Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный исследовательский университет ИТМО»
(Университет ИТМО)

Факультет Инфокоммуникационных технологий (ИКТ)

Образовательная программа Мобильные и сетевые технологии

О Т Ч Е Т

по Лабораторной работе

Дисциплина: Методы сетевого анализа.

Специальность: 09.03.03 Прикладная информатика.

Выполнил:

Балцат К. И.,

студент группы К33401

Санкт-Петербург
2023

ЗАДАНИЕ

Можно использовать один из графов, полученных в ходе предыдущей работы, или создать новый. Добавьте изображение графа в отчёт. Данная работа посвящена исследованию социального графа посредством вычисления различных метрик. Некоторые метрики были затронуты в предыдущей работе и не дублируются в этой, поэтому в случае создания нового графа повторите также пункты 6 и 7 из лабораторной работы №1.

Найдите наиболее авторитетных пользователей

В качестве метрики авторитетности пользователя используйте центральность - число минимальных кратчайших путей между любыми двумя его “друзьями” или “собеседниками”, проходящих через него.

Вычислите плотность социального графа

Плотность графа - это отношение реального числа связей в графе к максимально возможному в неориентированном графе с тем же числом вершин.

Является ли граф связным? Что это означает применительно к исследуемому социальному графу?

Связный граф - граф, содержащий ровно одну компоненту связности. Это означает, что между любой парой вершин этого графа существует как минимум один путь.

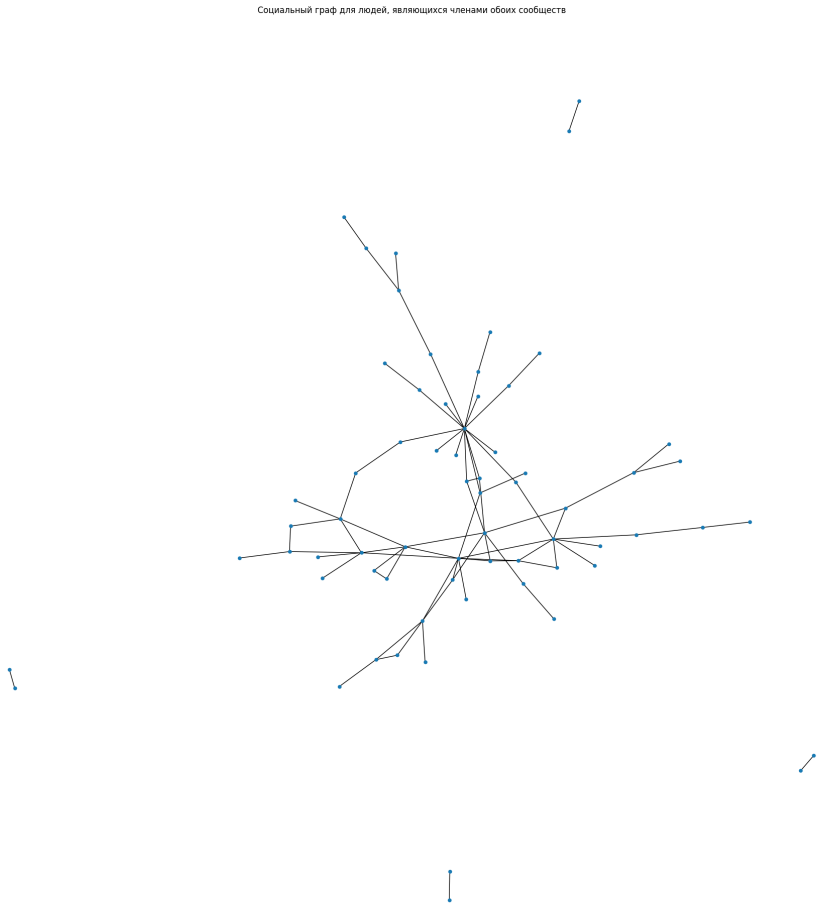
Рассчитайте максимальное, минимальное и среднее значение степени узлов графа

Рассчитайте модулярность графа

Модулярность - это скалярная величина из отрезка [−1, 1], которая количественно описывает неформальное определение структуры сообществ.

ВЫПОЛНЕНИЕ

Вывод графов из Лабораторной работы 1.



Поиск наиболее авторитетных пользователей по метрике центральности посредничества. Вверху для ориентированного графа, внизу – неориентированного.

