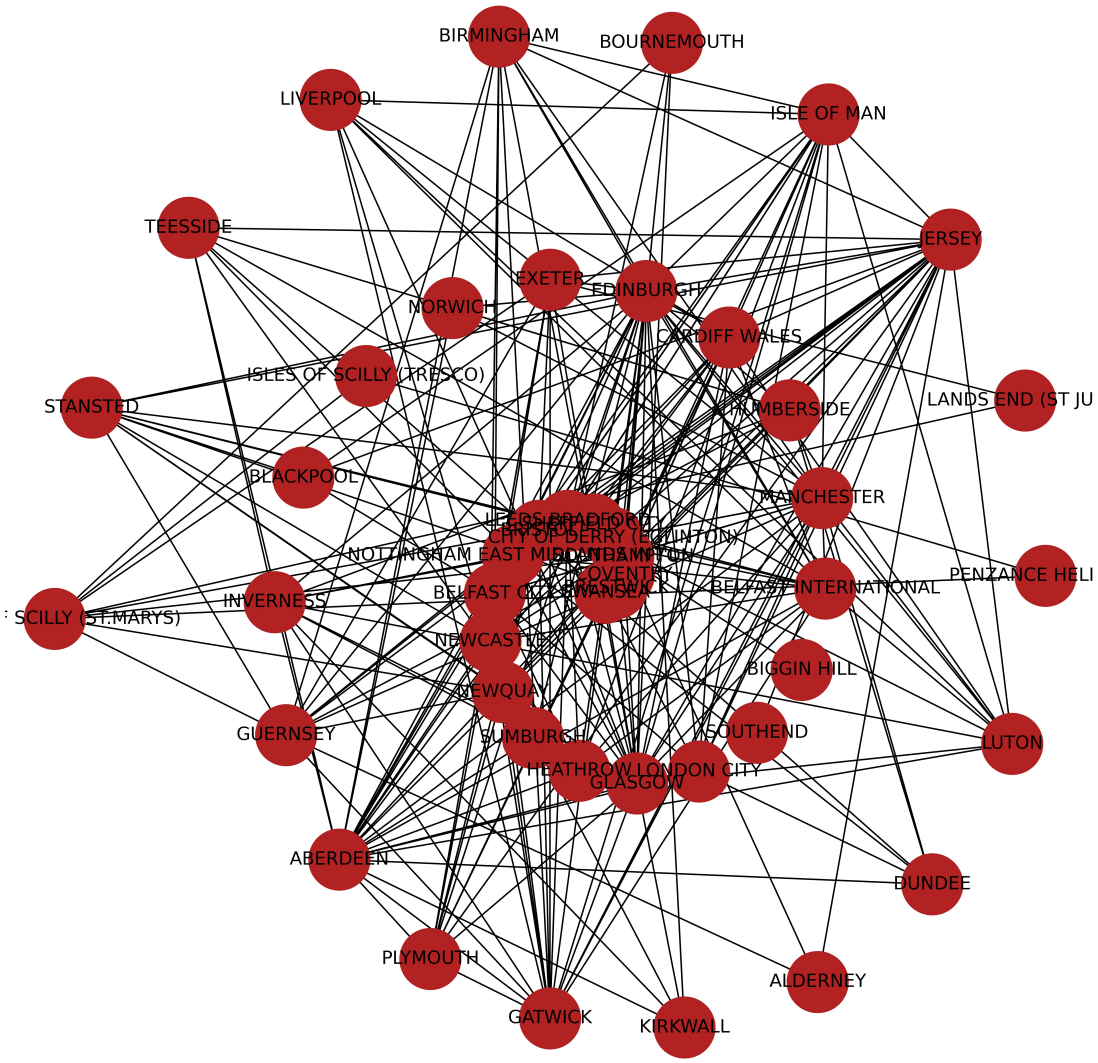

Figure 2: Domestic flights from the UK in the year of 2003.

3 Cytoscape

Cytoscape is an open source network-modelling software platform, created in 2002 at the Institute of Systems Biology in Seattle. It's mainly used for analysing and visualising networks, by creating graphs and calculating the different characteristics. One needs to load the network by giving an input consisting of the nodes and the edges. [4][5]

I was encouraged to use Cytoscape for this network visualisation project. The data set I first visualised is about protein-protein interactions. The properties of the network is the following:

- Edge Type: Co-complex association,
- Node Type: Protein Interactions,
- Avg Edges: 9,070,
- Avg Nodes: 1,622.

