Universal College of Engineering, Kaman
Department of Computer Engineering
Subject: Big Data Analytics Laboratory

Roll No: 65	Namee: Hari Jha	Batch:A3	Div:A

Experiment No. 06

Aim:

Social Network Analysis using R (for example: Community Detection Algorithm).

Theory:

1. Introduction to Social Network Analysis (SNA)

Social Network Analysis (SNA) is a method used to analyze the structure of social relationships through network theory. A network is composed of nodes (representing entities such as individuals, organizations, or concepts) and edges (representing the relationships or interactions between these nodes). SNA seeks to understand how the structure of these connections influences behaviors, spreads information, or defines roles within the network.

2. Data Representation and Network Construction

- Theory: Social networks are typically represented as graphs, with nodes representing the actors within the network and edges representing the relationships between them. In R, the `igraph` package is commonly used to create and analyze these networks.
- Directed vs. Undirected Graphs: Directed graphs have edges with a specific direction, indicating the flow of relationships (e.g., follower-following). In contrast, undirected graphs treat all edges as bidirectional.

3. Degree Centrality and Node Properties

- Theory: Centrality measures help identify the most important nodes in a network. Degree centrality is the simplest measure, representing the number of direct connections a node has.
- Node Degree: The degree of a node indicates its connectivity. Nodes with higher degrees are often more influential or central in the network.

4. Degree Distribution Analysis

- Theory: The degree distribution provides insights into the overall connectivity of the network. In many real-world networks, this distribution follows a power law, where a small number of nodes (hubs) have a disproportionately high degree, while the majority have few connections. Analyzing this distribution helps in understanding the resilience and potential vulnerabilities of the network.

5. Network Visualization

- Theory: Visualization is a crucial part of SNA as it allows for the intuitive understanding of network structure and the relationships between nodes. Different layout algorithms, such as Fruchterman-Reingold and Kamada-Kawai, are used to position the nodes in a way that reveals patterns like clustering and centrality.
- Layout Algorithms: These algorithms arrange the nodes in a space so that the graph is easier to read. Force-directed layouts simulate physical forces (e.g., attraction and repulsion) to position nodes such that connected nodes are closer together and overlapping is minimized.

6. Identifying Hubs and Authorities

- Theory: In directed networks, certain nodes act as hubs or authorities.
- Hubs: Nodes that point to many other nodes, acting as connectors within the network.
- Authorities: Nodes that are pointed to by many hubs, often representing important or authoritative entities within the network. The HITS (Hyperlink-Induced Topic Search) algorithm is used to identify these nodes.

7. Community Detection

- Theory: Community detection is the process of identifying groups of nodes that are more densely connected internally than with the rest of the network. This is a key aspect of SNA, as it helps to uncover hidden structures within the network.
- Edge Betweenness: One method of community detection involves calculating the betweenness of edges (i.e., how often an edge is part of the shortest path between any two nodes). By progressively removing edges with high betweenness, the network can be divided into distinct communities.
- Modularity: A measure used to evaluate the quality of community detection. High modularity indicates strong community structure, where nodes are densely connected within communities but sparsely connected between them.

8. Practical Application of Community Detection

- Theory: By applying community detection algorithms, one can uncover subgroups or clusters within the network. These clusters can represent communities of interest, influence, or other meaningful groupings depending on the context of the network.
- Use Cases: Identifying communities can help in various applications, such as targeted marketing, understanding social dynamics, improving recommendation systems, and analyzing the spread of information or diseases.

Code:

1. Loading Libraries and Data library(igraph)

data <- read.csv("/Users/krishlakhani/Downloads/socialnetworkdata (2).csv", header = T) y <- data.frame(data$first, data$second) net <- graph.data.frame(y, directed = T)

Theory:

- Loading Data: The `read.csv` function loads the social network data from a CSV file. The data is assumed to consist of pairs of nodes that represent edges in the network.
- Creating the Graph: The `graph.data.frame` function converts the data frame into a graph object. The `directed = T` argument indicates that the network is directed, meaning that the edges have a direction from one node to another.

