5-2018

A Comparative Study Of Large-Scale Network Data Visualization Tools

Prakash Joshi
University of New Orleans

A Comparative Study Of

Large-Scale Network Data Visualization Tools

An Honors Thesis Submitted to

The Department of Computer Science

Of the University of New Orleans

In Partial Fulfillment Of the Requirements

For the Degree of Bachelor of Science

With University High Honors and Honors in Computer Science

By

Prakash Joshi

May 2018

Acknowledgments

To my advisor and mentor Dr. Shaikh M. Arifuzzaman: I owe you this thesis. You have been a great teacher and guide throughout the writing process. I highly appreciate the feedbacks you gave me whenever I needed them.

To my advisor and second reader Dr. Christopher Summa, thank you for taking time from your busy schedule and being my second reader. I appreciate your patience and friendly attitude, a lot.

To Dr. Eishita Farjana, thank you for being the third reader and giving valuable suggestions on styling, editing and helping me shape the overall layout of the project.

To Erin Sutherland, from the honors office department, thank you for the support whenever I was stressed about meeting the deadlines.

To my friends Haydar Mahdi and Hiram Molina, thank you for being my resident grammarians.

And finally – I dedicate this thesis to my support system – my parents.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

Abstract

One of the most important parts of Data Analysis is Data Visualization [15]. The easy thing about Data Visualization is that there are hundreds of ways to do it, one better than the other. Ironically, however, it is difficult to choose the right tool for the job. This can be a concern because it is really important to know which tool is best depending on the resources we have. This thesis tries to answer that question – to an extent.

In this thesis, I have tried to compare three Data Visualization tools: Gephi, Pajek and NodeXL. I have mainly discussed what each tool can do, what each tool is best at, and when to and when not to use each tool.

Therefore, using the right tool can not only save us a lot of time by making the task easy and get the work done using a minimal number of resources, but also help to get the best results.

The comparison is based on what Visualization features each tool has, how each tool computes different graph features, and how Compatible and Scalable each tool is.

In the process, I used different Network datasets and tried to calculate certain features of the graph and wrote the findings. The end report discusses which tool can be best to use given the size of dataset, the problem we are trying to solve, the resources we have and the time we can spend.

Introduction

Between the dawn of civilization and 2003, we only created five billion gigabytes of data; now we're creating that amount every two days. By 2020, that figure is predicted to sit at 53 zettabytes (53 trillion gigabytes) - an increase of 50 times [1]. This is a staggering amount of data, which is a wonderful thing. However, data is only as good as the information we can extract from it, that's where data science comes into play. Data science is a way of making sense of data sets, finding patterns and making the best out of them – by solving real problems and asking more questions [13, 14, 15]. For instance; weather data from the past 100 years can be used to make weather predictions in the future, social media data can help present ads to targeted customers based on their likes, following patters and other activities on that particular platform. Protein to protein interaction analysis can help us study which proteins are most important (for a certain problem set) and help us bring closer to knowing more about diseases- such as cancer- and maybe even help cure them. Data science can be used anywhere to study things in depth, find new patterns, solve new problems and make the world a better place to live in [14, 17, 18].

Data Science is a broad field, but to summarize, it basically involves six steps

1. Raw data collection: Raw Data Collection, as the name itself suggests, is collecting data in raw form. This data needs some extraction, organization and sometimes even analysis before it can be made of some use. For example: as a researcher, if we need data tuples

(age, gender, disease history and smoking habit) of people aged 50-60, who visit a hospital on a monthly basis. While collecting data, it is very rare we get exactly the data we need. For example, the hospital might only keep records of their name, address and social security number. The data gathered from the hospital may not be the specific data researchers were looking for, this is the raw data. Raw data will go on to be "cleaned" and only the useful and relevant data kept.

2. Data cleaning and munging: Once the data is collected, it is then cleaned and only the useful data is kept. For example: if we have more data than relevant, we filter it out. Furthermore, if it's not in the right file format, we extract the data and convert it. For example, mined data from the internet usually comes in the Json form, but if we need data in spreadsheet form, we clean and munge it into the Excel.
3. Data visualization and Analysis: Once the selected data is saved in a desired file format, it is analyzed and visualized; usually both done in parallel.
4. Building models and algorithms based on problems we need to solve: Once the data is analyzed, models are build based on the questions we have.
5. Extracting the solution: The built models can then be used to extract questions to our specific answers.

Since Data Science itself is a very big field, because of the time limitations of the thesis, I have decided to focus my work on just Data Visualization part.

Importance of Visualization

Data Visualization is a very important aspect of Data Science. If the data is of very complex type, visualizing it can help understand it with a glance. One does not need to go through each data entity (which, in our case could be millions), but just a quick glance and some study can be enough to make sense out of the particular data. For example: Corporations might look at a graph to verify that marketing and sales are communicating, urban planners to monitor the interconnectedness, or isolation of neighborhoods, biologists to discover interactions between genes, and network analysts to monitor security [2].

Data Visualization is not just about the looks, even though the visualizations can be stunning to look at, readability is always the prime concern. A quick look at a picture can give a lot more information than manually going through the data and analyzing it.

For example, the following network shows the friendship network of a German boys' school class from 1880/1881 [13]. The node size signifies the number of connections; the bigger the node, the higher the connections each student has.

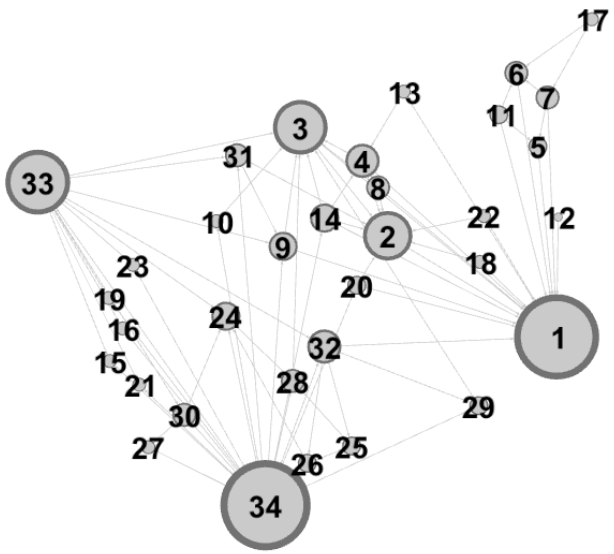


Figure 1: German School Boys' Friendship Network

A quick glance is enough to know what these data is trying to convey and that's the power of Visualization. This is really important when we are deadline enormous amounts of data.

Data is ubiquitous. Hence, it is a natural thing that it comes in all shapes, sizes and types. With computing storage getting cheaper than ever [9], it's no surprise that we can (and we should) save all data. Every second, we are creating new data. For example: we perform 40,000 search queries every second (on Google alone), which makes 3.5 searched per day and trillion searches per year [3].

Also, since there is so much data being produced, it's not possible to work on all kinds of data. And because of the time limits of thesis, I have decided to focus on Large-Scale Network Data.

Network Data and Terminologies

A network is a representation of entities (nodes represented by V) and that are linked *via* lines (edges, represented by E) or arrows (arcs) in cases of directed graphs [16]. The nodes and edges may represent a weight and/or a label. The entities can be anything (people, protein, companies, animals etc.) and the relationships might include anything. Edges are directed when the relation between two nodes u and v is not same as the relation between v and u. Edges are undirected when the relation between u and v is same as v and u.

Network (Graph) Terminologies

Node Degree – The degree of a node is simply how many edges are connected to it. Degree distribution is the probability of these degrees over the whole network as a whole

Out Degree – The total number of edges leaving a vertex is known as Out Degree.

In Degree – The number of edges entering a vertex is known as In Degree.

Size – The total number of edges in a graph is defined as its size.

Weight – A weighted graph associates a label (weight) with every edge in the graph. Weights are usually real numbers. The weight of an edge is often referred to as the "cost" of the edge. In

applications, the weight may be a measure of the length of a route, the capacity of a line, the energy required to move between locations along a route etc.

Network Features

- *Average Degree* - The average number of links per node is called Average Degree
- *Average Weighted Degree* – The average of sum of weights of the edges of nodes is called the average weighted degree.
- *Distance* - The distance between two nodes is defined as the number of edges along the shortest path connecting them.
- *Average Distance* – The average of distance between all pairs of nodes is called the average distance.
- *Network Diameter* - The network diameter is the longest shortest path in a graph.
- *Modularity* – Modularity is a way to measure the strength of division of a network into modules (also referred as groups, clusters or communities). Networks with high modularity have dense connections between the nodes within modules but sparse connections between nodes in different modules.
- *Connected Components* - a connected component (or just component) of an undirected graph is a subgraph in which any two vertices are connected to each other by paths, and which is connected to no additional vertices in the supergraph.

Node Features

- *Clustering Coefficient* **-** A clustering coefficient is a measure of the degree to which nodes in a graph tend to cluster together.
- *Centrality* **-** centrality refers to indicators which identify the most important vertices within a graph. Applications include identifying the most influential person(s) in a social network, key infrastructure nodes in the Internet or urban networks, and super spreaders of disease.
    - Closeness Centrality **-** In connected graphs there is a natural distance metric between all pairs of nodes, defined by the length of their shortest paths. The far ness of a node is defined as the sum of its distances to all other nodes, and its closeness is defined as the reciprocal of the farness. Thus, the more central a node is the lower its total distance to all other nodes.
    - Betweenness Centrality **-** Betweenness is a centrality measure of a vertex within a graph (there is also edge Betweenness, which is not discussed here). Betweenness centrality quantifies the number of times a node acts as a bridge along the shortest path between two other nodes.
    - Eigenvector Centrality **-** Eigenvector centrality is a measure of the influence of a node in a network. It assigns relative scores to all nodes in the network based on the concept that connections to high-scoring nodes contribute more to the score of the node in question than equal connections to low-scoring nodes.

Edge Features

Average Path Length **-** Average path length is defined as the average number of steps along the shortest paths for all possible pairs of network nodes. It is a measure of the efficiency of information or mass transport on a network.

Data Visualization Tools

Network data is intuitive and many people know about it. However, what people do not know is how important it can be in so many areas. Examples include:

Network data is something that the most people know about, but they do not know about the analysis, which is important in many areas. Examples of networks include: Biologists who study the formation of protein interactions of a network, Criminologists and law enforcement agencies analyze crime networks, Epidemiologists study the relationships between individuals, Zoologists study the animal behaviors in the network and researchers of telecommunications analyze contact networks. Social network analysis is an attempt to answer some questions such as: Which entities or people are leaders and which of them are followers? What are the influential elements? Are there any groups and how they are formed? Which elements are important in a group? Which elements are the outliers? Which relationships are important? When these networks are small, the manual analysis will be easy but impossible with large networks, so when the network is large and complex, the social network analysis software can be used [13, 14, 15, 16, 17, 18].

To analyze and mine the desired information from these gigantic Network datasets, graph based mining tools are available on the internet. Each one of the tools has its own benefits and core features. Choosing the right tools for a particular task can not only save time, but can give best results. So, it is really important to know what tools to use depending on the type and size of the dataset, the problem we are trying to solve, the computational resources we have and the time we

can spend.

