# 经济管理学院

# 课　程　报　告

## （复杂网络与社会计算）

题　　目：week5 作业

课程教师：赵吉昌

学院/专业：信息管理与信息系统

学生姓名：范春

学　　号：21377061

2024 年 3 月 27 日

北京航空航天大學
BEIHANG UNIVERSITY

作业要求如下：

下载https://github.com/marcessz/Twenty-Years-of-Network-Science中的合作者网络，并做如下分析。

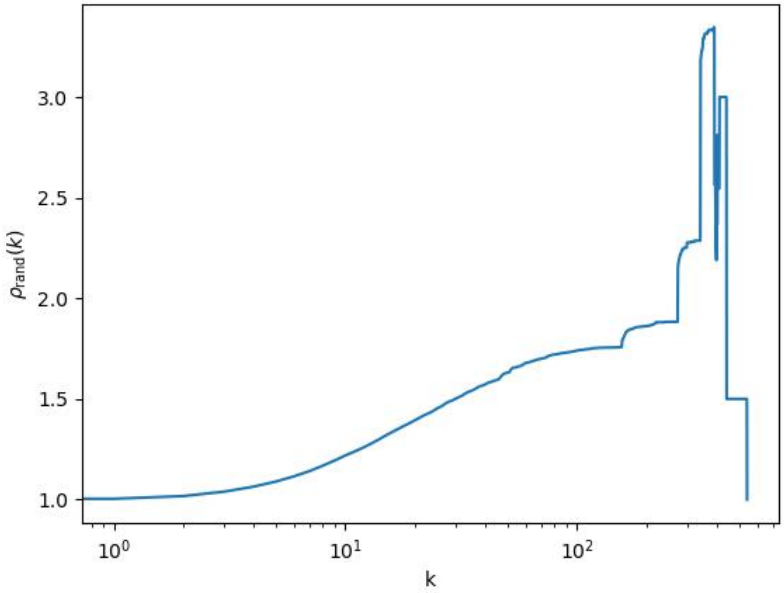
1. 分析网络的rich-club特征，并观察其随度的变化趋势。

2. 分析网络的度-度相关特征，可以分别观察节点度与邻居平均度的关系，以及同配性系数的正负。

3. 谣言的传播亦可用病毒传染模型来建模。请使用SIS和SIR模型分别建模该网络上的谣言传播过程，并观察在不同传播参数（感染率，治愈率）下，不同状态所占比例的变化情况。

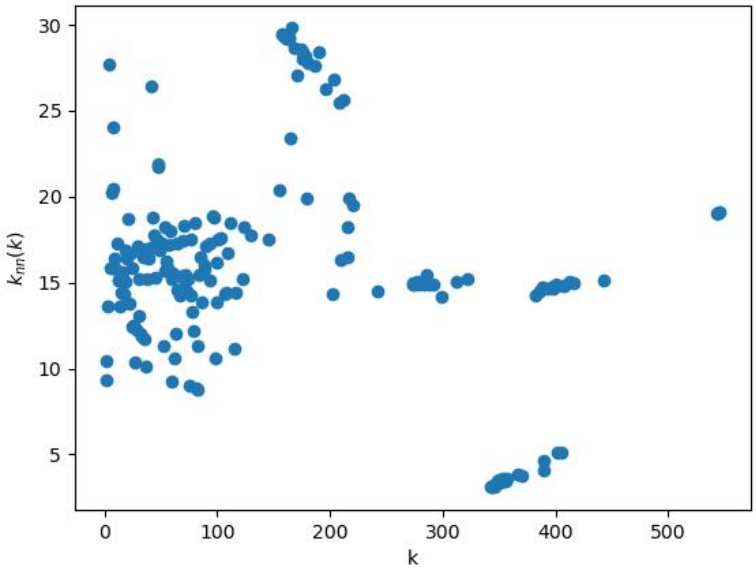
4. 假如需要采取策略来进行谣言的“阻断”（辟谣），以SIS模型为例，比较随机策略、目标策略和熟人策略的优劣。注意，可以先在网络中对相同数目的节点进行“阻断”（将被辟谣节点移除），再观察SIS模型仿真结果的变化。

