Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Computer Science and Engineering (Data Science)

Subject: Social Network Analysis Laboratory (DJ19DSL8014)

Bhuvi Ghosh
60009210191

AY: 2024-25

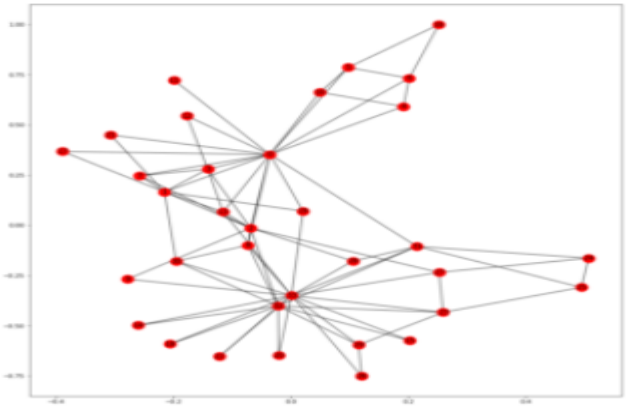
Experiment 2

Aim: Building a network and network measures using NetworkX: a) Degree & Degree Distance b) Clustering c) Node Centrality measure d) Helper Function

Theory:

1. Introduction

Building a Network and Network Measures using "NetworkX" is an insightful experiment that dives into the world of social network analysis using Python's powerful NetworkX library. This process involves constructing a network graph to model relationships and interactions, followed by an in-depth examination of key network properties. The experiment focuses on assessing network characteristics such as degree and degree distribution, clustering, and node centrality measures, providing a comprehensive understanding of the structural and influential dynamics within the network. The inclusion of helper functions streamlines the analysis, making it an efficient and effective approach to uncovering the complexities of social networks.



Network Representation

To build a network and analyze network measures using NetworkX, follow this stepwise procedure:

- Import NetworkX: Start by importing the NetworkX library in Python.
- Create a Network: Use NetworkX to create a graph object, which could be directed or undirected.
- Add Nodes and Edges: Populate the graph with nodes and edges representing your network's entities and their relationships.
- Degree & Degree Distribution: Calculate the degree of each node (number of connections) and analyze the degree distribution across the network.
- Clustering: Compute the clustering coefficient for each node to understand the degree to which nodes tend to cluster together.
- Node Centrality Measures: Calculate centrality measures like betweenness centrality, closeness centrality, and eigenvector centrality to identify influential nodes in the network.
- Helper Functions: Utilize or create helper functions in Python to streamline repetitive tasks or calculations in your network analysis.
- Degree & Degree Distance:

Degree: The degree of a node in a network is the number of connections or edges the node has to other nodes.

- Degree Distance: This is not a standard term in network analysis. If you meant the sum of the degrees of a pair of nodes, you can simply sum their degrees.

```
import networkx as nx

def calculate_degree(graph):

 return dict(graph.degree())

def calculate_degree_distance(graph, node1, node2):

 degree_node1 = graph.degree(node1)

 degree_node2 = graph.degree(node2)

 return degree_node1 + degree_node2
```

- **Clustering:**

Clustering coefficient of a node measures the likelihood that its neighbors are also connected. The global clustering coefficient is the average local clustering coefficient over all the nodes in the graph.

```
def calculate_clustering(graph):
    return nx.clustering(graph)
def calculate_average_clustering(graph):
    return nx.average_clustering(graph)
```

- **Node Centrality Measures:**

Centrality measures help to identify the most important vertices within a graph. Examples are degree centrality, betweenness centrality, closeness centrality, and eigenvector centrality.

```
def calculate_degree_centrality(graph):
    return nx.degree_centrality(graph)
def calculate_betweenness_centrality(graph):
    return nx.betweenness_centrality(graph)
def calculate_closeness_centrality(graph):
    return nx.closeness_centrality(graph)
def calculate_eigenvector_centrality(graph):
    return nx.eigenvector_centrality(graph)
```

- **Helper Function:**

A helper function could be many things depending on what you need. Here's a simple example to add edges to a graph from a list of tuples.

```
def add_edges(graph, edge_list):
    for edge in edge_list:
        graph.add_edge(edge[0], edge[1])
```

Interpretation: Students should focus on understanding the significance of each network metric, recognizing how Degree indicates a node's connectivity and potential influence, how Clustering Coefficients reveal the propensity for nodes to form close-knit groups, and how various Centrality measures uncover the most pivotal nodes, whether in terms of control over information flow, communication efficiency, or strategic connections within the network.

