АҚПАРАТТЫҚ ЖӘНЕ КОММУНИКАЦИЯЛЫҚ ТЕХНОЛОГИЯЛАР
ИНФОРМАЦИОННО-КОММУНИКАЦИОННЫЕ ТЕХНОЛОГИИ
INFORMATION AND COMMUNICATION TECHNOLOGIES

DOI 10.51885/1561-4212_2024_2_165
МРНТИ 20.53.19

**Ш.Ж. Мусиралиева[1], Г.Б. Байспай[2], Е. Абайұлы[3], Р.К. Оспанов[4], Д.Ж. Агабеков[5]**
Казахский национальный университет имени аль-Фараби, г. Алматы, Казахстан
[1]*E-mail: mussiraliyevash@gmail.com*
[2]*E-mail: gulshat.bgb2@gmail.com**
[3]*E-mail: erulan_97@mail.ru*
[4]*E-mail: ospanov.ruslan.k@gmail.com*
[5]*E-mail: daagabekov@gmail.com*

# РАЗРАБОТКА АЛГОРИТМА И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ГРАФИЧЕСКОЙ ВИЗУАЛИЗАЦИИ СВЯЗЕЙ ВОВЛЕЧЕННЫХ ПОЛЬЗОВАТЕЛЕЙ

# ҚАТЫСУШЫ ПАЙДАЛАНУШЫЛАРДЫҢ БАЙЛАНЫСТАРЫН ГРАФИКАЛЫҚ ВИЗУАЛИЗАЦИЯЛАУ АЛГОРИТМІ МЕН БАҒДАРЛАМАЛЫҚ ҚАМТАМАСЫЗ ЕТУДІ ӘЗІРЛЕУ

# DEVELOPMENT OF AN ALGORITHM AND SOFTWARE FOR GRAPHIC VISUALIZATION OF RELATIONSHIPS OF INVOLVED USERS

***Аннотация.*** *В последние годы наблюдается всплеск исследовательского интереса к выявлению преступников через аккаунты в социальных сетях и анализу визуализации связей вовлеченных пользователей, поскольку преступники активно используют социальные сети, а количество призывов к экстремизму через социальные сети растет каждый год. В данной работе рассматривается актуальная проблема использования методов анализа социальных сетей и идентификации на основе общедоступных данных профилей пользователей. Рассматривается разработка алгоритма и программное обеспечение графической визуализации связей вовлеченных пользователей для выявления узлов распространения криминальной информации в социальных сетях. Представлен обзор существующих решений и подходов, а также предложен собственный алгоритм графической визуализации. Результаты эксперимента показывают высокую точность определения вовлеченных пользователей.*

***Ключевые слова:*** *визуализация; вовлеченные пользователи; алгоритм графической визуализации; анализ социальных сетей; связи вовлеченных пользователей.*

***Аңдатпа.*** *Соңғы жылдары қылмыскерлерді әлеуметтік желідегі аккаунттар арқылы анықтауға және оған қатысы бар пайдаланушылардың байланысының визуализациясын талдауды зерттеуге қызығушылық арта түсуде, өйткені қылмыскерлер әлеуметтік желілерді белсенді пайдаланады, ал әлеуметтік желілер арқылы экстремизмге шақырулар саны жыл сайын артып келеді. Бұл мақалада жалпыға қолжетімді пайдаланушы профилі деректеріне негізделген әлеуметтік желіні талдау және сәйкестендіру әдістерін пайдаланудың өзекті мәселесі қарастырылды. Әлеуметтік желілерде қылмыстық ақпаратты тарату түйіндерін анықтау үшін негізгі пайдаланушылардың байланыстарын графикалық визуализациялау алгоритмі мен бағдарламалық қамтамасыз етуді әзірлеу қарастырылды. Мақалада қолда бар шешімдер мен тәсілдерге шолу жасалды, сонымен қатар авторлар өзінің графикалық визуализация алгоритмін ұсынды. Эксперимент нәтижелері негізгі пайдаланушыларды анықтаудың жоғары дәлдігін көрсетті.*

***Түйін сөздер:*** *визуализация; негізгі пайдаланушылар; графикалық визуализация алгоритмі; әлеуметтік желілерді талдау; негізгі пайдаланушылардың байланыстары.*

***Abstract.*** *In recent years, there has been a surge in research interest in identifying criminals through social media accounts and analyzing the visualization of the connections of involved users, as criminals actively use social networks, and the number of calls for extremism through social networks is growing every year. This paper deals with the actual problem of using social network analysis and identification methods based on publicly available user profile data. The development of an algorithm and software for graphical*

*visualization of the connections of involved users to identify nodes for the dissemination of criminal information in social networks is considered. It provides an overview of existing solutions and approaches, and also proposes its own graphical visualization algorithm. The results of the experiment show a high accuracy in determining the involved users.*