任务一：




由上图可知，$\rho_{rand}(k)$>1，因此该网络结构表现出明显的 rich-club 性质。

任务二：

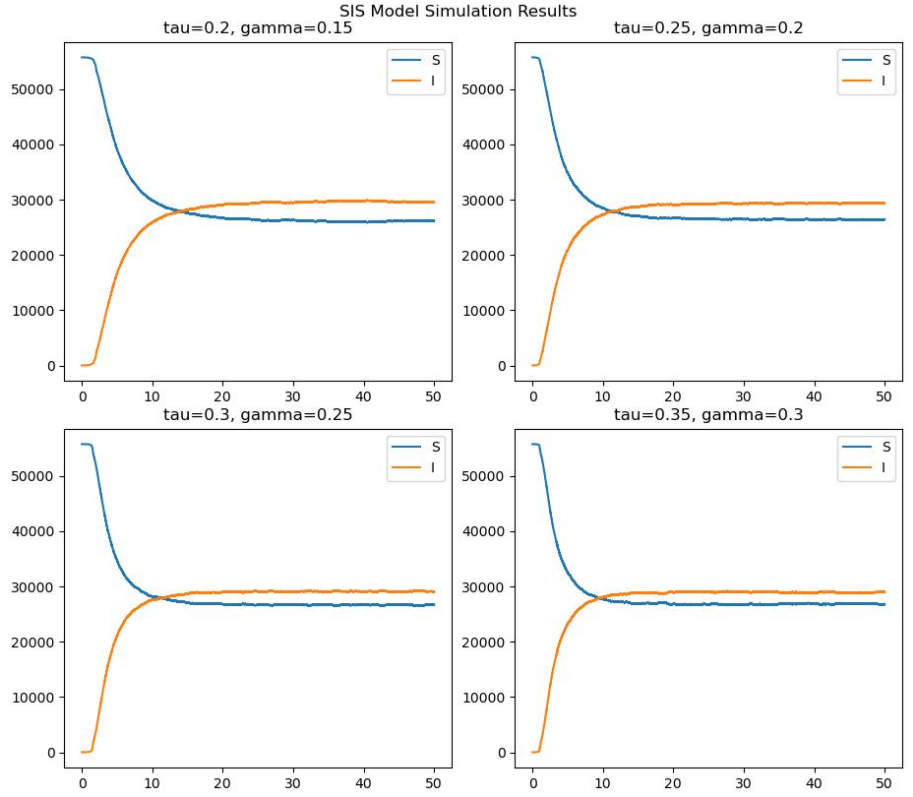


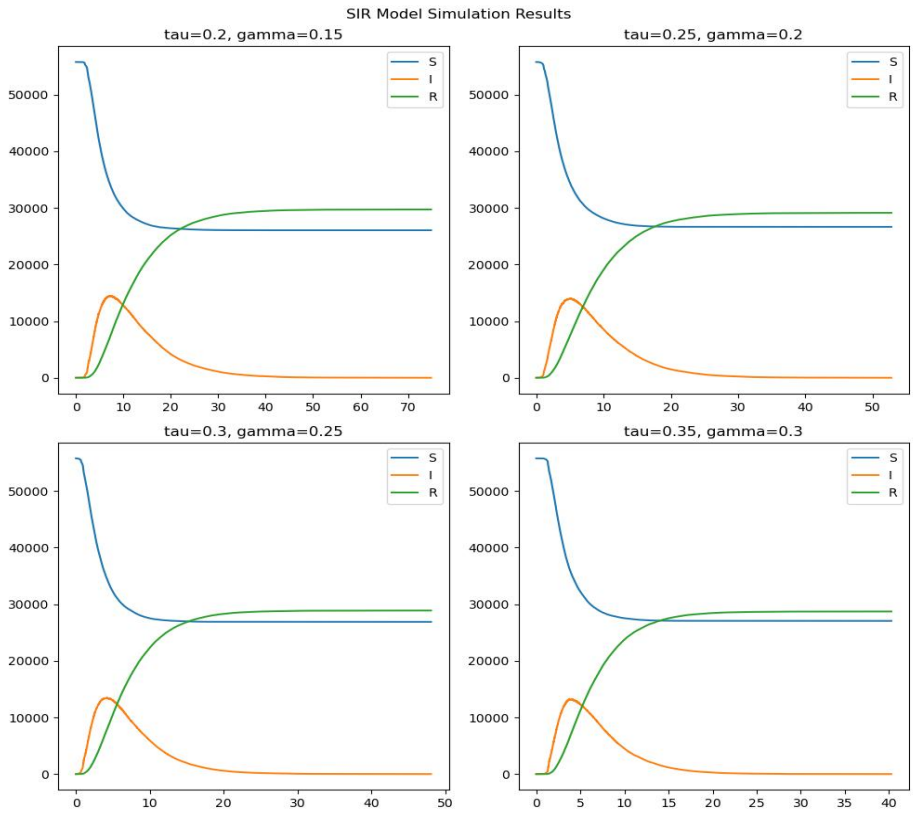

由上图可知，当节点的度在 100 至 200 之间时，其邻居节点平均度较高，而当节点的度大于 200 时，其邻居节点的平均度出现明显的下降。

同配性系数：0.9807135015561302。该网络的同配性系数大于零，表示网络是同

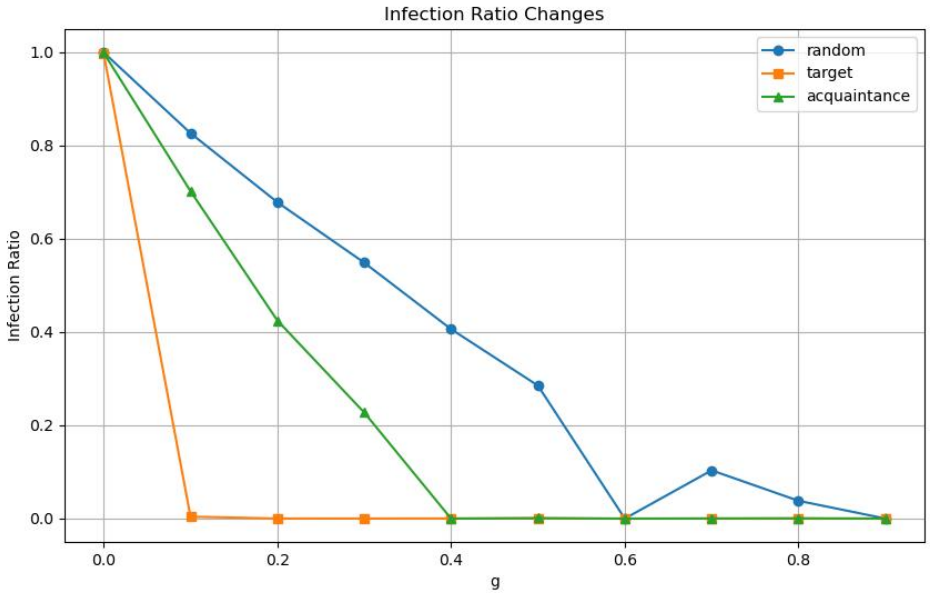
配的，意味着度大的节点倾向于和度大的节点连接。

任务三：



任务四：




由上图可知，在该网络中，目标免疫的效果最好。

主要代码如下：

```
import networkx as nx
import pandas as pd
import pickle
import EoN
import numpy as np
import random

#生成网络
def network(filePath):
    G = nx.Graph()
    df = pd.read_excel(filePath)
    columns = df.columns
    for index, row in df.iterrows():
        col1_value = row[columns[0]]
        col2_value = row[columns[1]]
        G.add_edge(int(col1_value),int(col2_value))
    return G
#分析网络的 rich_club 特征
def rich_club(G):
    rc = nx.rich_club_coefficient(G, normalized=True, Q=1)

    with open('C:\\Users\\范春\\Desktop\\week5\\rich_club_coefficient.pkl', 'wb') as f:
        pickle.dump(rc, f)
```

```
#分析网络的度-度相关性
def degree_degree(G):
    avg_neighbor_degrees = nx.average_neighbor_degree(G)
    r = nx.degree_assortativity_coefficient(G)
    degrees_to_avg_neighbors = {}
    for node, degree in G.degree():
        if degree in avg_neighbor_degrees:
            if degree not in degrees_to_avg_neighbors:
                degrees_to_avg_neighbors[degree] = []
            for neighbor in G.neighbors(node):
                if G.degree(neighbor) in avg_neighbor_degrees:
                    degrees_to_avg_neighbors[degree].append(avg_neighbor_degrees[G.degree(neigh
bor)])