[7]

	ID	Centrality
18	155846028	0.20535742035742036
33	144735090	0.16442501942501941
19	463569168	0.15987567987567988
1	5379200	0.13185703185703188
17	155218497	0.11613442113442111
22	69518491	0.1003108003108003
5	227014	0.08158508158508158
12	435009255	0.06705516705516706
8	70936303	0.05975135975135976
30	157040051	0.05205128205128205

```
[8] b = nx.betweenness_centrality(G1)          # Для поиска наиболее авторитетных пользователей на неориентированном графе мы выбрали центральность посредничества
    result = b.items()                         # Достаём ID пользователей и полученные для них значения центральности и
    data = list(result)                        # помещаем их в список,
    arrayG1 = np.array(data)                   # после чего помещаем их в массив
    for i in range(len(arrayG1)):              # Создаём цикл for, который проходит по каждому значению центральности в нём и
      arrayG1[i][1] = Decimal(arrayG1[i][1])   # переводит это значение в формат float без использования e^(-x)
```

```
[9] df = pd.DataFrame(arrayG1)                                                  # Создаём датасет с ID и изменёнными значениями центральности
    df = df.rename(columns={df.columns[0]: 'ID', df.columns[1]: 'Centrality'})  # Переименовываем название столбцов
    df = df.sort_values(by='Centrality', ascending=False)                       # Сортируем по мере уменьшения центральности
    df[0:10]                                                                    # Выводим топ-10
```

	ID	Centrality
33	144735090	0.3898290598290598
18	155846028	0.27924630924630933
1	5379200	0.18975135975135984
19	463569168	0.1567987567987568
57	709276449	0.1431235431235431
22	69518491	0.11836829836829837
17	155218497	0.10264957264957267
5	227014	0.1020979020979021
61	126346259	0.10069930069930069
12	435009255	0.08958818958818958

Вычисление плотности социального графа.

Вычисление плотности социального графа

1) https://networkx.org/documentation/stable/reference/generated/networkx.classes.function.density.html

```
[10] nx.density(G)  # Плотность ориентированного графа
```

```
0.03143374038896427
```

```
[11] nx.density(G1)  # Плотность неориентированного графа
```

```
0.03573043871551334
```

Проверка графа на связность.

Проверка графа на связность

Для ориентированного графа не работает
1) https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.components.is_connected.html

```
[12] nx.is_connected(G1)  # Проверка неориентированного графа на связность
```

```
False
```

Расчёт максимального, минимального и среднего значения степени узлов графа.

Расчёт максимального, минимального и среднего значения степени узлов графа

1) https://networkx.org/documentation/stable/reference/classes/generated/networkx.Graph.degree.html

```
[13] degrees = [val for (node, val) in G.degree()]                          # Достаём значения степеней узлов ориентированного графа и помещаем их в массив
     print("Максимальная степень узла ориентированного графа:", max(degrees))  # Выводим максимальное значение степени узла
     print("Минимальная степень узла ориентированного графа:", min(degrees))   # Выводим минимальное значение степени узла
     print("Средняя степень узла ориентированного графа:", mean(degrees))      # Выводим среднее значение степени узла
```

```
Максимальная степень узла ориентированного графа: 19
Минимальная степень узла ориентированного графа: 1
Средняя степень узла ориентированного графа: 4.149253731343284
```

```
[14] degrees = [val for (node, val) in G1.degree()]                         # Достаём значения степеней узлов неориентированного графа и помещаем их в массив
     print("Максимальная степень узла ориентированного графа:", max(degrees))  # Выводим максимальное значение степени узла
     print("Минимальная степень узла ориентированного графа:", min(degrees))   # Выводим минимальное значение степени узла
     print("Средняя степень узла ориентированного графа:", mean(degrees))      # Выводим среднее значение степени узла
```

```
Максимальная степень узла ориентированного графа: 14
Минимальная степень узла ориентированного графа: 1
Средняя степень узла ориентированного графа: 2.3582089552238807
```

Расчёт модулярности графа.

Расчёт модулярности графа

Не работает для ориентированного графа
1) https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.community.quality.modularity.html
2) https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.community.label_propagation.label_propagation_communities.html
3) https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.community.louvain.louvain_communities.html

```
[15] nx_comm.modularity(G1, nx_comm.label_propagation_communities(G1))  # Расчёт модулярности неориентированного графа с разбиением на сообщества с помощью метода библио
```

```
0.524755648133312
```

```
[16] partition = nx.community.louvain_communities(G1)  # Метод Лувейна для разбиения на сообщества (его мы использовали в лабораторной работе №1)
     nx_comm.modularity(G1, partition)                 # Расчёт модулярности неориентированного графа с разбиением на сообщества с помощью метода Лувейна
```

```
0.647892965870854
```

ВЫВОД

Я выполнил лабораторную работу и на практике освоил методы анализа сетей.