Since there are so many Data Visualization tools in the market and the time to write this thesis was limited, it was really important for me to choose a handful of tools. For this thesis, I have decided to choose three Network Analysis Tools, namely:

1. Pajek
2. Gephi
3. NodeXL

Gephi

Gephi is an open-source software for network visualization and analysis. It helps data analysts to intuitively reveal patterns and trends, highlight outliers and tells stories with their data. It uses a 3D render engine to display large graphs in real-time and to speed up the exploration. Gephi combines built-in functionalities and flexible architecture to explore and analyze the dataset, spatialize and filter necessary information, manipulate the colors, shapes, structures to reveal the hidden properties [4].

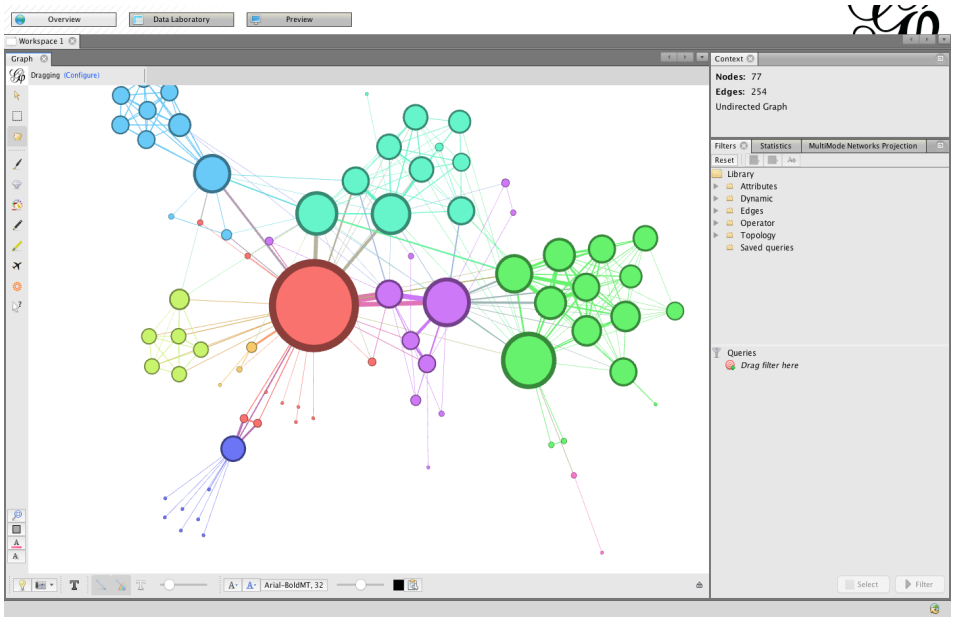


Figure 2: Gephi in action

Gephi has a very intuitive layout. The visualization occurs in the middle and the filters and analysis can be done on the left and right panels. It's easier to see what's happening when calculation and filtering go in parallel.

NodeXL

NodeXL is the MS Paint of Networks. A free, open-source template for Microsoft® Excel® that makes it easy to explore network graphs, NodeXL, can enter a network edge list in a worksheet, where you can easily get accustomed with the environment of Excel [6]. Mostly used for Graph Metric Calculations, Vertex Grouping and Dynamic Filtering, NodeXL has Flexible Import and

Export function, Direct Connection to Social Network, Flexible Layout and Task Automation features, which makes it a very robust and powerful Data Visualization Tool [6].

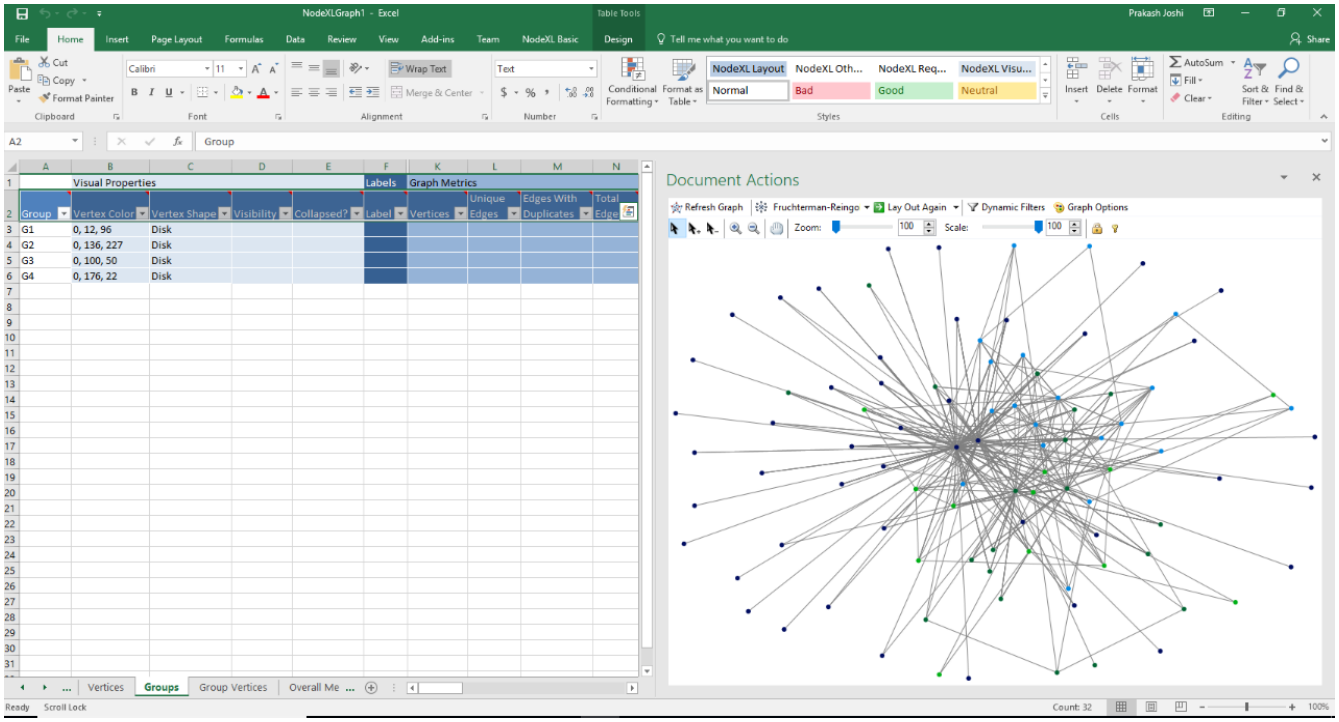


Figure 3: NodeXL in action

Pajek

Pajek is a very popular software for drawing networks and has been a great choice for many to compute block-model, identifying structural holes and compute centrality measures [5].

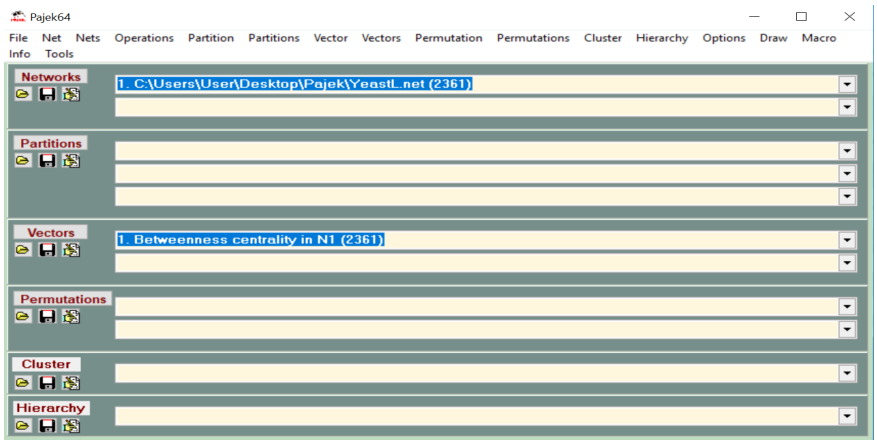


Figure 4: Initial Pajek Welcome Window

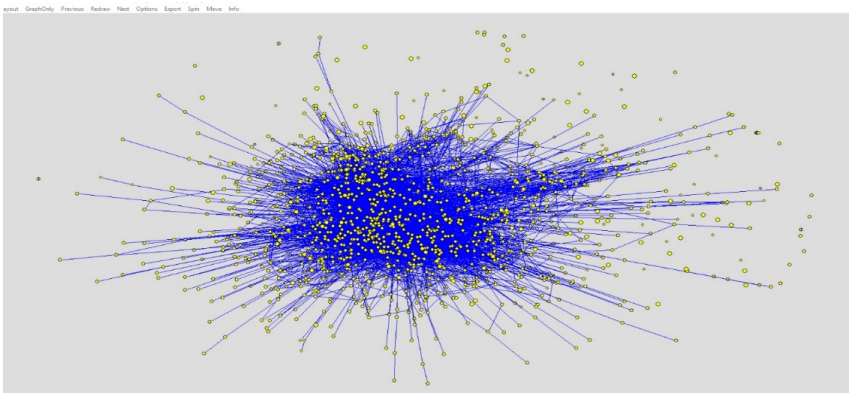


Figure 5: Pajek in Action

One of the things that can be noticed is Pajek is setup differently than most of the network tools. At first, it can be a little confusing to work with, given its welcome screen layout, which is not intuitive when you look at. Unlike Gephi and NodeXL, Pajek's layout looks very dull and unimaginative, given that it's a visualization tool. Nevertheless, Pajek is one of the most powerful Network Analysis tools out there[12].

Basis of Comparison:

Since there are so many things each tool can do, it was really important to choose particular characteristics that I needed to base my comparison on. In this thesis, I have used following matrices to compare the tools. Since this thesis mainly focuses on the Visualization, that was the most important feature to delve into. Following are the bases of comparison I used for this this thesis.

1. Visualization
    a. Graph Layouts
    b. Groups and Cluster Visualization
    c. Overall Visualization Strength and flexibility
2. Graph Features Calculation
3. Scalability
4. Compatibility

Visualization

In the Visualization comparison, it has been divided into following three bases:

1. Graph Layouts
2. Groups and Cluster Visualization
3. Overall Visualization quality

One of the most important aspects of Visualization is Graph Layout. Graph Layout is setting up the shape of the graph. Each Layout option has a defined way of handling edges and vertices in a graph. Layouts can help in readability, usability, symmetricity and other aesthetics. Using the right layout can save time and memory, and can help find the right solutions in minimal time. Therefore, it's much important to know about Layouts in a tool before doing anything else.

In the Visualization analysis of the tools below, I have first shown the data in different Layouts and then tried to show how each tool visualizes data in different approaches.

After discussing Layouts, I have focused on how groups and clusters are visualized in each tool and finally written an analysis on the overall visualization properties.

a. *Gephi*

Gephi is very strong in terms of Graph Layout and overall visualization. There are not only inbuilt tools, but also external plugins that can be downloaded for free just for the Layout

part. For this paper, we are going to discuss few Layouts that are most popular and useful when it comes to Data analysis and Visualization. For the Visualization part, I started with a dataset that was in the rawest form in terms of shape. Eventually, I visualized data emphasizing other graph features.