    with open('C:\\Users\\范春\\Desktop\\week5\\avg_neighbor_degrees.pkl', 'wb') as f:
        pickle.dump(degrees_to_avg_neighbors, f)
    return r
def SIS_SIR(G):
    tau = [0.2,0.25,0.3,0.35]
    gamma = [0.15,0.2,0.25,0.3]
    for i in range(4):
        t, S, I = EoN.fast_SIS(G, tau[i], gamma[i],tmax=50, initial_infecteds=range(5))
        filename = f"C:\\Users\\范春\\Desktop\\week5\\SIS_tau_{tau[i]}_gamma_{gamma[i]}.csv"
        np.savetxt(filename,np.column_stack((t,S,I)),delimiter=',',header='t,S,I')

    for i in range(4):
        t, S, I, R = EoN.fast_SIR(G,tau[i],gamma[i], initial_infecteds=range(5))
        filename = f"C:\\Users\\范春\\Desktop\\week5\\SIR_tau{tau[i]}_gamma_{gamma[i]}.csv"
        np.savetxt(filename,np.column_stack((t,S,I,R)),delimiter=',',header='t,S,I,R')
def remove_nodes(G, strategy, num_nodes_to_remove):
    if strategy == 'random':
        nodes_to_remove = random.sample(G.nodes(),num_nodes_to_remove)
    elif strategy == 'target':
        nodes_to_remove = sorted(G.degree, key=lambda x: x[1], reverse=True)[:num_nodes_to_remove]
        nodes_to_remove = [node[0] for node in nodes_to_remove]
    elif strategy == 'acquaintance':
        nodes_to_remove = []
        attempts = 0
        max_attempts = len(G.nodes()) # 设置尝试的最大次数
        while len(nodes_to_remove) < num_nodes_to_remove and attempts < max_attempts:
            random_node = random.choice(list(G.nodes()))
            if random_node not in nodes_to_remove:
                neighbors = list(G.neighbors(random_node))
                if neighbors:
```

```
                    acquaintance = random.choice(neighbors)