After importing the data set into the software, Cytoscape asks the user to choose the source- and target node, so the visualisation may begin. After selecting the nodes and edges, the graph is created. These are the steps I followed through with the protein-protein data set with the results in graph 3.

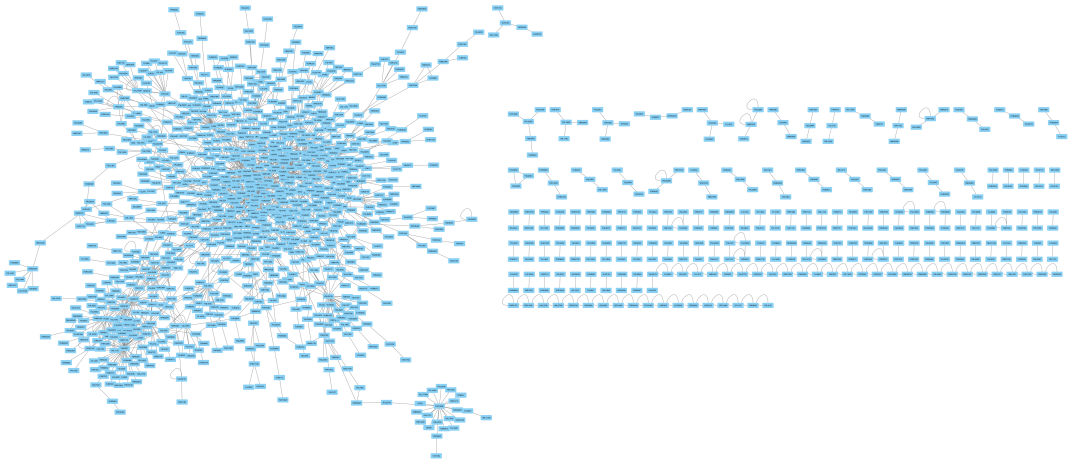

Figure 3: Protein-protein interaction graph.

As it's seen, the left part of the graph is a well connected, dense network, while the right part is consisting of proteins, which are connected to only a few other ones. This was nothing more than just loading and visualising the network. There are many more features of this software, which I'll be actively using in the rest of this network-research.

4 Graph-tool

Graph-tool is a Python library for creating and manipulating different networks. The installation works in several ways on different operating systems. Since I'm using Windows, I used docker for installing the library. Graph-tool is able to draw an extensive amount of graphs with degree/property histograms, vertex-vertex correlations, average vertex-vertex shortest path, etc.

Graph-tool supports several graph-theoretical algorithms, from which hierarchical block models are going to be used in this research.

After installing Graph-tool with docker, I tried to create a very simple network, which consists of two nodes and a directed edge. This can be seen in figure 4.

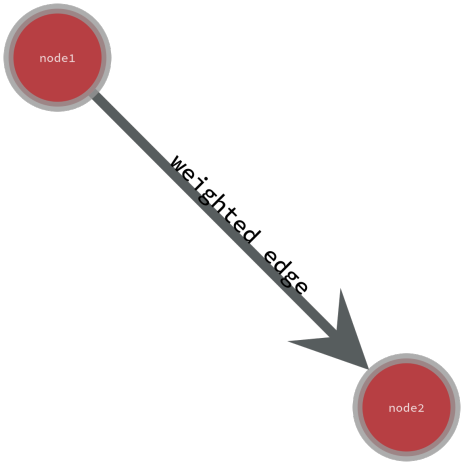

Figure 4: Two nodes example with Graph-tool.

5 Theoretical Background

5.1 Degree Centrality

The investigation of centralities in networks is very important when researching connections amongst the nodes. The centrality measure indicates the importance of nodes within the graph. Centrality measures can be crucial when for example investigating virus spreads or new friendships. It technically gives us an overview on how fast the information can spread.