2. Analyzing Node Properties

```
V(net)
V(net)$label <- V(net)$name
V(net)$degree <- degree(net)
```

Theory:

- Vertex (Node) Attributes: The `V(net)` function accesses the vertices (nodes) of the graph. The script assigns labels to each node based on their names and calculates the degree of each node (i.e., the number of edges connected to the node).
- Degree: The degree of a node is a fundamental measure in SNA, indicating the level of connectivity a node has within the network.

3. Visualizing Node Degree Distribution

```
hist(V(net)$degree,
     col = 'green',
     main = 'Histogram of Node Degree',
     ylab = 'Frequency',
     xlab = 'Degree of the vertices')
```

Theory:

- Degree Distribution: The histogram provides a visual representation of the degree distribution in the network. This distribution often follows a power-law in social networks, indicating that a few nodes have many connections while most have few.

4. Basic Network Visualization

```
set.seed(222)      plot(net,
vertex.color     =     'blue',
```

```
vertex.size = 2,
edge.arrow.size = 0.1,
vertex.label.cex = 0.8)
```

Theory:
- Network Plot: The `plot` function generates a basic visualization of the network, with nodes colored in blue and edges directed with small arrows. The size and label size of the vertices are customized to enhance the readability of the plot.

5. Advanced Network Layouts

```
plot(net, vertex.color=rainbow(52),
    vertex.size=V(net)$degree*0.4,
    edge.arrow.size=0.2,
    layout=layout.fruchterman.reingold)

plot(net, vertex.color=rainbow(52),
    vertex.size=V(net)$degree*0.4,
    edge.arrow.size=0.2,
    layout=layout.graphopt)

plot(net, vertex.color=rainbow(52),
    vertex.size=V(net)$degree*0.4,
    edge.arrow.size=0.2,
    layout=layout.kamada.kawai)
```

Theory:
- Graph Layouts: The script uses different layout algorithms to arrange the nodes in the plot:
- Fruchterman-Reingold: A force-directed layout that spreads nodes based on simulated physical forces to reduce overlap and improve readability.
- Graphopt: An alternative force-directed layout that emphasizes avoiding edge crossings.
- Kamada-Kawai: Another force-directed layout, focusing on preserving the graph's geometric structure.

6. Identifying Hubs and Authorities

```
hs <- hub_score(net)$vector as <-
authority.score(net)$vector
set.seed(123) plot(net,
vertex.size=hs*30, main = 'Hubs',
```

```
    vertex.color = rainbow(52),
    edge.arrow.size=0.1,
      layout = layout.kamada.kawai)
set.seed(123) plot(net,
    vertex.size=as*30, main =
    'Authorities', vertex.color =
    rainbow(52),
    edge.arrow.size=0.1, layout =
    layout.kamada.kawai) Theory:
```

- Hubs and Authorities: The `hub_score` and `authority.score` functions compute hub and authority scores using the HITS (Hyperlink-Induced Topic Search) algorithm.
- Hubs: Nodes that link to many authoritative nodes.
- Authorities: Nodes that are linked by many hubs.
- Visualization: The size of the nodes is proportional to their hub or authority scores, highlighting the most influential nodes in the network.