***Keywords:*** *visualization; involved users; graphic visualization algorithm; social network analysis; connections of involved users.*

*Введение.* В современном мире социальные сети являются одним из основных способов общения, основной сферой самовыражения. Социальные сети обеспечивают немедленную и глобальную обратную связь по любому сообщению в виде комментариев, которые формируют общественное мнение о высказанных взглядах. Воздействие сообщения в Интернете зачастую создается потоком комментариев, которые можно использовать для различных целей: получения информации, времяпрепровождения, знакомств, поиска сообществ со схожими интересами, распространения политических идей и даже продвижения радикального контента и экстремистских идеологий, привлечения новых последователей. Молодежь занимается преступной деятельностью, а в результате вовлечения молодежи в различные суицидальные сообщества фиксируется множество случаев самоубийств. Существует множество сообществ, привлекающих молодежь к экстремистским и террористическим движениям. Подобная пропаганда вызывает большой интерес не только в научном сообществе, но и в органах национальной безопасности, поскольку все это наносит вред как гражданам, так и обществу.

В этой статье показано, как взаимодействие между пользователями определяет функции сообщества и как каждое сообщество создает свою уникальную структуру. В последние годы графы играют важную роль в аналитике больших данных и социальных сетей. Обнаружение сообществ широко используется для получения групп узлов, которые тесно взаимодействуют и тесно связаны друг с другом, что помогает получить положительные результаты анализа социальных сетей. Например, если узлы в сети представляют профили пользователей в социальной сети, а ребра представляют взаимодействие между этими узлами, то сообщество узлов – это группа пользователей, которые тесно связаны между собой и имеют некоторые схожие характеристики [1].

В статье представлен обзор метода анализа социальных сетей и методов определения влияния пользователей социальных сетей. Разработана методика поиска влиятельных пользователей, разработано программное обеспечение для сбора и анализа данных социальных сетей. Созданы пользовательские графы для постов в группах, графы для комментариев под постами. Разработанные методы были апробированы на готовых наборах данных, а также на собственных собранных данных из социальных сетей. Полученные результаты подтверждают корректность работы устройства и достоверность результатов средним значением 91 %.

*Литературный обзор.* Проблема визуализации и анализа данных является крайне важной в наше время перенасыщения информацией. Важно уметь пользоваться и автоматическими средствами анализа, и инструментами визуализации для ручного анализа информации. В нашем случае требуется уметь находить наиболее влиятельных людей в группах с заведомо экстремистскими текстами.

В бесчисленных ландшафтах онлайн-социальных сетей (OSN) и их последующем влиянии на социальные конструкции исследователи глубоко углубились в анализ закономерностей [2], структур [3] и поведенческой динамики [4], присущих этим цифровым платформам. Предпринята попытка представить синтез соответствующих работ, которые проложили путь в области анализа OSN, киберкриминологии, выявления влияния и распространения информации в цифровых сферах, обеспечивая контекстуальную основу, на которой строятся текущие исследования.

Для анализа и моделирования сетей применялся метод анализа социальных сетей (АСС). АСС – это инструмент количественного анализа социальных сетей, используемый для выявления и понимания взаимоотношений между пользователями. Метод визуально отображает данные, поэтому исследователи мира могут видеть поведенческие взаимосвязи на микроуровне и закономерности на сетевом уровне [5]. Метод АСС может использовать такие данные, как свидетельство сотрудничества, социальные сети, например, лайки и репосты и т.д. На данный момент этот метод используется в разных целях, например, организационные процессы, маркетинг, кража личностей и разные виды кибератак [6].

Авторы разработали различные методы интеллектуального анализа данных и текстового поиска для анализа ссылок, контент анализа, анализа веб-метрик, сентимент анализа, анализа авторства и видеоанализа. Они разработали специализированные методы для разметки и сбора мультимедийных файлов, контента веб-сайтов и форумов и различные методы SNA для изучения взаимосвязи веб-сайтов и форумов. Авторы разработали несколько методов кластеризации для визуализации их связей [7].

В работе [8] авторы представляют метод составления короткого списка влиятельных членов преступных организаций и определения их важных каналов связи. По мнению авторов, чаще всего под арест попадают мелкие преступники, а преступники более высокого ранга избегают наказания. Авторы предлагают новую систему судебно-медицинской экспертизы под названием IICCC, которая помогает выявить влиятельных преступников при помощи короткого списка каналов связи. Авторы используют термин «критический канал связи» для обозначения части «важного канала связи», которая имеет высокую и стабильную скорость потока информации относительно других путей в сети. Они вычислили $\chi 2$ (Chi Squared), Betweenness centrality с помощью своего алгоритма. Алгоритмы были оценены с использованием трех реальных наборов данных: набора данных Caviar, набора данных электронной почты Enron и набора данных 9/11 Кребса. Авторы преобразовали каждый набор данных в сеть, отражающую попытки связи между лицами, обвиняемыми в инцидентах, сравнили влиятельные узлы, возвращаемые каждой системой, с соответствующими узлами, возвращаемыми стандартными метриками Betweenness, Closeness, Out Degree, In Degree centrality, и вычислили значения отзыва (Recalls), точности (Precisions) и F (F-value) для каждой системы в отношении каждой метрики централизации сети и каждой метрики качества.