One of the first centrality measures conceptually created was the degree centrality. This is a very easy measure to calculate. The degree is a basic network property, describing the connections a node has to other edges. This describes the number of link incidents upon a node. When the network is directed, the indegree and outdegree are distinguished; the indegree describes the head ends, while the outdegree describes the tail ends. The degree centrality is simply the degree of the given node. For example, a node with 10 social connections has a degree centrality of 10, while a node with only 2 edges has a degree centrality of 2. [6] [7]

Let's take a look at the following example: I chose a data set about US flights in the year of 2010 with the edge number of 5960 and node number of 500. The airports are represented by numeric codes. The degree distribution can be easily calculated by Networkx's built in functions. Since the result is given as a normalised version of the actual distribution, the values still need to be multiplied by the maximum number of edges possible, if one wants to have a non-normalised version of the degree distribution. After calling the functions, a figure for the degree distribution can be created, which is usually plotted on a log-log scale. The result can be seen in figure 5.

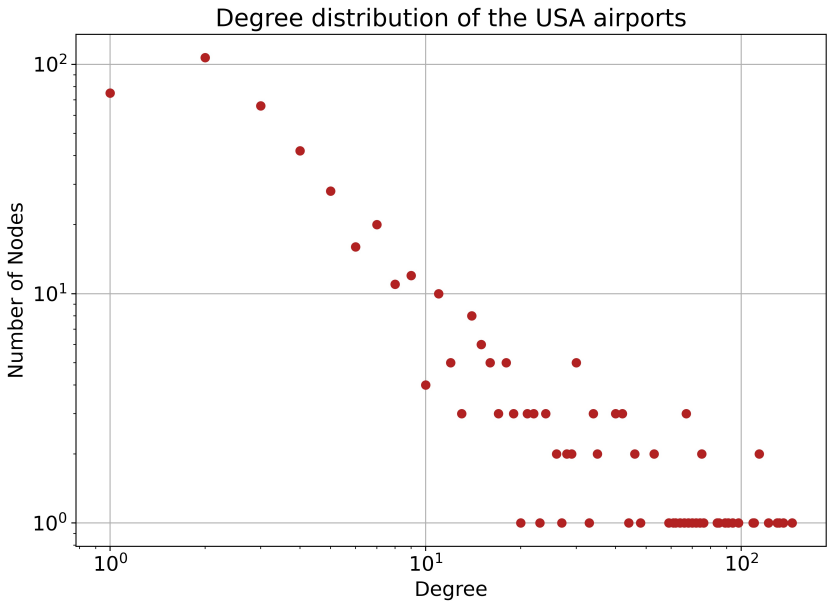

Figure 5: Degree distribution of flights in the USA in the year of 2010.

5.2 Closeness Centrality

Another centrality measure to take into consideration is the closeness centrality. It indicates how close a node is to all other nodes inside of a network. This gives information about the placement of nodes, capturing the mean distance amongst all nodes in the network. When the closeness centrality has a low value, it indicates that either the network is well connected, or it is small. However, when this value is a high value, it means that the node has a less central position [8].

The normalised closeness centrality - C - is calculated by taking the reciprocal of the average shortest path lengths - d - to the $n-1$ reachable nodes. For node u: [9]

$$C(u) = \frac{n-1}{\sum_{v=1}^{n-1} d(v,u)}. \tag{1}$$

Let's take the following example: In figure 6 a small graph can be seen. When one wants to calculate the closeness centrality from point E, one needs to consider the average shortest path length from the node. This can be seen in the 5.2.

Node	A	B	C	E	F	G	H
Shortest path from D	3	2	1	1	2	2	1

After calculating the path lengths, the mean value of this must be taken for the average shortest path lengths:

$$\frac{3+2+1+1+2+2+1}{7} = \frac{12}{7} = 1.71. \tag{2}$$

After receiving this value, simply the reciprocal of it should be taken, to yield the non-normalised closeness centrality of node D.

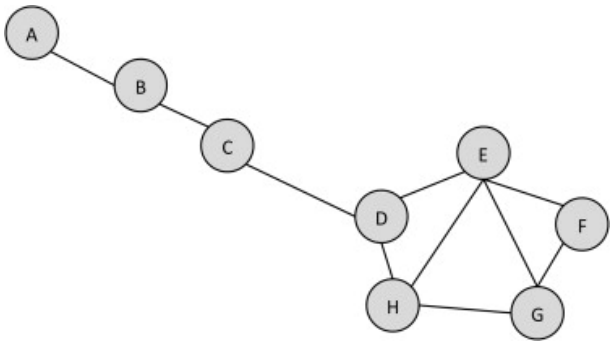

Figure 6: A small graph

$$C(D) = \frac{1}{1.71} = 0.58. \tag{3}$$

I calculated all the closeness centralities with python, which can be seen in 5.2.