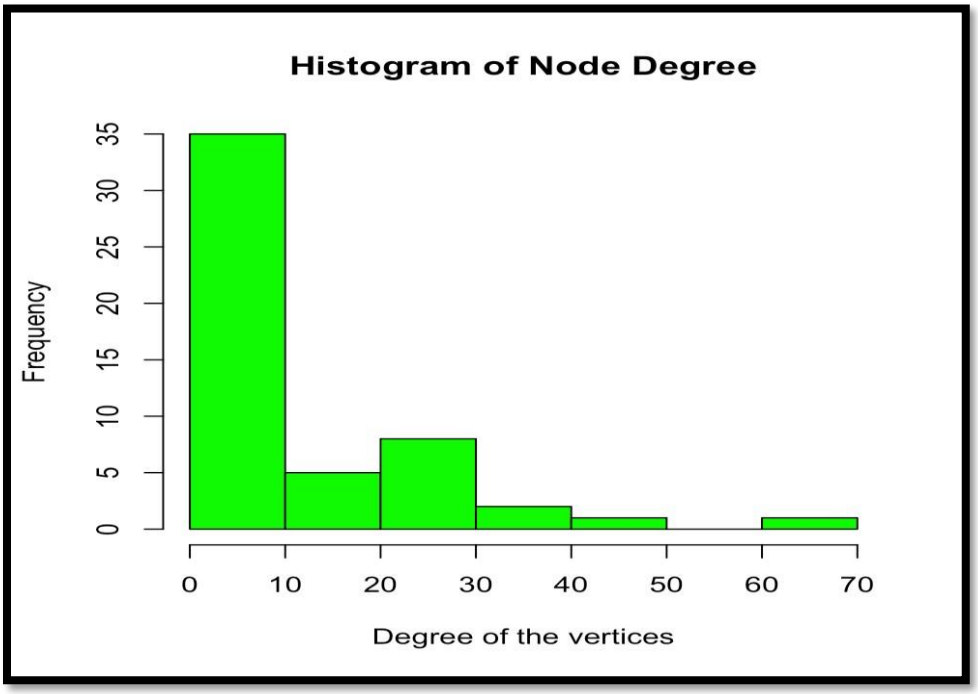
7. Community Detection using Edge Betweenness

```
net <- graph.data.frame(y, directed = F)
cnet <- cluster_edge_betweenness(net)
plot(cnet, net, vertex.size = 10,
vertex.label.cex = 0.8) Theory:
```

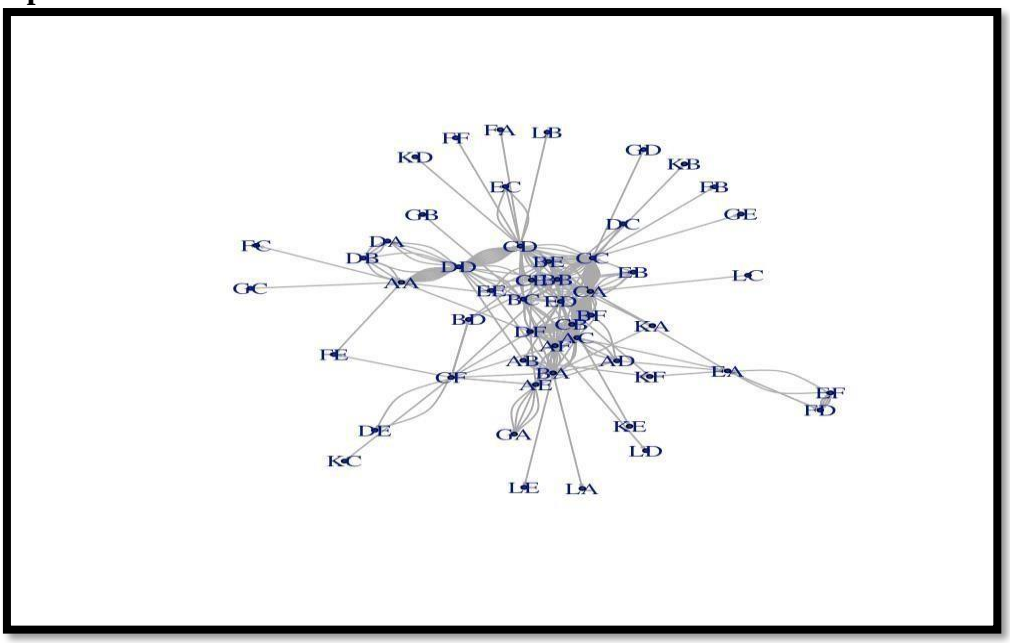
- Undirected Network: The script converts the network to an undirected graph, which is necessary for some community detection algorithms.
- Edge Betweenness Community Detection: This method identifies communities by iteratively removing edges with high betweenness, which are likely to connect different communities. The resulting communities are then visualized, with node sizes and labels adjusted for clarity.