В работе [9] авторы обсудили возможность обнаружения влиятельных узлов в неявных социальных сетях с использованием многозадачных моделей гауссовской копулы. Многие существующие методы основываются на предположении, что структура сети полностью известна априори. Однако во многих приложениях сетевая структура недоступна для объяснения лежащего в основе феномена распространения информации. Чтобы решить проблему анализа распространения информации при неполном знании сетевой структуры, авторы разработали многозадачную модель линейного влияния низкого ранга. Используя взаимосвязи между заражениями, их подход может одновременно прогнозировать объем (прогнозирование временных рядов) для каждого заражения (или темы) и автоматически определять наиболее влиятельные узлы для каждого заражения.

В статьях [10, 11] авторы исследуют выявление ключевых игроков в онлайн-группах активистов в социальной сети Facebook. Они используют методы анализа социальных сетей, чтобы понять динамику взаимодействия между пользователями в группе активистов на базе Facebook. Авторы использовали метрики АСС, такие как Network Diameter, Network Density, Centrality, Average Degree.

В исследовании [12], авторы написали про обнаружение джихадизма в социальных сетях с использованием методов больших данных, поддерживаемых графами и нечеткой кластеризацией. Исследование, представленное в этой статье, сосредоточено на анализе

сообщений Twitter с целью выявления лидеров, организующих террористические сети, и их последователей. В исследовании данные собраны через Twitter API. Предлагается архитектура больших данных для анализа сообщений в реальном времени с целью классификации пользователей по различным параметрам, таким как уровень активности, способность влиять на других пользователей и содержание их сообщений. Графики использовались для анализа того, как сообщения распространяются по сети, и это включает изучение подписчиков на основе ретвитов и общего воздействия на других пользователей.

*Материалы и методы исследования.* Для разработки алгоритма и ПО графической визуализации связей вовлеченных пользователей социальной сети в первую очередь потребуется определиться с выбором сообществ. Информации и контингент данных групп используются при создании визуализации. Чтобы сделать хорошую и анализируемую визуализацию, которая показывает связи вовлеченных пользователей, стояла задача выбрать тот тип социальной сети, которая активно используется среди казахоязычной аудитории. В результате анализа была выбрана социальная сеть ВКонтакте. Датасет активистов состоит из религиозных постов групп социальной сети ВКонтакте. Следующим шагом является составление списков групп. Для работы были выбраны те группы, которые были под запретом на территории Республики Казахстан. В результате поиска были выявлены 76 групп, распространение которых запрещено на территории Республики Казахстан.

*Этапы визуализации.* Первый этап визуализации – в качестве графов было получено пересечение пользователей по идентификатору в разных сообществах для получения общих аудиторий между группами. Здесь применялась библиотека matplotlib.

*1 шаг. Принцип построения графа.* Здесь авторы работали с библиотекой NetworkX. Данная библиотека хорошо подходит для визуализации и анализа графов.

*2 шаг. Визуализация постов в группе (количество лайков, количество репостов, количество комментариев).* На вход подается словарь, где ключом является идентификатор поста, публикации, а значением – количество лайков, репостов, комментариев. В данном случае вершинами являются посты в группе. Весами является количество лайков, репостов, комментариев. Весами для ребер является количество комментирующих и их пересечение по идентификатору. Это будет означать, что у людей, лайкнувших и прокомментировавших один и тот же пост, пересекаются интересы.

Для реализации алгоритма были рассмотрены методы VK API. В первом этапе была получена данная визуализация. Авторы получили пересечение пользователей по идентификатору в разных сообществах для получения общих аудиторий между группами. Для тестирования разработанных алгоритмов были выбраны открытые группы в случайном порядке (рис. 1).

Далее, изучив данные задачи, были сделаны следующие работы: визуализация связей между группами, связей между пользователями групп, связей между постами групп и визуализация связей групп, постов и пользователей в виде дерева.

Связь между группами осуществляется следующим образом: для всех групп получаются пользователи, подписанные на эту группу; количество таких пользователей является весом вершины (группы), количество общих между группами пользователей является весом ребра между вершинами (группами).