Node	A	B	C	D	E	F	G	H
Closeness centralities	0.29	0.39	0.5	0.58	0.54	0.39	0.41	0.5

Let's take another example. The UK flights data set was previously illustrated by the Networkx library. Now I'll create a similar graph with Cytoscape. This can be seen in figure 7.

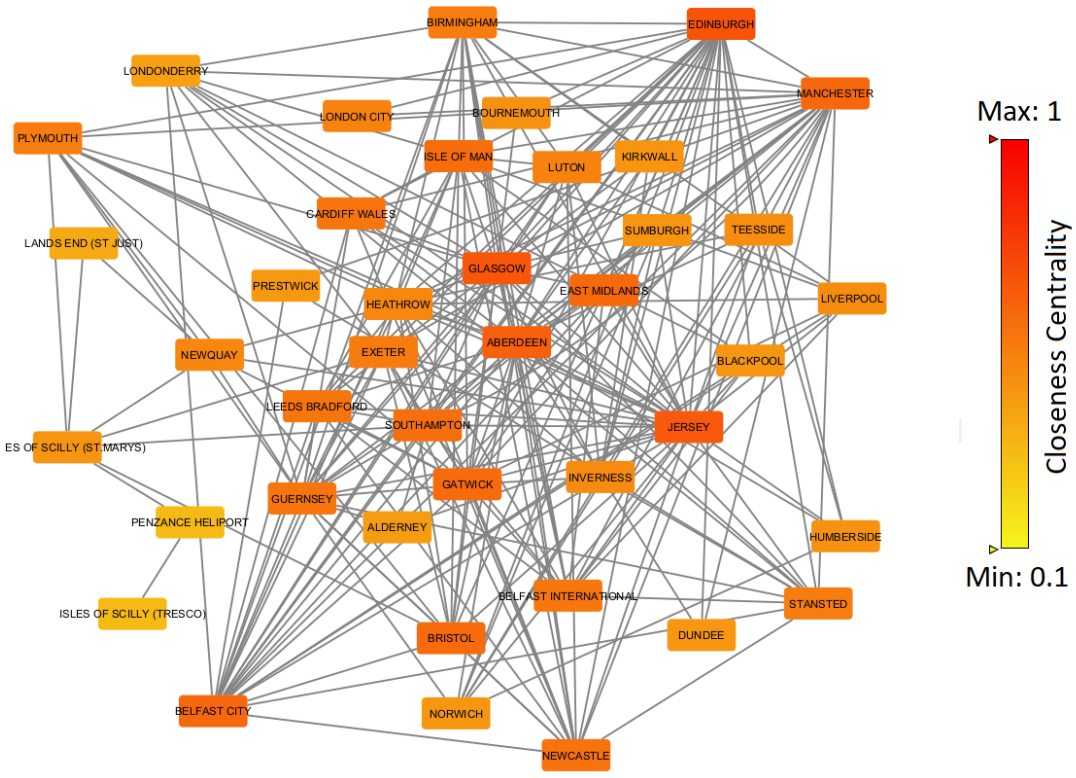

Figure 7: Heatmap of the US flights network according to the closeness centrality.

The minimum value of the closeness is around 0.3, while the maximum value is around 0.7. As it's seen, Cytoscape allows to apply a continuous mapping when plotting the network. [10]

5.3 Eigenvalue Centrality

Another type of centrality measure worth mentioning is the eigenvector centrality. Let's take the following example in a social network: even though one might have not many connections, if those one has have a high degree centrality, one can be very influential in the network. The eigenvalue centrality technically describes the benefit of having a connection to a node. The more connections a node has, the more benefiting it is to connect to that node. Another social example: a person on Facebook who has 300 unpopular friends has a lower eigenvalue vector than a person with 200 very popular friends. [11] [12]

When calculating the eigenvalue centrality, the adjacency matrix a occurs, which either has the value $a_{ji} = 1$, if vertex i is linked to vertex j, or has the value $a_{ji} = 0$, if there is no edge between i and j. Then the eigenvalue centrality x_j corresponding to the node j can be defined as the following: [13]

$$x_j = \frac{1}{\lambda} \sum_{t \in G} a_{j,i} x_i, \tag{4}$$

where G represents the network and λ is a constant. This can be written in an eigenvector equation:

$$\underline{\underline{A}} \cdot \underline{x} = \lambda \underline{x} \tag{5}$$

Let's take the protein-protein interaction as an example. As it's seen in figure 3, there are a lot of nodes and edges in the central part, however there are lots of nodes with only one or two connections. This implies a very low eigenvalue centrality. The eigenvalue centrality distribution can be seen in figure 8.