Output:

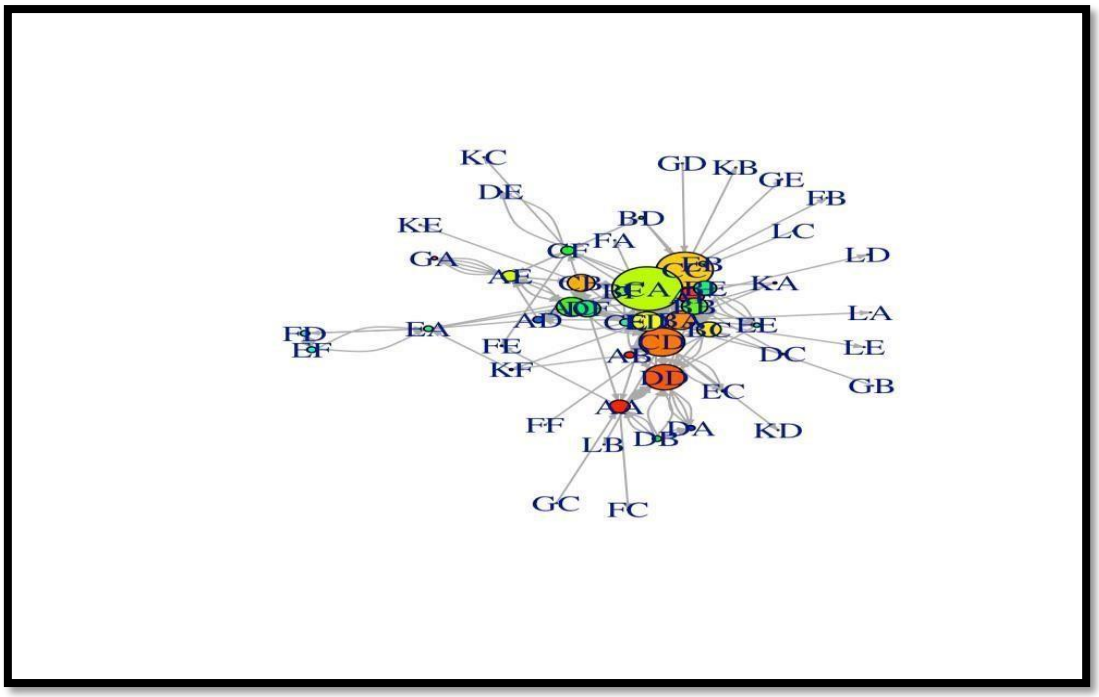
Histogram of Node Degree



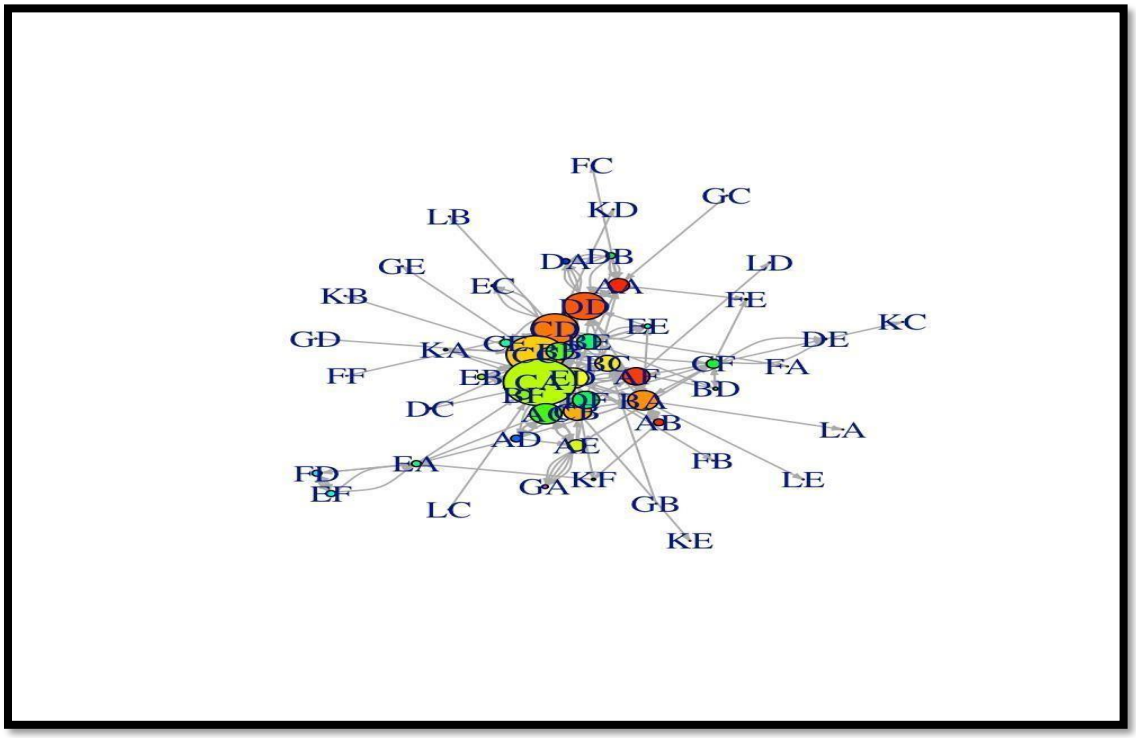
Base Graph



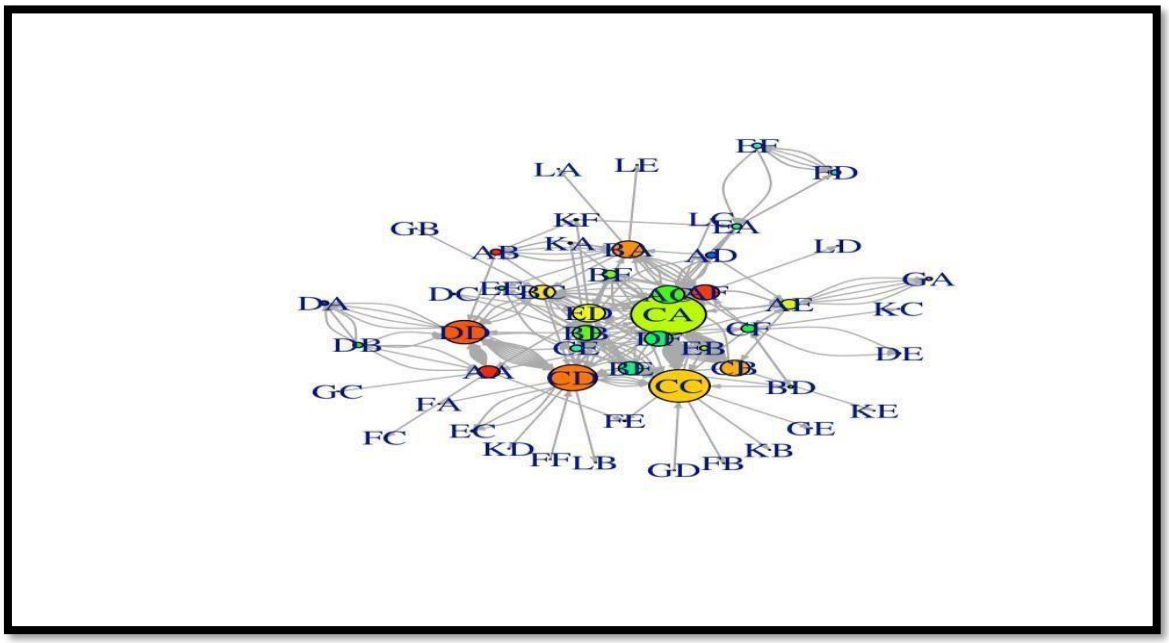
Fruchterman-Reingold