                    if acquaintance not in nodes_to_remove:
                        nodes_to_remove.append(acquaintance)
            attempts += 1
    G.remove_nodes_from(nodes_to_remove)
    return G
def question4(G):
    # 初始化免疫比例列表和对应的感染比例变化字典
    immunization_fractions = np.linspace(0, 0.9, 10)
    strategies = ['random','target','acquaintance']
    tau = 0.3
    gamma = 0.15
    initial_infecteds1 = random.sample(list(G.nodes()), 5)
    t0, S0, I0 = EoN.fast_SIS(G, tau, gamma,tmax=50, initial_infecteds=initial_infecteds1)
    columns = ['g', 'random', 'target', 'acquaintance']
    df = pd.DataFrame(columns=columns)
    for immunization_fraction in immunization_fractions:
        data = []
        data.append(immunization_fraction)
        print(immunization_fraction)
        for strategy in strategies:
            G_copy = G.copy()
            G_remove = remove_nodes(G_copy, strategy, int(G.number_of_nodes() *
immunization_fraction))
            initial_infecteds2 = random.sample(list(G_remove.nodes()), 5)
            t, S, I = EoN.fast_SIS(G_remove, tau, gamma,tmax=50,
initial_infecteds=initial_infecteds2)
            p = I[-1] / I0[-1]
            data.append(p)
        new_row = pd.DataFrame([data], columns=columns)
        df = df.append(new_row, ignore_index=True)
    df.to_excel('C:\\Users\\范春\\Desktop\\week5\\simulation_results.xlsx', index=False)
def main():
    filePath = "C:\\Users\\范春\\Desktop\\week5\\twenty_years_edgelist.xlsx"
    G = network(filePath)
    rich_club(G)
    r = degree_degree(G)
    print(r)
    SIS_SIR(G)
    question4(G)

if __name__=='__main__':
    main()
```