Dataset1

Dataset Name	Les Misérables
Vertices	77
Edges	254
Edge Weight	Positive Weights
Direction Type	Undirected
Loops	0
Information: This undirected network dataset contains co-occurrences in Victor Hugo's novel 'Les Misérables'. Each vertex represents a character and an edge signifies that these two particular characters occurred on the same page of the book. The weight link indicates how often such co-appearance occurred.	

Table 1: Dataset 1 (Les Misérables)

1. Layouts in Gephi

   Initial Layout immediately after loading the file.

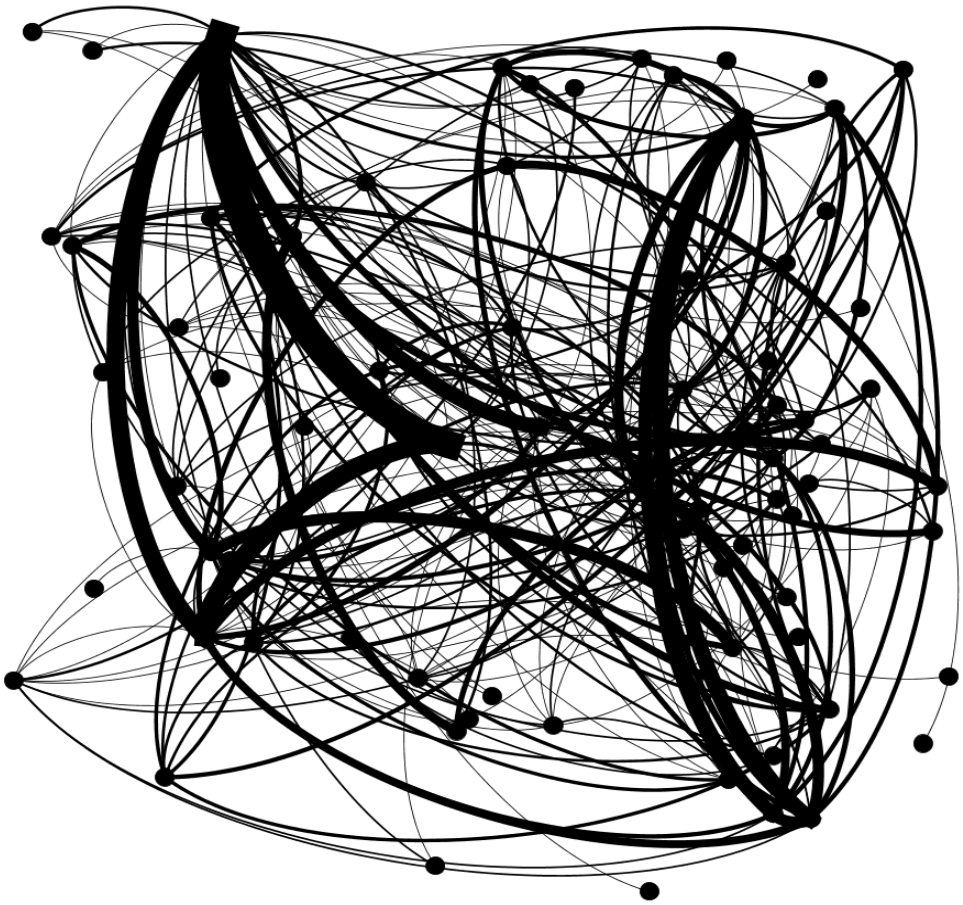


Figure 6: Raw Layout Les Misérables

The visualization was done solely by Gephi itself (without any filters), and we can see that all nodes are of same size. The edges, however have different thickness. We will see changes to all

that once we start applying filters and effects. One of the most popular layouts, Fruchterman Reingold Layout tries to minimize the energy within the system. The nodes connected strongly get closer and the nodes that do not have connection are send farther away. It is slower than other Layouts, however it is a good way to just get an overview of how nodes are behaving in a certain Network data.

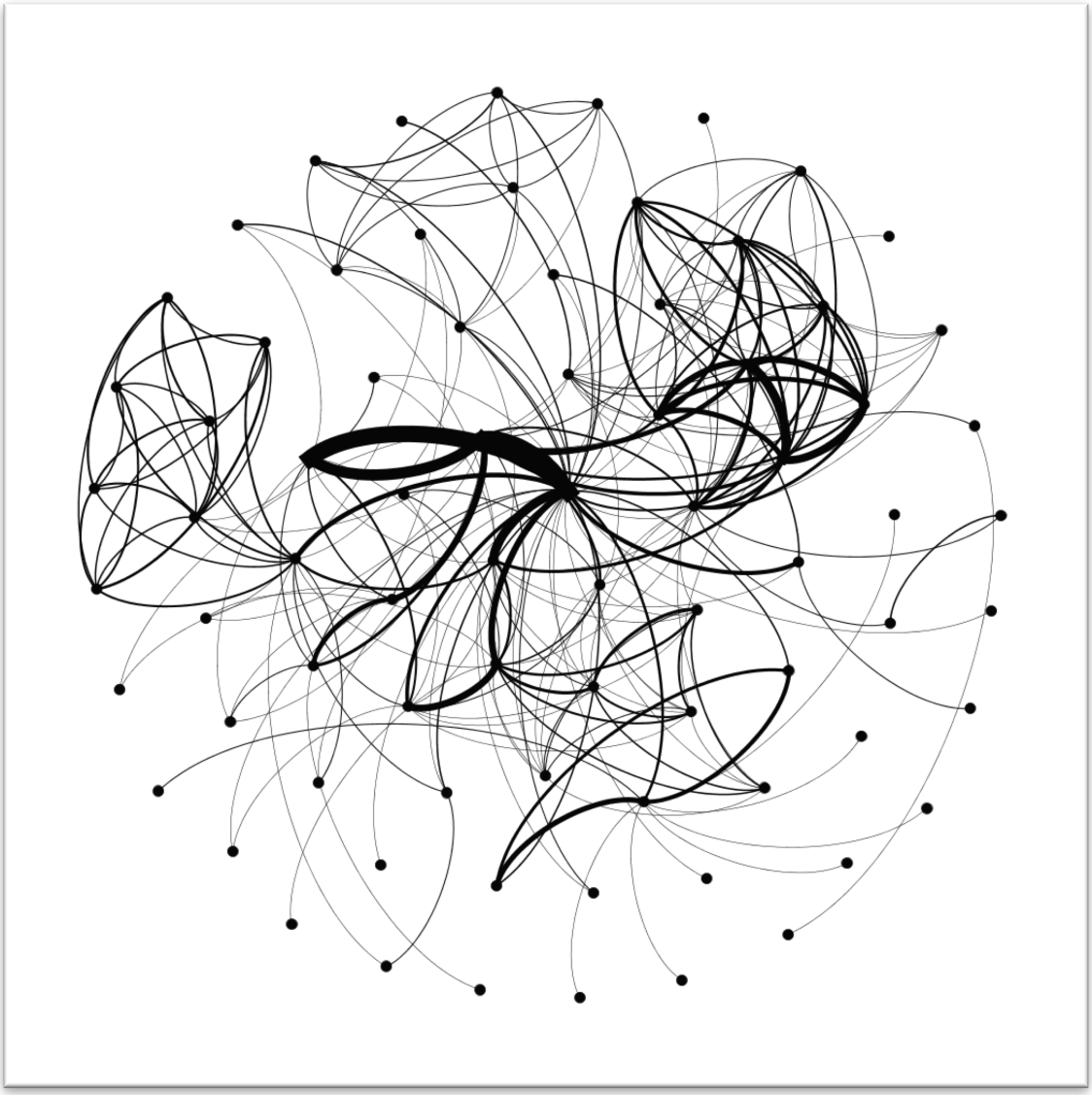


Figure 7: Fruchterman Reingold Les Misérables

Apart from this, Gephi has numerous other Layouts such as Force Atlas, Noverlap, OpenOrd, Random Layout, Yifan Hu, Yifan Hu proportional etc. These Layouts can help you get started with data Visualization with a single click, without much hassle.

2. Groups and Clusters Visualization in Gephi

   As we can see the figure below, the more degree centrality the node has, the bigger it is. The node on the farthest right has the most degree centrality and is the most important node in the Network. It can be also seen that the nodes with same color have almost the same degree centrality. This is a way good way to figure out which nodes are the most

important and which ones are least, depending on what we are trying to look for.

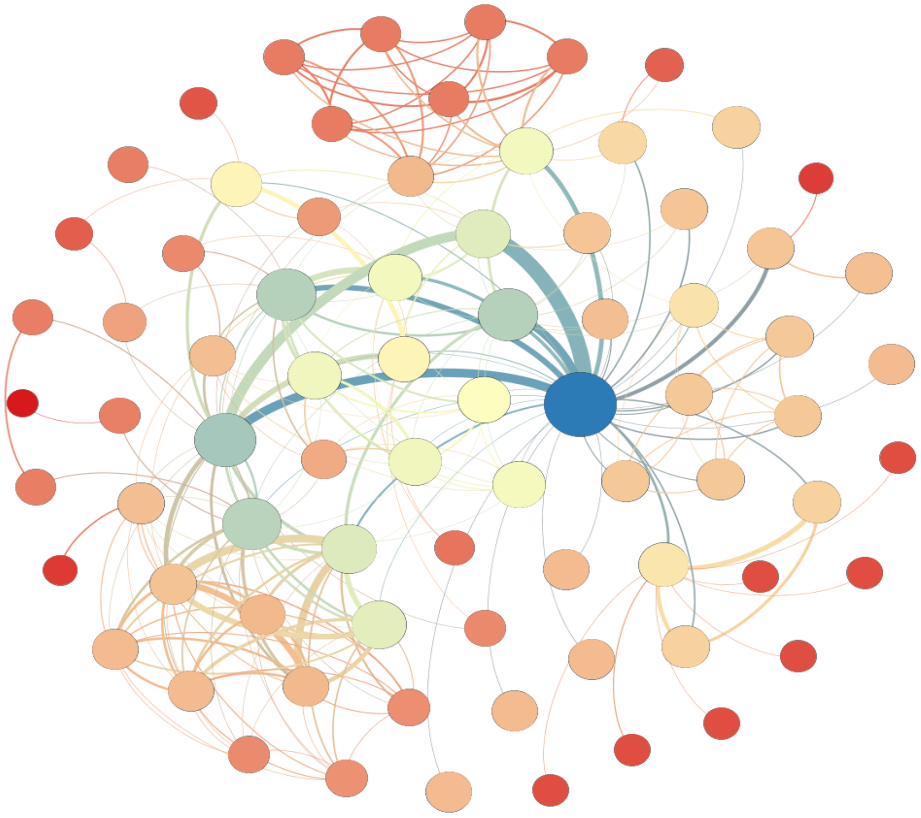


Figure 8: Les Misérables Groups and Clusters

3. Overall Visualization properties of Gephi:

   The overall visualization characteristic of Gephi are quite amazing. We have numerous inbuilt plugins and other several plug ins that can be downloaded externally. If the goal is to visualize data, then choosing Gephi is definitely the way to go.

b. NodeXL

NodeXL seems almost as strong as Gephi in terms of Layouts, because it also has almost the same options as Gephi. However, there is another advantage of using NodeXL – you can extract data based on a keyword (or hashtag) from Facebook, Twitter and other social networking sites. This saves a lot of time, because data collection and filtering becomes no longer a problem

Although mining data on NodeXL from these social networking websites is easy, you can only extract 2000 vertices, which can be a significant disadvantage, since we are dealing with big data here, 2000 vertices definitely seem negligent. Another thing that sets behind NodeXL from Gephi and Pajek is that since it is an extension to the Windows Excel tool, it does have a few limitations of how much data it can handle, because Excel itself isn't the best tool when it comes to handling big data. Therefore, there are limitations to NodeXL when it comes to size.

