**Social Network Analysis Using Gephi**

**Requirements**

Download Gephi: <https://gephi.org/users/download/>

Data Input:

* Nodes.csv:
  + Column 1: Nodes (list of persons in a network)
  + Column 2: Node Type (source and target)
* Edges.csv: Each row represents a single connection between node in the network.
  + Column 1: Source
  + Column 2: Target

**Step 1: Load the data into Gephi under “New Project”**

* File -> Import…
* Choose ‘nodes.xlsx’
* Import Parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| **Separator:** Comma | **Charset:**  UTF-8 | **Node:** String | **Graph Type:**  Directed or Undirected |
| **Import as:** Nodes Table | **Time Rep:** Intervals | **Type:** String | **Edge Merge Strategy:**  Sum |

* File -> Import Spreadsheet
* Choose ‘edges.xlsx’
* Import Parameters:

|  |  |  |
| --- | --- | --- |
| **Separator:**  Comma | **Charset:**  UTF-8 | **Graph Type:**  Directed or Undirected |
| **Import as:**  Nodes Table | **Time Rep:**  Intervals | **Edge Merge Strategy:**  Sum |

* Wait for the graph to appear (it will likely just look like a black blob)

**Step 2: Choose the Layout**

**Diagram

Description automatically generated**

**Force Atlas:** Emphasizes complementarities and pulls strongly connected nodes together and pushes weakly connected nodes apart; useful for exploration of the network as it doesn’t include any bias in plotting.Best for communities of less than 10,000 nodes that are expected to be well connected among one another, in other words, there is a small amount of steps between most nodes. Also, best used when there is node overlapping. It has the following parameters:

* + **Inertia**: Increase to conserve node speed at each new pass
  + **Repulsion Strength:** Increase to strengthen each node rejecting others
  + **Attraction Strength:** Increase to strengthen node-pair attraction
  + **Maximum Displacement:** Increase toprevents node rejection when nodes are too close
  + **Auto stabilize function:** Check to move unstable nodes more slowly
  + **Autostab Strength:** Increase to auto stabilize funtion
  + **Autostab sensibility:** Increase sensitivity of auto stabilize function
  + **Gravity:** Increase to bring nodes to center and lessens dispersion of disconnected components
  + **Attraction Distrib:** Increase to push nodes with high out-degree (hubs) at the periphery and puts nodes with a high in-degree (authorities) more central
  + **Adjust by Sizes:** Avoid node overlapping, depending on size of node
  + **Speed:** Increases convergence speed at the cost of precision

**Force Atlas 2:** Improved version of Force Atlas for large networks using a scaling parameter instead of the “attraction” and “repulsion” forces. It has the following parameters**:**

* + **Linear attraction & logarithmic repulsion:** Increase to tightens clusters
  + **Scaling:** Increasing to make the graph sparser
  + **Edge-Weight Influence:** 0 to 1; with 0 being no edge weight

**Fruchterman-Reingold*:*** This layout emphasizes complementarities and works by placing similar nodes in the same vicinity and placing dissimilar nodes far from each other. Disconnected components are thus easy to visualize. Best for datasets with less than 1,000 nodes and builds a graph that does not use edge weights. It has the following parameters:

* + **Area:** Graph size area
  + **Gravity:** Increase to attract nodes to center and lessens dispersion of disconnected components
  + **Speed:** Increases convergence speed at the cost of precision

**Yifan Hu Multilevel Layout:** This layout emphasizes complementarities and is useful on large graphs as the algorithm is very fast. It uses Barnes-Hut calculation to calculate repulsive forces between distant nodes. Best used on datasets including 100 to 100,000 nodes and builds a graph that does not use edge weight. It has the following parameters:

* + **Step Ratio:** Updates the step size; increasing it provides better quality but slows speed
  + **Optimal Distance:** Increasing places nodes further apart
  + **Theta:** Decreasing values gives a higher accuracy

**OpenOrd:** This layout emphasizes divisions and is useful in detecting clusters of nodes on large graphs by cutting long edges to allow clusters to separate. Best for communities of less than 1M nodes when building an undirected weighted graph. It has the following parameters:

* + **Edge Cut:** Controls the maximum length of an edge ex) for an edge cut of .8 every edge with a length over 80% of the next longest edge length is cut.
  + **Num iterations:** Increasing expands the clusters
  + **Random seed:**

**Circular:** This layout emphasizes ranking by drawing nodes in a circle ordered by ID, a centrality measure, or an attribute. Best used on datasets with less than 1M nodes and shows the distribution of nodes with their links. It has the following parameters:

* + **Node Layout Direction:** Clockwise or counter-clockwise
  + **Prevent Node Overlap:** Yes or no
  + **Fixed Diameter:** Choose fixed diameter
  + **Order Nodes by:** ID, centrality measure or attribute

**Radial:** This layout emphasizes ranking by grouping nodes and drawing the groups in axes that radiate outward from the center of the graph. The groups are generated by a centrality measure or an attribute. Best used with datasets with less than 1M nodes and to study similarity of nodes by showing the distribution of nodes inside their groups.

**GeoLayout:** This layout emphasizes geographic repartition using latitude and longitude coordinates to set node positions in the graph.

**Step 3: Using the Statistics Tab**

**Conducting Network Level Analysis**

Under the Statistics Tab you can run any of the following:

**Average Degree: Average number of edges connected to a node**

This will tell you how many in-degree and out-degrees a node has. In-degree is defined as incoming connections from a particular node while out degree is outgoing connections from a particular node to others. The degree centrality measure will output a **degree distribution chart** and metric for each node which is essentially a chart showing you the nodes with the highest popularity metric i.e. well-connected.

**Avg. Weighted Degree: Average sum of weights of edges connected to a node**

This will output a **weighted degree distribution** chart and metric for each node. The number of unique connections and the weight of those connection (how many times 2 people have connected) a specific user has within the network.

**Network Diameter: Longest shortest path between nodes in a graph**

This will output measure for **closeness** and **betweenness** centrality:

**Closeness:** How well connected a node is to all other nodes in the network. How many nodes does one particular node have to go through to reach another particular node. Example: In a hierarchical organization, how many people would I have to go through to get a message to the CEO?

**Betweenness:** The number of shortest paths the node is included in relative to the total number of shortest paths. Example: In a hierarchical organization, someone who has direct access to the CEO but also the most connections to individuals in the organization would have the highest betweenness centrality.

**Graph Density: Measures how close the graph is to complete**

**HITS: 2 Values; Value of info stored at each node and quality of nodes links**

**Modularity: Community Detection Algorithm**

Outputs a metric that defineshow easily the network can be split into smaller, more well-connected, communities.

**PageRank: Ranks nodes according to how often a user following links will non-randomly reach the node page**

**Connected Components: Determines the number of connected components in a network**

**Conducting Node Level Analysis**

Under the Statistics Tab you can run any of the following:

**Average Clustering Coefficient: Averages how nodes are embedded in their cluster**

**Eigenvector Centrality: Measures node importance based on connections; the sum of the centrality measures of all nodes connected to a node**

**Conducting Edge Level Analysis**

Under the Statistics Tab you can run any of the following:

**Avg. Path Length:** Avg. number of steps to get from one randomly selected node to another

**Using the Appearance Tab**

There are three options to encode information in the color of nodes:

**Unique:** change the color of all nodes of the graph to the same color

**Partition:** break the nodes into color-coded groups

**Ranking:** color-code the nodes on a scale (for example, by degree)

You can also change the node by size

**References**

https://gephi.org/tutorials/gephi-tutorial-layouts.pdf