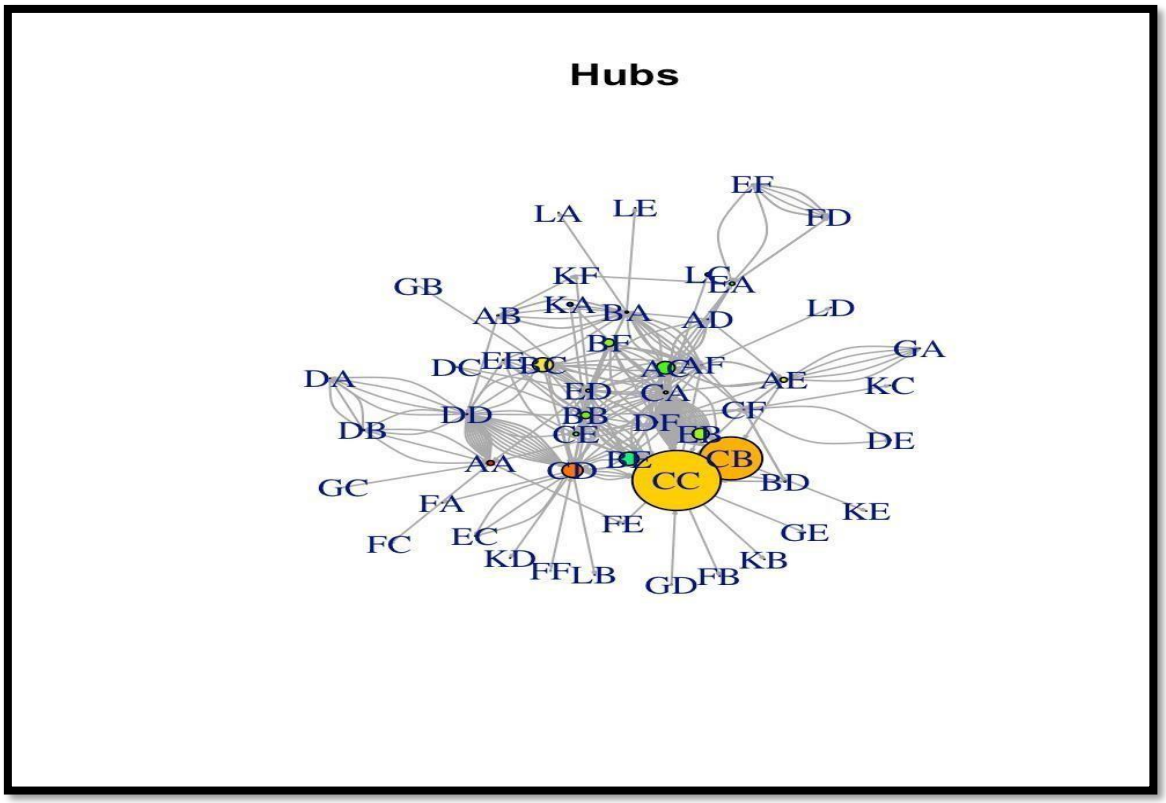
Rainbow Graphopt



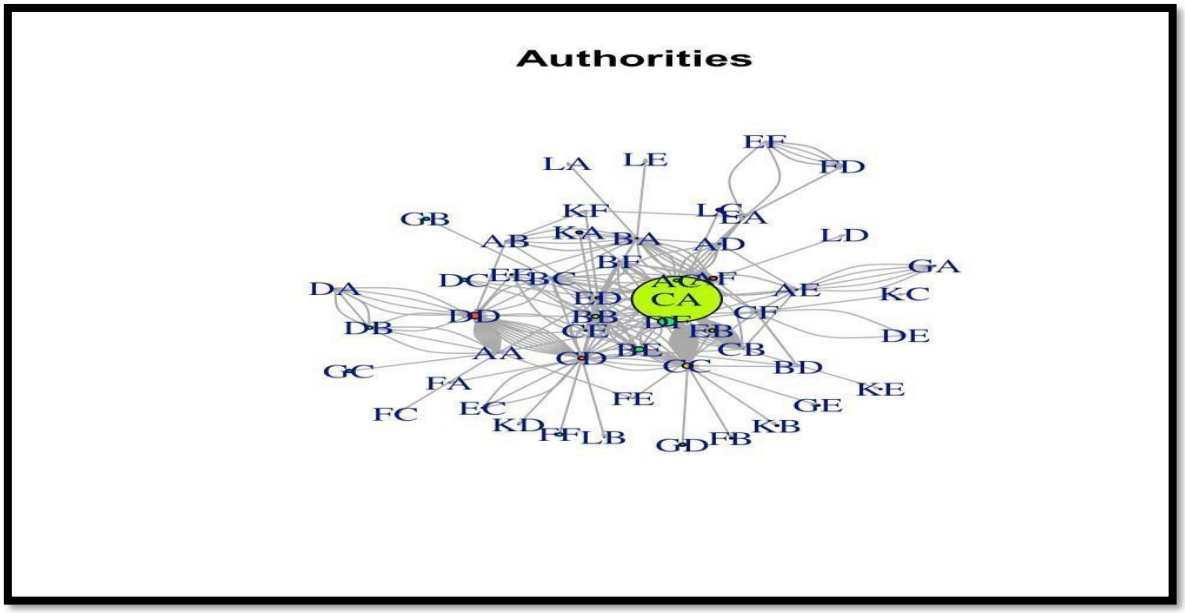
Kamada Kawai



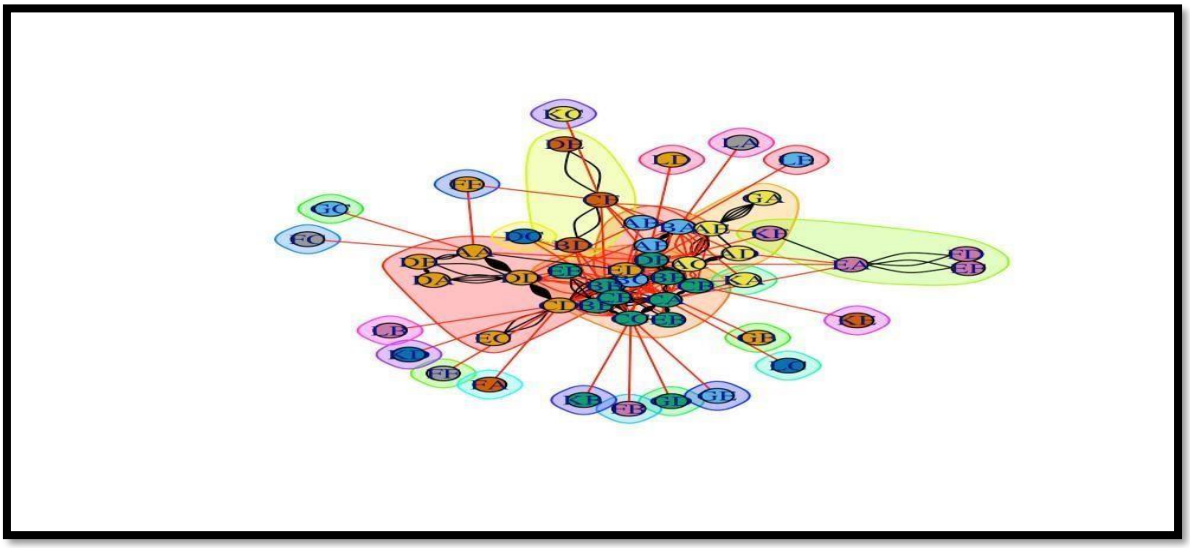
Hubs



Authorities



Community Detection using Edge Betweenness



Conclusion:

SNA and community detection provide powerful tools for understanding complex networks. By combining various measures like centrality, degree distribution, and community structure, one can gain deep insights into the behaviour and characteristics of the network. The results from these analyses can be used to inform decision-making in fields ranging from sociology and marketing to epidemiology and computer science.