To show Visualization characteristics and capabilities of NodeXL, I have used the inbuilt tool "Import from Twitter Search Network".

Dataset 2 Name	Twitter query "mardigras"
Vertices	1019
Edges	2141

Edge Weight	Positive Weights
Direction Type	Undirected
Loops	394
Information: Vertices are unique Twitter users who either tweeted "mardigras", or were replied to in of those tweets, or were mentioned in of those tweets. If a tweet was a reply to someone else, NodeXL creates an edge from tweeter to the replied-to user and gives it a "Relationship" value of "Replies-to". There can be only one such Replies-to for each tweet. If a tweet mentioned someone else, NodeXL creates an edge from the tweeter to the mentioned user and gives it a "Relationship" value of "Mentions". There can be multiple Mentions edges for each tweet. (Note that a "Replies-to" is NOT also a "Mentions.") If a tweet neither replied to nor mentioned anyone else, NodeXL creates a self-loop edge from the tweeter to herself and gives it a "Relationship" value of "Tweet."	

Table 2: Dataset 2 (People who tweeted "mardi gras")

1. Layouts in NodeXL

   For the following visualization, I used Fruchterman-Reingold Layout. This layout tries to minimize the energy in the system (basically making edges non-overlap). Apart from this, we can also use other Layouts such as Harel-Koren Fast Multiscale (brings most highly connected nodes together and send least highly connected nodes further), Circle, Spiral, Horizontal Sine Wave, Vertical Sine Wave, Grid, Polar, Polar Absolute, Sugiyama, and Random.

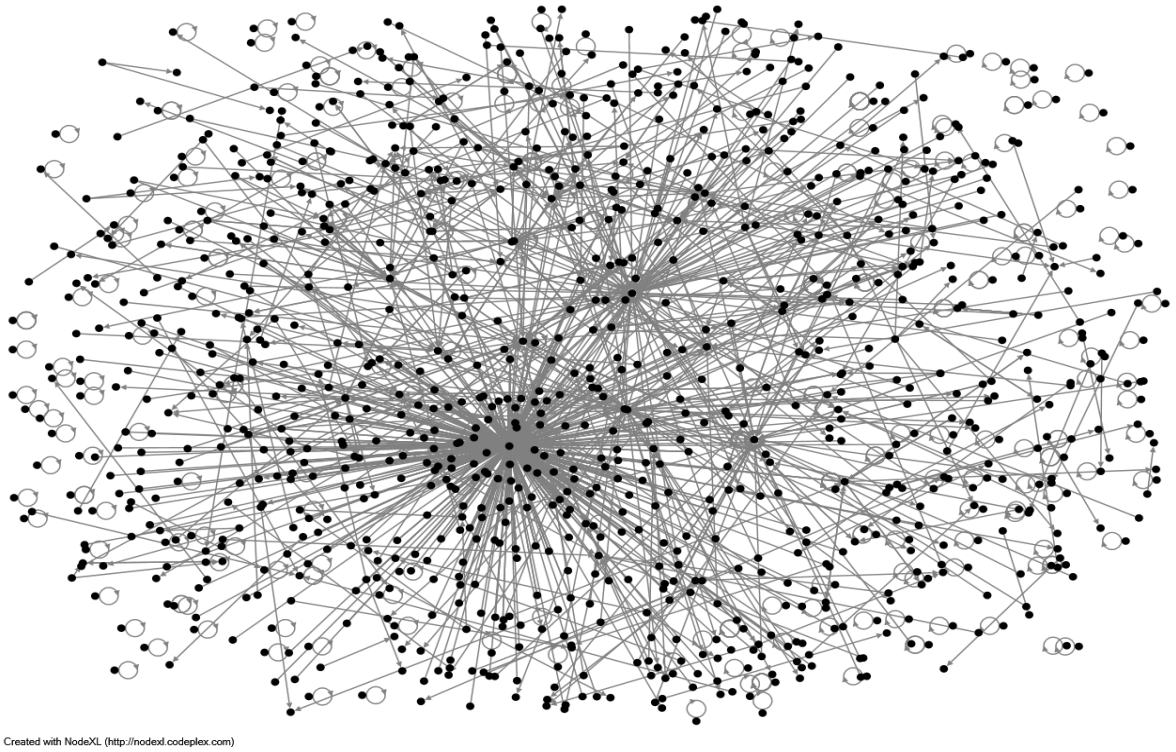


Figure 9: NodeXL Fruchterman Reingold

2. Visualizing Groups and Clusters in NodeXL

Visualization based on grouping and clustering in NodeXL are quite robust. We can group by Vertex Attribute (Node degree, Betweenness Centrality etc.), Connected Component (connected set of vertices), Cluster and Motifs. Following are few examples from the same dataset.

Grouped on the basis of degree, the following graph highlights the vertices with in degree between 50-100. (For this network, maximum in degree = 224, and lowest = 0)

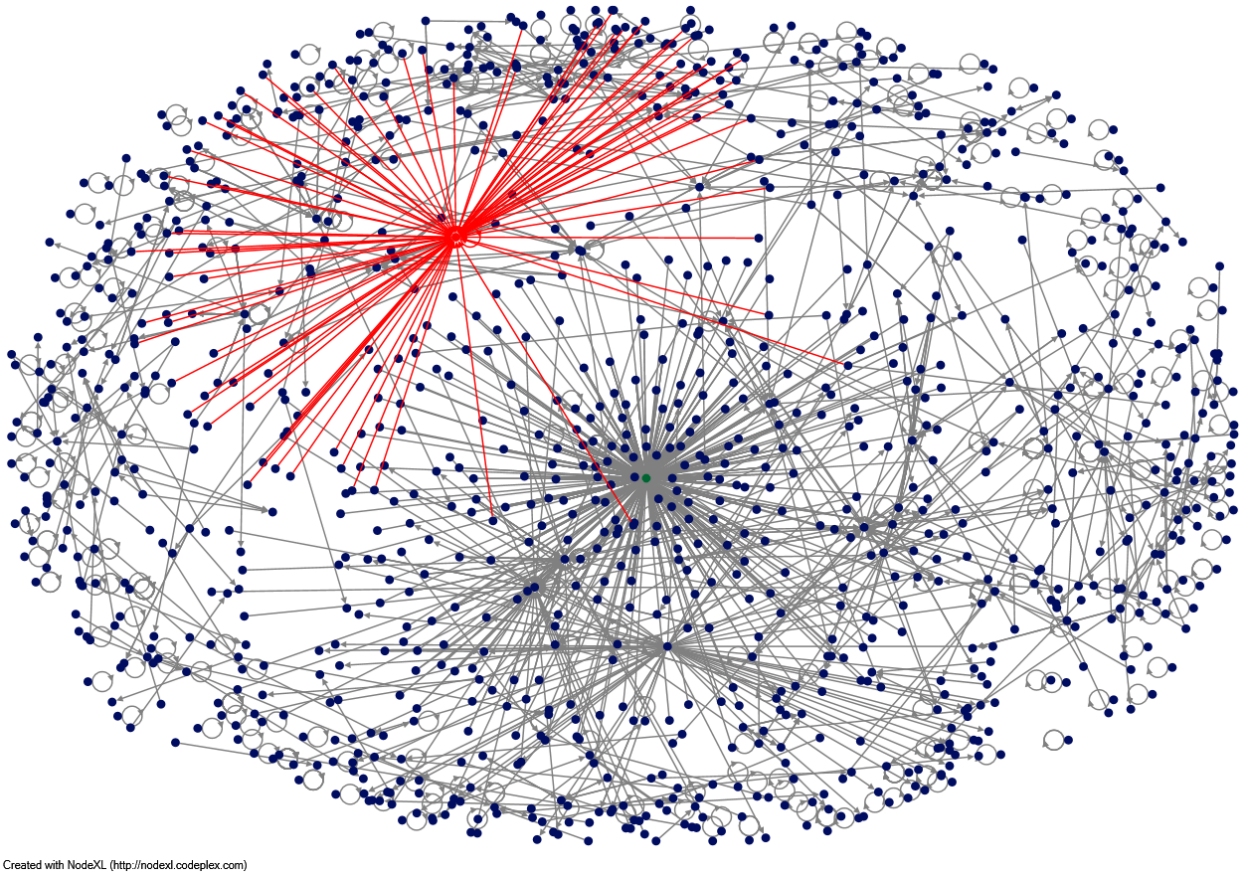


Figure 10 NodeXL Clusters and Groups

In the following visualization, the vertices are first grouped by Connected Component and each group is presented in a box. With this visualization, it is much easier to see data as groups and definitely even easier to analyze it.