Research Perspective:

- **Degree & Degree Distance:**Explore the impact of individual nodes' connectivity on network structure and resilience. Assess how nodes with higher degrees influence network functionality and stability.
- **Clustering Coefficient:**Investigate the formation of tight-knit groups within the network and their effects on information dissemination, network robustness, and community emergence.
- **Node Centrality Measures:**Analyze the roles of central nodes in shaping network dynamics, information flow, and structural integrity, assessing their contribution to the network's overall performance.
- **Helper Functions:**Focus on developing and utilizing helper functions to enhance the efficiency and depth of network analysis, streamlining data processing, analysis, and visualization.

Lab Assignment:

Airline Route Network Analysis

You have been given the task to analyze the route network of an international airline. The nodes represent cities, and the edges represent direct flights between them. The airline has hubs in New York (NY), London (LD), Dubai (DB), and Tokyo (TK). Other cities included in the network are Los Angeles (LA), Mumbai (MU), Sydney (SY), and Beijing (BJ).

The direct flights (edges) between these cities are as follows:

NY ↔ LD
NY ↔ LA
LD ↔ DB
LD ↔ TK
DB ↔ MU

DB ↔ BJ

TK ↔ SY

TK ↔ BJ

LA ↔ SY

Tasks: Using NetworkX, examine various network measures.

Reference:

https://www.geeksforgeeks.org/network-centrality-measures-in-a-graph-using-networkx-python/

https://www.kirenz.com/post/2019-08-13-network_analysis/

Department of Computer Science and Engineering (Data Science)

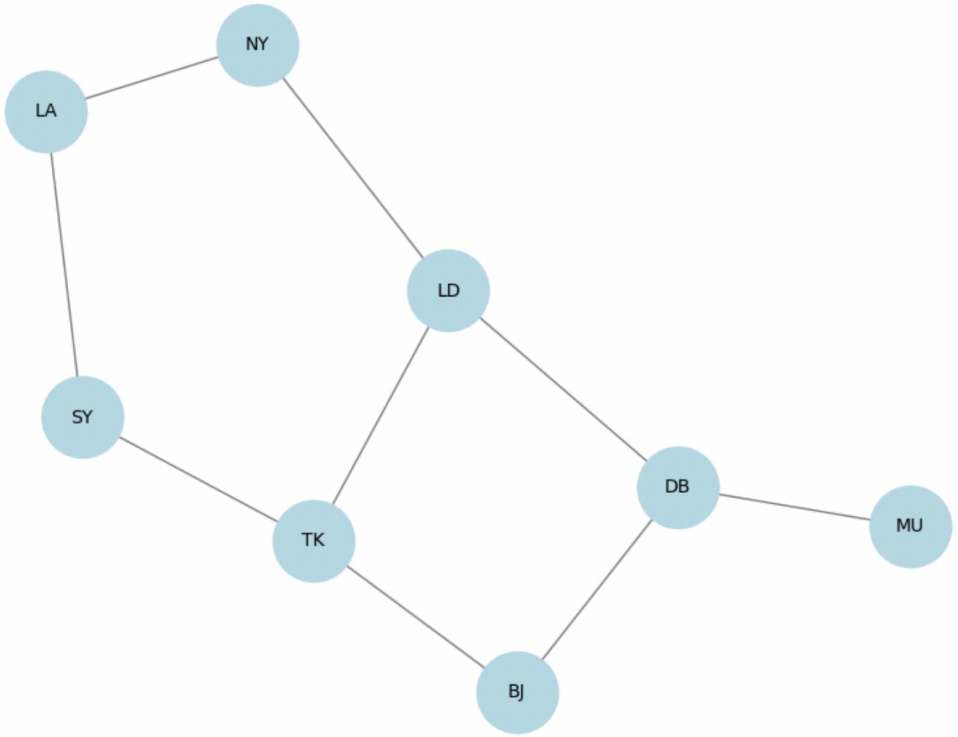
[14]
```
import networkx as nx
import matplotlib.pyplot as plt
```

[15]
```
G = nx.Graph()
```

[16]
```
cities = ["NY", "LD", "DB", "TK", "LA", "MU", "SY", "BJ"]
G.add_nodes_from(cities)
```

```
edges = [("NY", "LD"), ("NY", "LA"), ("LD", "DB"), ("LD", "TK"),
         ("DB", "MU"), ("DB", "BJ"), ("TK", "SY"), ("TK", "BJ"),
         ("LA", "SY")]
G.add_edges_from(edges)
```

```
plt.figure(figsize=(8, 6))
pos = nx.spring_layout(G)
nx.draw(G, pos, with_labels=True, node_color='lightblue', edge_color='gray', node_size=2000, font_size=10)
plt.title("Airline Route Network")
plt.show()
```



Degree Centrality:

```
[5] degree_centrality = nx.degree_centrality(G)
    print("Degree Centrality:", degree_centrality)
```

```
Degree Centrality: {'NY': 0.2857142857142857, 'LD': 0.42857142857142855, 'DB': 0.42857142857142855, 'TK': 0.42857142857142855, 'LA': 0.2857142857142857, 'MU': 0.142857142857141
```