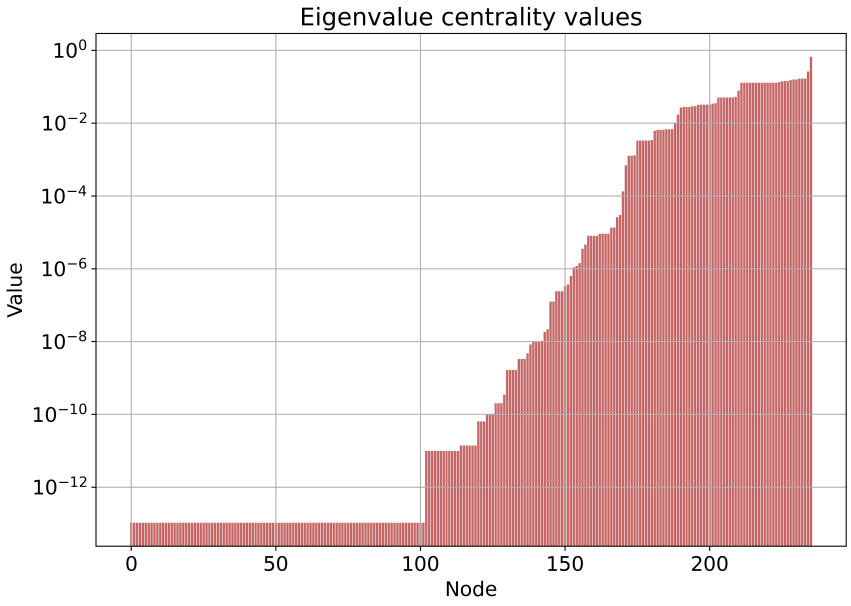

Figure 8: Eigenvalue centrality distribution of the protein-protein interaction data set.

As it's seen in the distribution, there are lots of nodes which have a quite small eigenvalue centrality, however, there is one (the maximum) which has a value of 0.66. This implies a node,

which is very well connected to other highly connected nodes.

6 Conclusion

In this part of the report, we got familiar with the Cytoscape software, which will be very useful in modelling the different networks. Moreover, the degree-, closeness- and eigenvalue centralities were defined with different network-examples.

References

[1] Modelling Static Complex Networks with Networx:
https://github.com/sdam-elte/modellinglab2019/tree/master/networx

[2] UCINET Software Baseball Steroid Use:
https://sites.google.com/site/ucinetsoftware/datasets/covert-networks/baseballsteroiduse

[3] Alessio Cradillo Datasets - UK flights 1990 - 2003:
https://www.bifi.es/~cardillo/data.html

[4] Cytoscape Introduction:
https://en.wikipedia.org/wiki/Cytoscape

[5] Cytoscape tutorials github page:
https://github.com/cytoscape/cytoscape-tutorials/wiki

[6] ScienceDirect - Degree Centrality:
https://www.sciencedirect.com/topics/computer-science/degree-centrality

[7] Radicchi, Filippo Castellano, Claudio Cecconi, Federico Loreto, Vittorio Parisi, Domenico. (2004). Defining and identifying communities in networks. Proceedings of the National Academy of Sciences of the United States of America. 101. 2658-63. 10.1073/pnas.0400054101

[8] ScienceDirect - Closeness Centrality:
https://www.sciencedirect.com/topics/computer-science/closeness-centrality

[9] Networkx Closeness Centrality:
https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.centrality.closeness_centrality.html#networkx.algorithms.centrality.closeness_centrality

[10] Enrichment Network - Network Visualization:
https://enrichmentmap.readthedocs.io/en/latest/Network.html

[11] ScienceDircet - Eigenvector Centrality:
https://www.sciencedirect.com/topics/computer-science/eigenvector-centrality

[12] Centrality Measures - Dr. Natarajan Meghanathan:
http://www.jsums.edu/nmeghanathan/files/2015/08/CSC641-Fall2015-Module-2-Centrality-Mea
pdf?x61976

[13] Geeks for Geeks - Eigenvector Centrality Centrality Measure:
https://www.geeksforgeeks.org/eigenvector-centrality-centrality-measure/