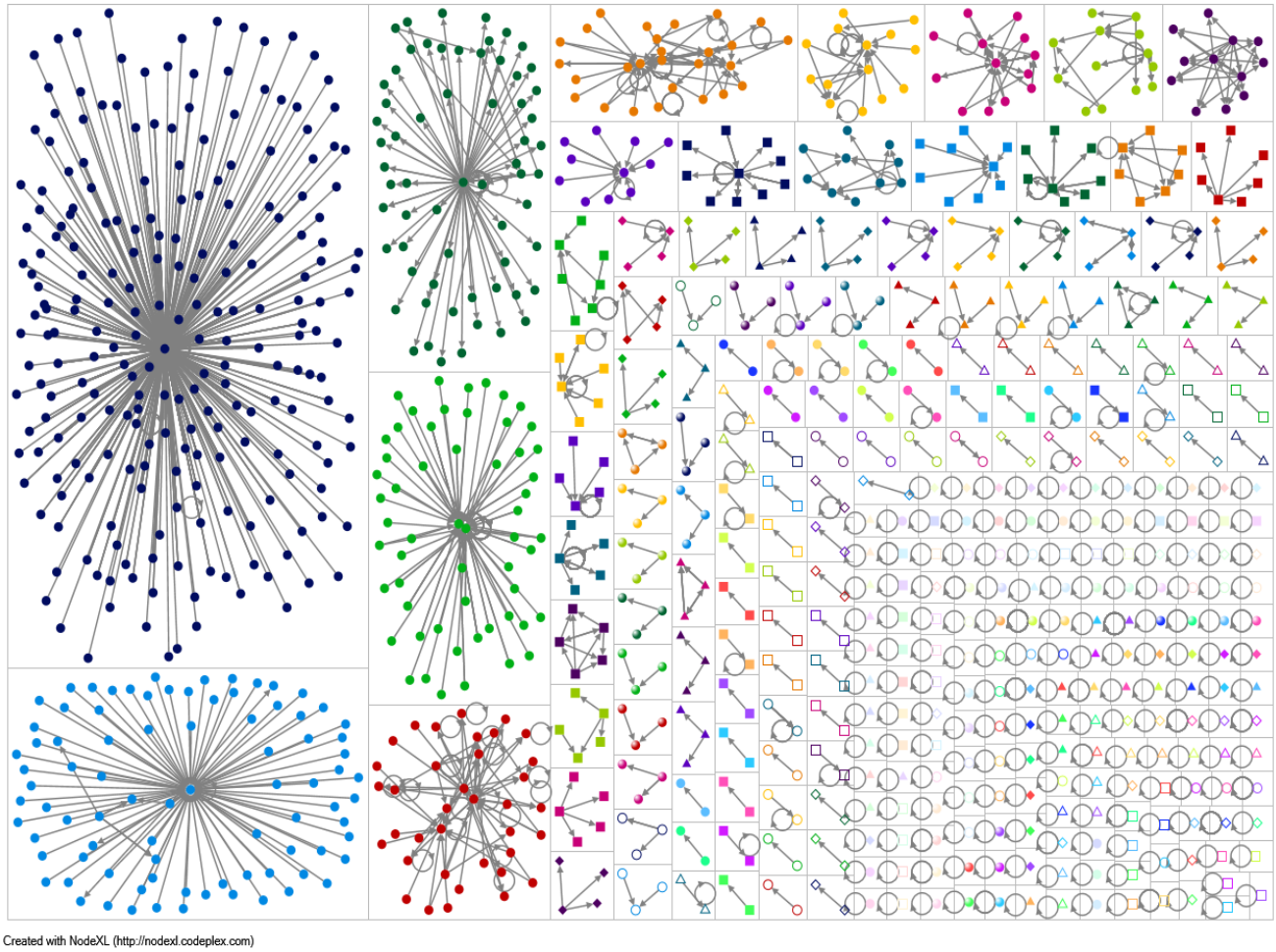


Figure 11: NodeXL groups in a box

The clusters in the picture are based on nodes with similar range of in degree. The particular one selected in red had the range in degree of 50- 100.

3. Other features and Overall Visualization

In compared to Pajek, the overall aesthetics of the visualization is great with NodeXL. However, it does lack that extra sharpness and stunning images that Gephi can produce. NodeXL is paint, while Gephi is Photoshop for network visualization [8, 10].

c. Pajek

Pajek is a little weaker in terms of Layout flexibility. We don't have a lot of options and that overall aesthetic quality doesn't look as appealing as Gephi and Pajek. That being said, it definitely isn't your go to tool when it comes to making good looking visualizations. However, Pajek is very robust and comes as a winner when it has to handle big amount of data. If the computer has enough amount of RAM and processing power, Pajek can handle more than a million edges and can be used to handle most of the big data. Following are the features that Pajek can do when it comes to Visualization.

1. Layouts in NodeXL

The layout features that Pajek has are Circular (Original, using Permutation and Random), Kamada Kawai and Fruchterman Reingold. The latter two try to minimize the energy in the system. As it can be seen that Pajek is not particularly as rich as NodeXL and Gephi in terms of Layouts. Following visualization is based on Fruchterman Reingold Layout.

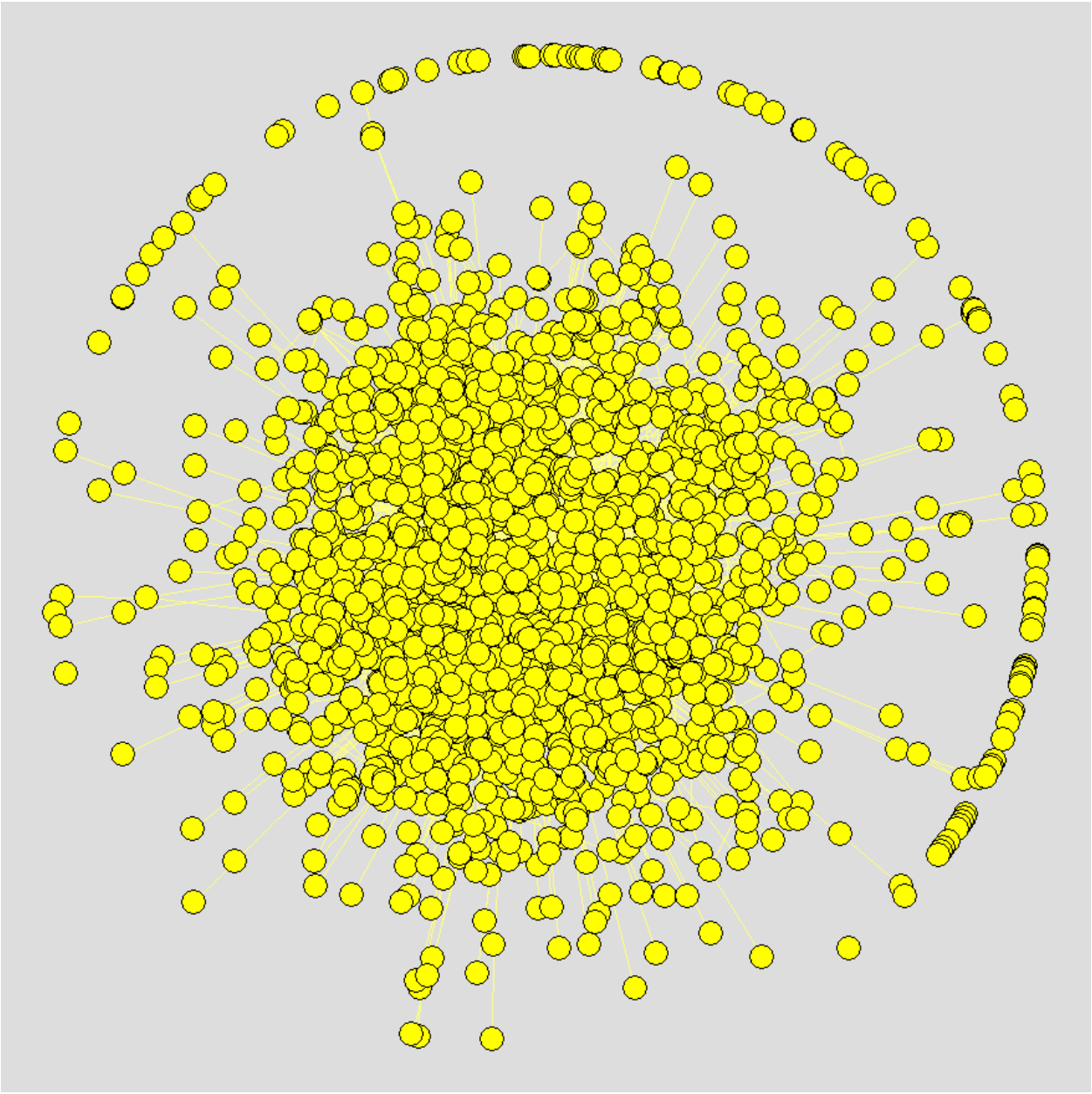


Figure 12: Fruchterman Reingold Pajek

Although Fruchterman Reingold minimizes the energy in the system and tried to un overlap the nodes, this looks rather crowded for this particular layout. The image isn't that sharp and default color selective doesn't look creative.

2. Visualizing Groups and Clusters in Pajek

   Pajek computes absolute values for degree centrality only; for other centrality measures, it performs relative centrality. In the following visualization, I used Fruchterman Reingold Layout. and set the size of the nodes according to their Betweenness Centrality.

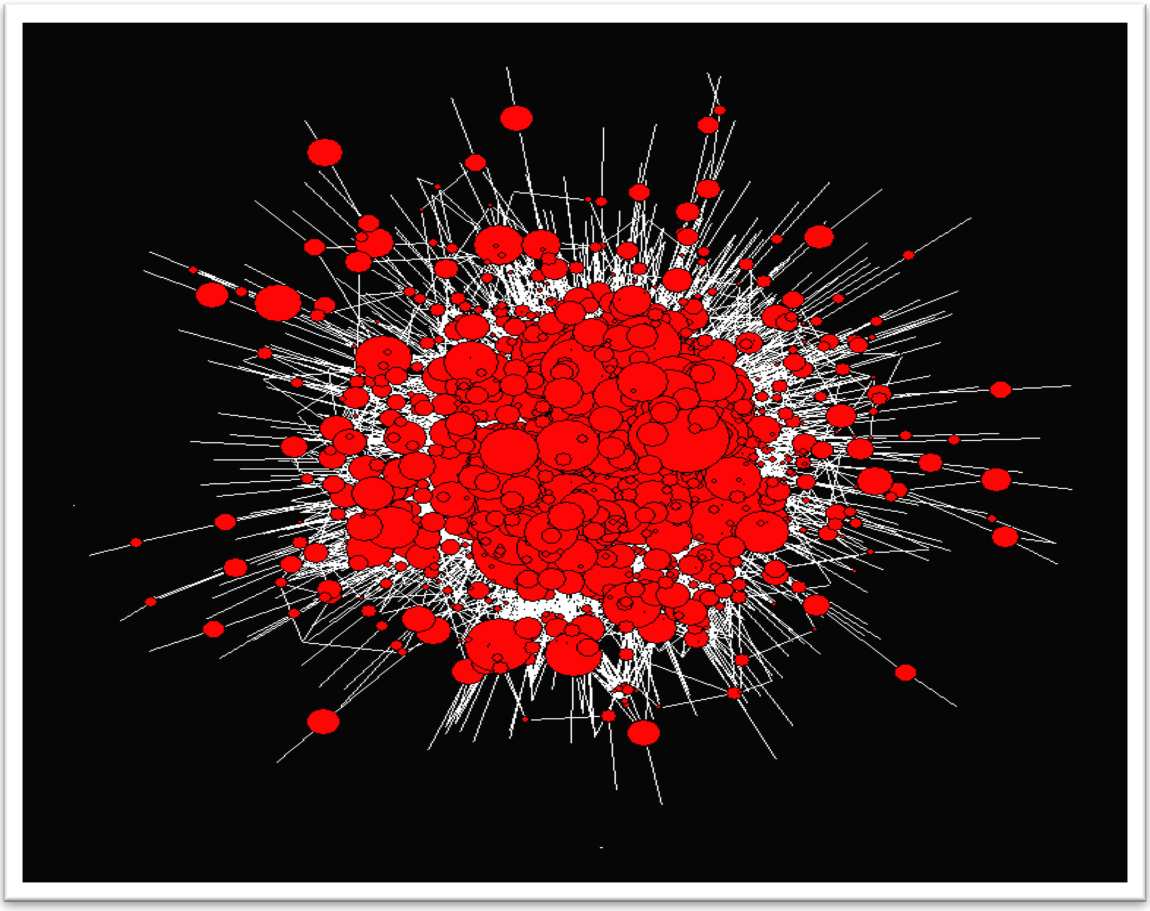


Figure 13: Clusters and groups in Pajek

The nodes that are the biggest have the most Betweenness Centrality and the ones that are smallest have the least.

3. Other features and overall Visualization

Since Pajek itself it a little unintuitive to work with; its visual properties are hard to set. The overall visualization of a graph can be made better, but it could be a lot of work, since all the things that need to be done need to be done through drop down properties. Unlike Gephi and NodeXL, where there are sliders and filtering data is much easy.

Graph Analysis Features

Apart from Visualization, the other most important characteristics of a tool is Data Analysis. I used tools to calculate all the major Network Properties and wrote my experience using them.