Betweenness Centrality:

```
[6] betweenness_centrality = nx.betweenness_centrality(G)
    print("Betweenness Centrality:", betweenness_centrality)
```

```
Betweenness Centrality: {'NY': 0.14285714285714285, 'LD': 0.38095238095238093, 'DB': 0.3333333333333333, 'TK': 0.2857142857142857, 'LA': 0.047619047619047616, 'MU': 0.0, 'SY':
```

Shortest Path Length:

```
shortest_paths = dict(nx.all_pairs_shortest_path_length(G))
print("Shortest Path Lengths:")
for city, paths in shortest_paths.items():
    print(f"{city}: {paths}")
```

```
Shortest Path Lengths:
NY: {'NY': 0, 'LD': 1, 'LA': 1, 'DB': 2, 'TK': 2, 'SY': 2, 'MU': 3, 'BJ': 3}
LD: {'LD': 0, 'NY': 1, 'DB': 1, 'TK': 1, 'LA': 2, 'MU': 2, 'BJ': 2, 'SY': 2}
DB: {'DB': 0, 'LD': 1, 'MU': 1, 'BJ': 1, 'NY': 2, 'TK': 2, 'LA': 3, 'SY': 3}
TK: {'TK': 0, 'LD': 1, 'SY': 1, 'BJ': 1, 'NY': 2, 'DB': 2, 'LA': 2, 'MU': 3}
LA: {'LA': 0, 'NY': 1, 'SY': 1, 'LD': 2, 'TK': 2, 'DB': 3, 'BJ': 3, 'MU': 4}
MU: {'MU': 0, 'DB': 1, 'LD': 2, 'BJ': 2, 'NY': 3, 'TK': 3, 'LA': 4, 'SY': 4}
SY: {'SY': 0, 'TK': 1, 'LA': 1, 'LD': 2, 'BJ': 2, 'NY': 2, 'DB': 3, 'MU': 4}
BJ: {'BJ': 0, 'DB': 1, 'TK': 1, 'LD': 2, 'MU': 2, 'SY': 2, 'NY': 3, 'LA': 3}
```

Clustering coefficient:

```
[8] clustering_coefficient = nx.clustering(G)
    print("Clustering Coefficient:", clustering_coefficient)
```

```
Clustering Coefficient: {'NY': 0, 'LD': 0, 'DB': 0, 'TK': 0, 'LA': 0, 'MU': 0, 'SY': 0, 'BJ': 0}
```

Degree distribution:

```
[9] degrees = [degree for node, degree in G.degree()]
    print("Degree Distribution:", degrees)
```

```
Degree Distribution: [2, 3, 3, 3, 2, 1, 2, 2]
```

```
[10] def degree_distance(G):
         degree_dist = {}
         for node in G.nodes():
             degree_dist[node] = sum(abs(G.degree(node) - G.degree(neighbor)) for neighbor in G.neighbors(node))
         return degree_dist
```

Degree Distance:

```
degree_distances = degree_distance(G)
print("Degree Distance:", degree_distances)
```

```
Degree Distance: {'NY': 1, 'LD': 1, 'DB': 3, 'TK': 2, 'LA': 0, 'MU': 2, 'SY': 1, 'BJ': 2}
```

Conclusion of Airline Network Analysis:

1. **Degree Centrality**
    - London (LD), Dubai (DB), and Tokyo (TK) have the highest degree centrality (0.428), meaning they are directly connected to more cities than others.
    - Mumbai (MU) has the lowest degree centrality (0.142), indicating it has fewer direct connections.
2. **Betweenness Centrality**
    - London (LD) has the highest betweenness centrality (0.381), making it the most critical hub for shortest paths between other cities.
    - Dubai (DB) and Tokyo (TK) also have high values, meaning they play significant roles in connecting different parts of the network.
    - Mumbai (MU) has a betweenness centrality of 0, indicating it is not on any shortest path between other cities.
3. **Shortest Path Lengths**
    - New York (NY) is at most 3 steps away from any other city, indicating good connectivity.
    - Mumbai (MU) and Los Angeles (LA) have higher distances (4 steps), showing they are more peripheral in the network.
4. **Clustering Coefficient**
    - All cities have a clustering coefficient of 0, meaning there are no triangular connections (closed loops) between cities.
    - This indicates that while the airline provides direct flights between major hubs, it does not offer many indirect connections between non-hub cities.
5. **Degree Distribution**
    - Most cities have a degree between 1 and 3, meaning the network is not highly centralized, though hubs like London, Dubai, and Tokyo have more connections.

6. Degree Distance

- Dubai (DB) has the highest degree distance (3), meaning it has a mix of highly connected and less-connected neighbors.
- Los Angeles (LA) has a degree distance of 0, meaning its neighbors have the same number of connections as itself.