**Social Media Analytics Lab**

**Instructor: Dr. Janani Selvaraj**

**Lab 7. Calculating centrality measures and creating an interest graph for a Github user**

In this lab, you need to calculate the centrality measures of a graph and also create an interest graph of a Github user

1. Calculate the degree, betweenness, and closeness centrality measures of a krackhardt kite graph
2. Create an interest graph of a github user by adding ‘follows’ as edges
   1. Find the stargazers of a github user
   2. Create a graph of the star gazers using the networkx package and get the information about the graph
   3. Add "follows" edges between stargazers in the graph if any relationships exist

Sample code to create ‘follows’ as edges

import sys

for i, sg in enumerate(stargazers):

# Add "follows" edges between stargazers in the graph if any relationships exist

try:

for follower in sg.get\_followers():

if follower.login + '(user)' in g:

g.add\_edge(follower.login + '(user)', sg.login + '(user)',

type='follows')

except Exception as e: #ssl.SSLError

print("Encountered an error fetching followers for", sg.login, \

"Skipping.", file=sys.stderr)

print(e, file=sys.stderr)

print("Processed", i+1, " stargazers. Num nodes/edges in graph", \

g.number\_of\_nodes(), "/", g.number\_of\_edges())

print("Rate limit remaining", client.rate\_limiting)

Save the above created graph using pickle library for future use of the graph using the code below

nx.write\_gpickle(g, "github.gpickle.1")

1. Explore the graph with the updates ‘follows’ edges
   1. Get the information about the updated graph
   2. Find the number of ‘follow’ edges

print(len([e for e in g.edges(data=True) if e[2]['type'] == 'follows']))

* 1. Find the number of popular users and the top 10 users

c = Counter([e[1] for e in g.edges(data=True) if e[2]['type'] == 'follows'])

popular\_users = [ (u, f) for (u, f) in c.most\_common() if f > 1 ]

print("Number of popular users", len(popular\_users))

print("Top 10 popular users:", popular\_users[:10])

* 1. Remove the super node from the graph and calculate the centrality measures

h = g.copy()

h.remove\_node('Mining-the-Social-Web(repo)')

1. Visualise the created interest graph
   1. Create a subgraph from the original interest graph- select the user nodes and get the information about the updated graph

mtsw\_users = [n for n in g if g.node[n]['type'] == 'user']

h = g.subgraph(mtsw\_users)

print("Stats on the extracted subgraph")

print(nx.info(h))

* 1. Visualise the extracted graph using matplotlib and networkx

import matplotlib.pyplot as plt

import warnings

warnings.filterwarnings("ignore")

%matplotlib inline

fig = plt.figure(figsize=(15,15))

ax = fig.add\_subplot(111)

labels = dict([(n, n.split('(user)')[0]) for n in h.nodes()])

nx.draw(h, pos=nx.spring\_layout(h), arrows=False, ax=ax, node\_size=50, edge\_color='#aaaaaa', alpha=0.8, labels=labels, font\_size=8)