Dataset3

Dataset Name	Yeast
Vertices	2361
Edges	7182
Direction Type	Undirected
Loops	536
Information: Protein-Protein Interaction network in budding yeast. Nodes are proteins and Edges shows the interaction relationship	

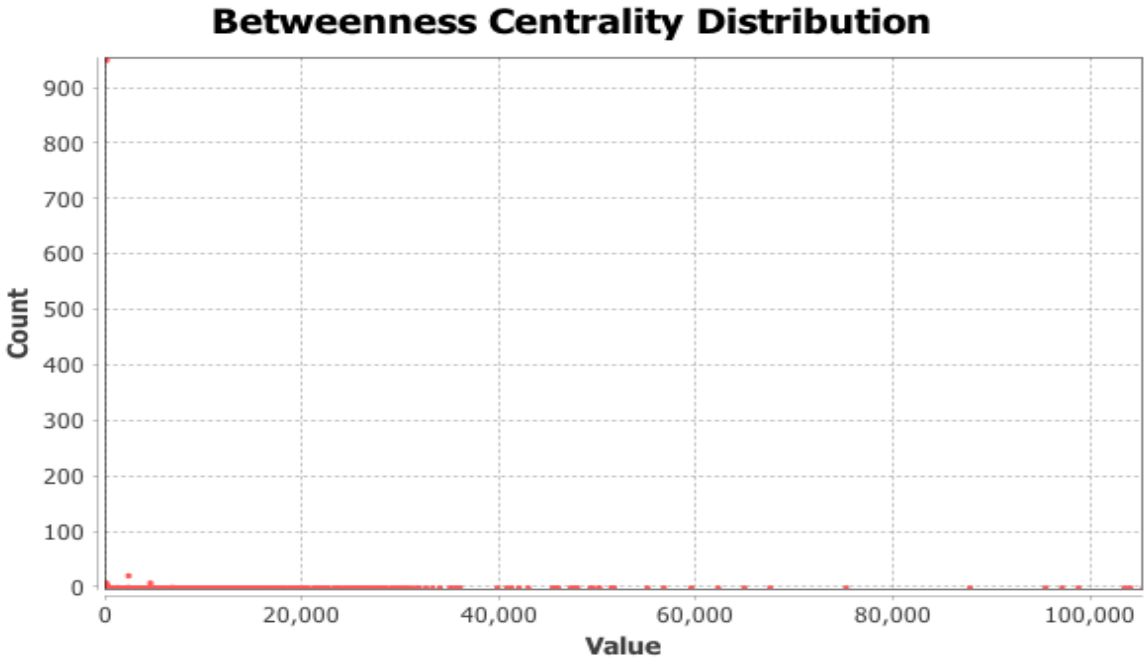
Table 3: Dataset 3 Yeast PPI interaction

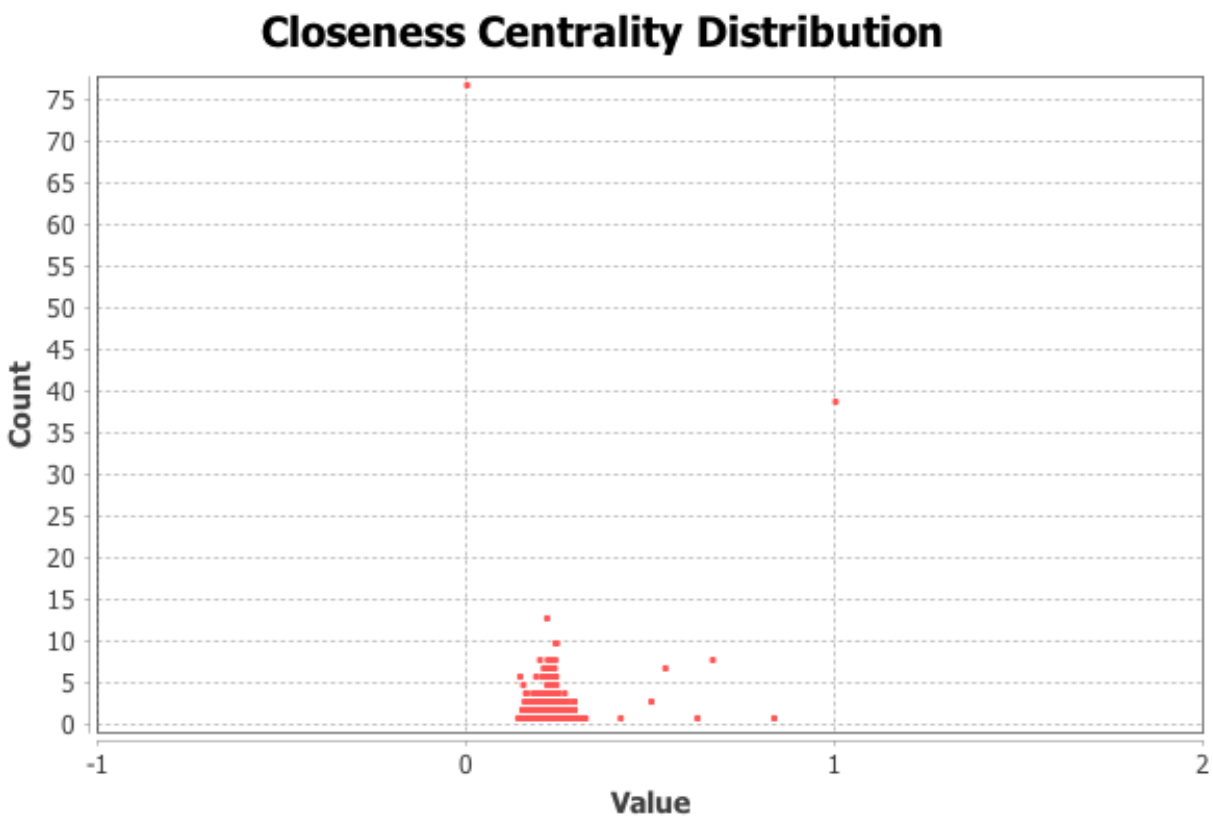
Gephi: All the network characteristics are one click away in Gephi. Not to forget the fact that every time Gephi analyses a data, it produces a very informative graph with it. Following were the statistics calculated with Gephi in the above Data.

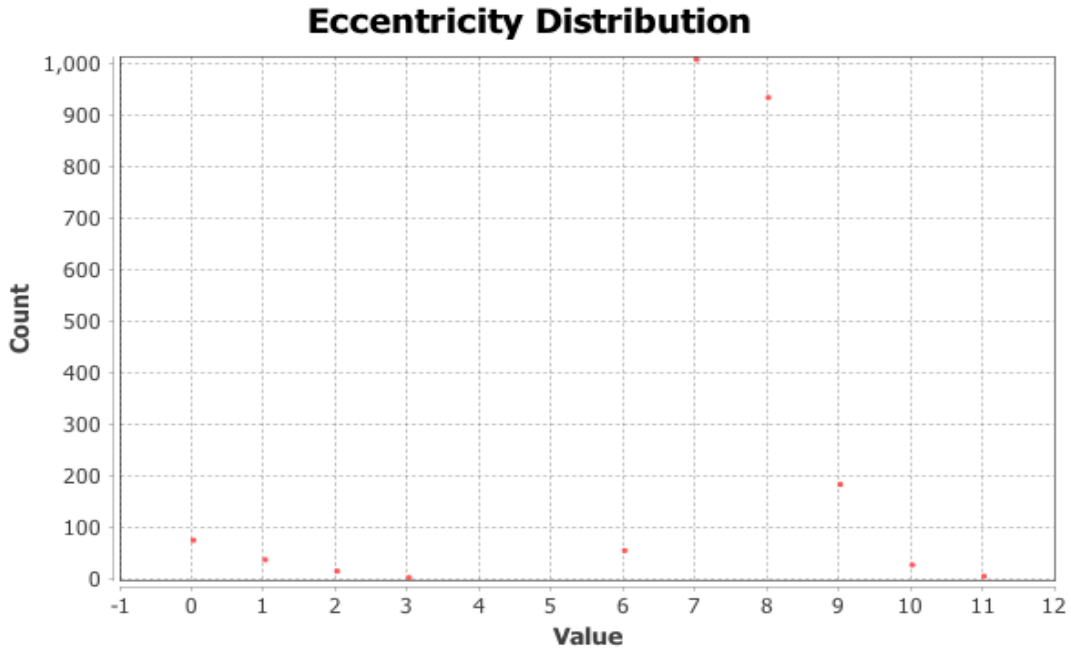
Average Degree	6.084
Avg. Weighted Degree	6.084
Network Diameter	11
Graph Density	0.003
Modularity	0.59
Connected Components	101
Average Clustering Coefficient	0.271

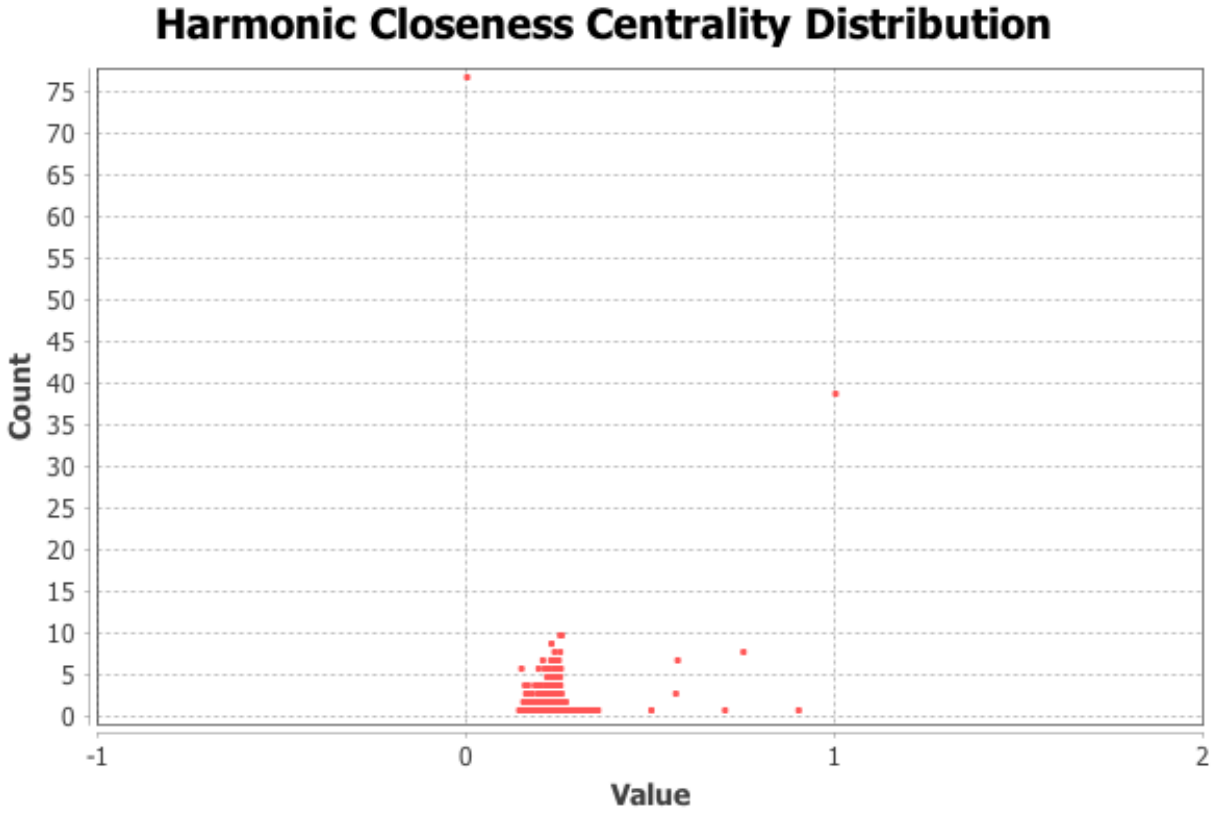
Table 4: Yeast Graph features calculated- by Gephi

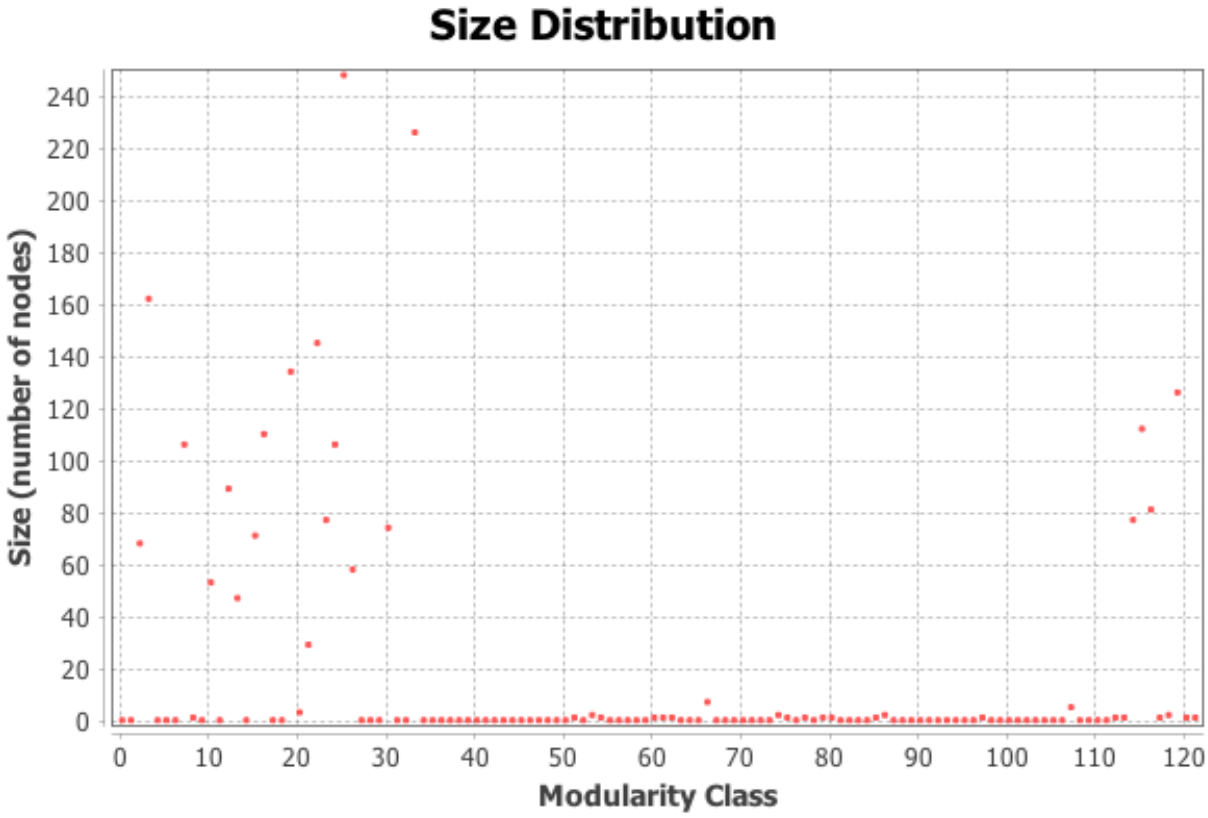
In addition to calculating the features, Gephi also automatically produces graphs of the calculations, as shown below:

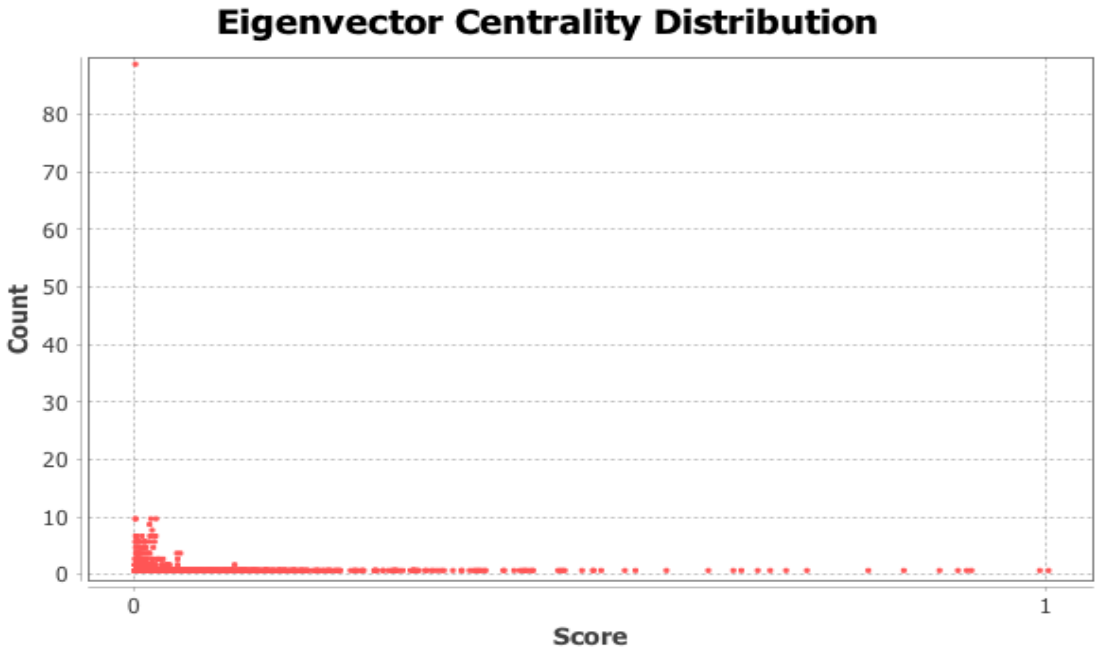

Betweenness Centrality Distribution
Count
900
800
700
600
500
400
300
200
100
0
0
20,000
40,000
60,000
80,000
100,000
Value


Closeness Centrality Distribution
Count
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5
0
-1
0
1
2
Value


Eccentricity Distribution
Count
1,000
900
800
700
600
500
400
300
200
100
0
-1
0
1
2
3
4
5
6
7
8
9
10
11
12
Value


Harmonic Closeness Centrality Distribution
Count
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5
0
-1
0
1
2
Value


Size Distribution
Size (number of nodes)
240
220
200
180
160
140
120
100
80
60
40
20
0
0
10
20
30
40
50
60
70
80
90
100
110
120
Modularity Class


Eigenvector Centrality Distribution
Count
80
70
60
50
40
30
20
10
0
0
1
Score

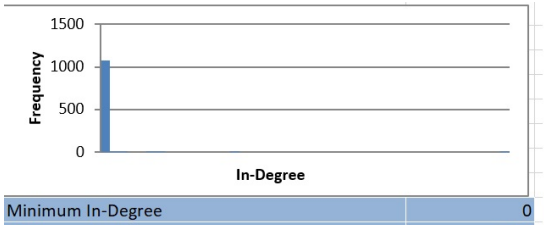
NodeXL: Like Gephi, NodeXL can compute all the major Graph features with one click.

Following were the results, when the same dataset 2 was used for analysis.

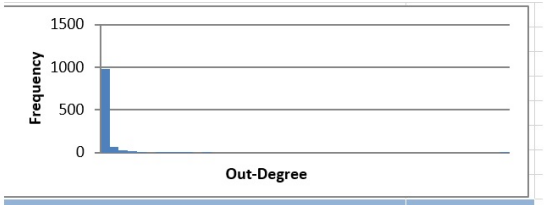
Average Degree	1.052
Avg. Weighted Degree	1.052
Network Diameter	6
Graph Density	0.000758
Modularity	0.4516
Connected Components	306
Average Clustering Coefficient	0.036

Table 5: Twitter data graph feature calculated by NodeXl

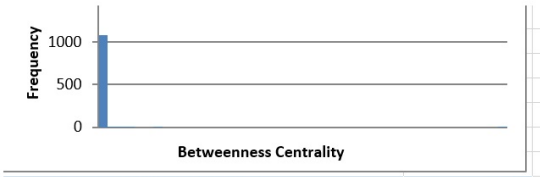
NodeXL creates graphs and analysis reports as well, as shown below:



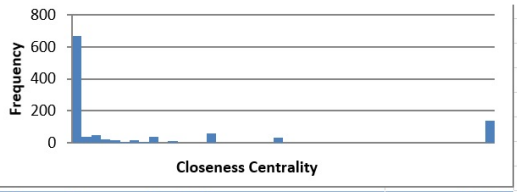
Minimum In-Degree	0
Maximum In-Degree	224
Average In-Degree	1.052
Median In-Degree	0.000



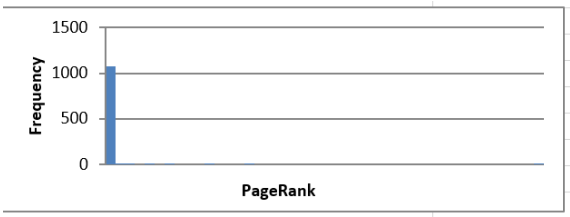
Minimum Out-Degree	0
Maximum Out-Degree	56
Average Out-Degree	1.052
Median Out-Degree	1.000



Minimum Betweenness Centrality	0.000
Maximum Betweenness Centrality	49506.000
Average Betweenness Centrality	65.799
Median Betweenness Centrality	0.000

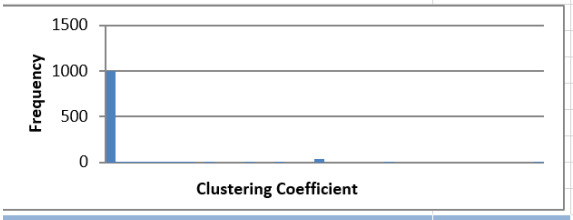


Minimum Closeness Centrality	0.000
Maximum Closeness Centrality	1.000
Average Closeness Centrality	0.181
Median Closeness Centrality	0.008



Minimum PageRank	0.358
Maximum PageRank	102.779
Average PageRank	1.000
Median PageRank	0.655

Minimum PageRank	0.358
Maximum PageRank	102.779
Average PageRank	1.000
Median PageRank	0.655



Minimum Clustering Coefficient	0.000
Maximum Clustering Coefficient	1.000
Average Clustering Coefficient	0.036
Median Clustering Coefficient	0.000

Minimum Clustering Coefficient	0.000
Maximum Clustering Coefficient	1.000
Average Clustering Coefficient	0.036
Median Clustering Coefficient	0.000

As we can see that NodeXL not only calculates, but also creates graphs for us. Hence Data Analysis with is much simpler and easier with NodeXL, because everything gets calculated in a single click.

Pajek: With Pajek, it's a little harder to perform Data Analysis, but most of the things can be calculated with it. As opposed to Gephi and NodeXL, where all the calculations are one click away, with Pajek you need to calculate each metric individually. This can be rather frustrating if you have a lot of metrics to calculate. However, Pajek can easily handle more than a million vertices (given that you have enough computing resources). Therefore, Pajek should be the choice when it comes to analyzing and visualizing big data.

As we can see below, if we need to get all the data metrics, we need to get all the calculations separately. This can be very time consuming. In the following figures, I have shown how data is represented in Pajek (Betweenness Centralization, Clustering Coefficients and Diameter)

```
 Working...
  8.47% finished. Time spent: 0:00:00
 16.94% finished. Time spent: 0:00:00
 25.41% finished. Time spent: 0:00:00
 33.88% finished. Time spent: 0:00:00
 42.35% finished. Time spent: 0:00:00
 50.83% finished. Time spent: 0:00:00
 59.30% finished. Time spent: 0:00:00
 67.77% finished. Time spent: 0:00:00
 76.24% finished. Time spent: 0:00:00
 84.71% finished. Time spent: 0:00:00
 93.18% finished. Time spent: 0:00:00
------------------------------------------------
Network Betweenness Centralization = 0.03652700
------------------------------------------------
```

Figure 14: Betweenness Centrality calculated by Pajek

```
Computing Clustering Coefficients
------------------------------------------------------------------------------
 Working...
 Watts-Strogatz Clustering Coefficient: 0.20013457
 Network Clustering Coefficient (Transitivity): 0.10231489
 Time spent:  0:00:00
```

Figure 15 Clustering Coefficient Calculated by Pajek

```
------------------------------------------------------------------------------
Searching the longest shortest path in 1. C:\Users\User\Desktop\Pajek\YeastL.net (2361)
------------------------------------------------------------------------------
 Working...
42% finished. From AME1 actin related protein(604) to RAM2 protein farnesyltransferase, alpha subunit(1049) is distance 11.
 Time spent:  0:00:00
85% finished. From AME1 actin related protein(604) to RAM2 protein farnesyltransferase, alpha subunit(1049) is distance 11.
 Time spent:  0:00:00

          Result:
 The longest shortest path from AME1 actin related protein (604) to RAM2 protein farnesyltransferase, alpha subunit (1049). Diameter is 11.
 Time spent:  0:00:00
 Time spent:  0:00:00
```

Figure 16: Diameter calculated by NodeXL

Compatibility

When we are talking about software, compatibility is a significant factor to consider. If it works in one system and doesn't work in another, there could be problems. In the following comparison, I have shown compatibility of the tools based on Operating Systems and File Format (Input and Output).

Based on Operating Systems

Operating System	Gephi	Pajek	NodeXL
Windows	√	√	√
Mac	√	√ (Using Wine stable)	X
Linux	√	√(Using Wine stable)	X

Table 6: Compatibility based on Operating System

As we can see, Gephi is the only one that is compatible with all the Operating System. This is a big plus factor for Gephi. Pajek is built for Windows Operating System; however, using a software called Wine stable, it can be run on Mac OS and Linux. On the other hand, NodeXL is only compatible on Windows Operating System. This can be a significant problem for someone who doesn't have access to Windows Operating System. The only way Mac/Linux users can use

NodeXL is by installing a copy of Windows Operating System in a Virtual Box, which can be a hassle in terms of memory use, and extra time and money spent in just setting up the environment alone.

Data Types Supported

Input data types

Gephi	Pajek	NodeXL
GraphViz(.dot), Graphlet(.gml), GUESS(.gdf), LEDA(.gml), NetworkX(.graphml, .net), NodeXL(.graphml, .net), Pajek(.net, .gml), Sonivis(.graphml), Tulip(.tlp, .dot), UCINET(.dl), yEd(.gml), Gephi (.gexf), Edge list(.csv), databases	Convert text file into excel, The format pajek ، UCINet(dl) GED, Ore-graph, p- graph,	GraphML, Pajek, UCINet, and matrix formats.

Table 7: Input Data types supported by Tools

Output Data Types

Gephi	Pajek	NodeXL
CSV, SVG, GDF GEXF, GraphML Pajek NET, Spreadsheet	.SVG, .SVG.GZ .HTML files	GraphML, Pajek, UCINet, and matrix formats

Table 8: Output Data types supported by Tools

Gephi is most flexible when it comes to data types and that's what makes it very flexible to use. Generally, what happens is even though you have a dataset, it's not in the right format, and if it contains thousands of vertices, manually converting to the right format can take days. With Pajek and NodeXL, that's usually what happens unfortunately. It's rare that you will find a dataset that's in the native form. But since Gephi is compatible with so many datasets, it's much easier to get started with Gephi. Thus, Gephi is a true winner when it comes to Compatibility.

Scalability:

With data science, the main concern is always the data size. It's a very crucial aspect of a tool to be able to handle data with thousands of vertices and edges. Following are the details about how much each tool can handle.

Gephi: The maximum number of nodes and edges that Gephi can handle is 100k and 1000k respectively [7]. Although Gephi can create stunning images with its Visualization tools, it isn't the tool one should look for when the data is really big. If the data has more than a million vertices, using Gephi is out of the equation.

NodeXL The maximum number of edges NodeXL can handle is 200k. [8]. Therefore, NodeXL isn't an ideal tool when it comes to analysis and visualization of very large datasets. There is a feature where one can slice up the data into parts and analyze it, but it can be time consuming and there is a limit to that as well.

Pajek If there is enough RAM (~ 152 GB), Pajek can handle more than 10 billion vertices [12]. This is where Pajek wins over NodeXL and Gephi. Not a great tool for visualization, but Pajek definitely can handle really enormous datasets. And this is the main reason why Pajek still thrives in the market of Data Analysis.

When it comes to scalability, Pajek is the true winner.

Conclusion (End Report)

To conclude, we can say that when visualization is more important than Analysis, and the data size is small – Gephi is the way to go. With its so many options for Layouts to start with, and then the way it produces sharp and stunning visualization with the minimal amount of work, Gephi is the true winner.

If our analysis and visualization is based on Social Networks and the data size is small. NodeXL lets us readily important data from Twitter, Facebook and other networking websites. This saves us a lot of time in data collection and filtering, and help us get started with Visualization and Analysis without other hassles. NodeXL is a winner when it comes to collecting data from the Social Networking sites.

If the data size is gigantic and the main focus is more on Analyzing than Aisualizing, then Pajek is the way to go. If we have enough RAM, Pajek can handle more than 10 billion sets of nodes, which neither NodeXL or Gephi can do. Therefore, when it comes to Big Data Analysis, Pajek is the true winner. One disadvantage could be getting data in the right form. If the data is not in the right form and too big, it might take a lot of time just to get it into right file format, which is often why Pajek gets criticized.

[References]

[1] L. Rizzati, “Digital Data Storage is Undergoing Mind-Boggling Growth,” *EEtimes*, 14-Sep-2016. [Online]. Available: Eetimes.com. [Accessed: 20-Aug-2017].

[2] Information Visualization Research Group, “AT&T Researchers - Inventing the Science Behind the Service,” *AT&T Labs Research - Getting and Understanding the Bigger Picture*, 11-Oct-2009. [Online]. Available: http://web2.research.att.com/articles/featured_stories/2009/200910_more_than_just_a_picture.html?fbid=AmrsYFWD1Li. [Accessed: 01-Jan-2018]

[3] B. Murr, “Big Data: 20 Mind-Boggling Facts Everyone Must Read,” *Forbes*, 30-Sep-2014. [Online]. Available: [3] https://www.forbes.com/sites/bernardmarr/2015/09/30/big-data-20-mind-boggling-facts-everyone-must-read/#19e0f90217b1. [Accessed: 02-Feb-2018].

[4] “About Gephi,” *Graph exploration and manipulation*. [Online]. Available: https://gephi.org/about/. [Accessed: 12-Jan-2018].

[5] “Networks / Pajek ,” *Networks / Pajek*. [Online]. Available: http://vlado.fmf.uni-lj.si/pub/networks/Pajek/. [Accessed: 12-Jan-2018].

[6] “Keyword Analysis & Research: Nodexl,” *Invisible hit counter*. [Online]. Available: http://www.au-e.com/search/nodexl. [Accessed: 22-Feb-2018].

[7] “Gephi requirements,” *Graph exploration and manipulation*. [Online]. Available: https://gephi.org/users/requirements/. [Accessed: 24-Jan-2018].

[8] “NodeXL Frequently Asked Questions › Social Media Research Foundation,” *NodeXL FAQ*. [Online]. Available: https://www.smrfoundation.org/nodexl/faq/. [Accessed: 23-Feb-2018].

[9] C. Tozzi, “Hard Disk Storage is Cheaper than Ever: Why Do You Still Use Tape?,” *Syncsort & Trillium Software Blog*, 09-May-2017. [Online]. Available: http://blog.syncsort.com/2017/05/big-data/hard-disk-storage-cheap-tape/. [Accessed: 03-Mar-2018].

[10] “Tutorial Gephi Tutorial Layouts,” *Gephi Tutorials*. [Online]. Available: [10] https://gephi.org/tutorials/gephi-tutorial-layouts.pdf. [Accessed: 04-Mar-2018].

[11] “Datasets,” *Datasets*. [Online]. Available: http://moreno.ss.uci.edu/data.html. [Accessed: 07-Mar-2018].

[12] A. Mrvar, “How to use Pajek-XXL and Pajek-3XL,” *Program Package: Pajek / Pajek-XXL*. [Online]. Available: http://mrvar.fdv.uni-lj.si/pajek/PajekXXL.htm. [Accessed: 12-Mar-2018].

[13] Heidler, R., Gamper, M., Herz, A., Eßer, F. (2014): Relationship patterns in the 19th century: The friendship network in a German boys' school class from 1880 to 1881 revisited. Social Networks 13: 1--13.

[14] Arifuzzaman, S., Khan, M. and Marathe, M. (2015). A Space-efficient Parallel Algorithm for Counting Exact Triangles in Massive Networks. In: 17th IEEE International Conference on High Performance Computing and Communications (HPCC). IEEE, pp.527-534.

[15] Arifuzzaman, S., Khan, M. and Marathe, M. (2015). A Fast Parallel Algorithm for Counting Triangles in Graphs using Dynamic Load Balancing. In: IEEE International Conference on Big Data (BigData). IEEE, pp.1839–1847.

[16] Abdelhamid, S., Alam, M., Alo, R., Arifuzzaman, S. (2014). CINET 2.0: A Cyber-Infrastructure for Network Science. In: 10th IEEE International Conference on eScience (eScience), pp 324-331.

[17] Arifuzzaman, S. (2016). Parallel Mining and Analysis of Triangles and Communities in Big Networks. PhD. Dept. of Computer Science, Virginia Tech.

[18] Arifuzzaman, S., Khan, M. and Marathe, M. (2013). PATRIC: a parallel algorithm for counting triangles in massive networks. In: 22nd ACM International Conference on Information & Knowledge Management (CIKM), pp.529-538.