Связь между пользователями групп осуществляется следующим образом: получаются все посты из всех групп; для каждого поста получаются комментировавшие пользователи; количество таких комментариев от каждого пользователя является весом вершины (пользователя), комментарий и ответ на этот комментарий образуют связь, таким образом, количество таких ответов между пользователями является весом ребра между вершинами (пользователями).

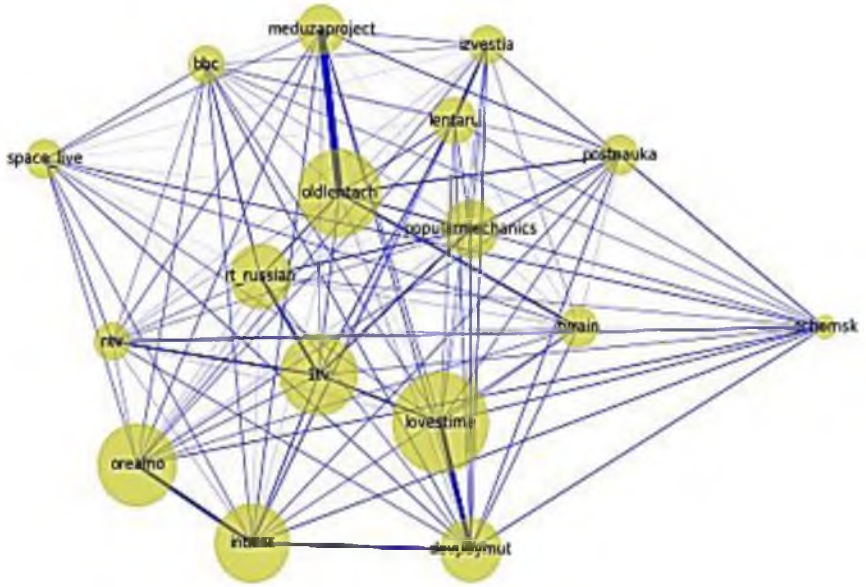


**Рисунок 1.** Визуализация пересечения аудитории пользователей между группами

Связь между постами осуществляется следующим образом: для каждой группы достаются посты в определенном временном промежутке; в качестве весов выбираются лайки или комментарии; количество лайков или комментариев является весом вершины (поста), количество лайков или комментариев от одного пользователя на обоих постах является весом ребра между вершинами (постами). В дереве корнем является группа, дети которой являются постами, дети которых являются пользователями. Для пользователей можно получить информацию по имени, фамилии и аватару.

*Результаты и их обсуждения. Модуль ПО графической визуализации связей вовлеченных пользователей.* Для анализа и визуализации данных существует огромное множество методов, однако было решено выбрать анализ и визуализацию данных, используя графы и алгоритмы на графах. В нашем случае визуализация и анализ данных состоит из 4 частей: сбор данных, составление графа из данных, анализ графа и визуализация графа. Таким образом, авторами проводились визуализация и анализ данных постов групп сети ВКонтакте. При этом составлены были два различных графа.

*Построение графа пользователей на основе постов в группах.* Сбор данных. При вводе списка групп через запятую и указании временного отрезка, в котором содержатся нужные посты из этих групп, можно получить граф пользователей. Здесь при сборе данных собирались посты указанных групп в определенном временном промежутке, а также информация о пользователях, оставивших комментарии. В частности, для каждой группы в списке запускался метод get_posts_if_reply, который состоит из следующих этапов:

1. Получение всех постов группы методом get_posts. Метод get_posts заключается в том, чтобы с помощью алгоритма бинарного поиска найти в списке постов, отсортированных по убыванию даты публикации, пост, начиная с которого нужные посты встречаются подряд. Таким образом мы обеспечиваем лучшую производительность, чем при простом получении всех постов, начиная с последнего. В среднем алгоритм потратит на нахождение нужного поста $log_2 n$ времени, где $n$ – количество всех постов группы.

2. Для каждого поста, после получения всех постов, получаются все его комментарии методом get_post_weights_if_reply (рис. 2). Для каждого комментария также получаются его подкомментарии (т.е. ветки комментариев). Код get_posts_if_reply приведен ниже:

```
def get_post_weights_if_reply(self, postId, groupId):
    weightsList = []
        i = 0
while True:
        addedToWeights = self.vk_api.wall.getComments(owner_id=-groupId, post_id=postId,
count=100,                                  offset=i * 100, v=5.92)['items']
        weightsList = weightsList + addedToWeights
    if len(addedToWeights) < 100:
        break
    i += 1
comments = {}
for comment in weightsList:
    nested_comments = self.get_thread(comment, postId, groupId)
    for key in nested_comments:
        if key in comments:
            comments[key] += nested_comments[key]
        else:
            comments[key] = nested_comments[key]
return comments
def get_posts_if_reply(self, group, start_date, end_date):
    posts = self.get_posts(group, start_date, end_date, '')
    group = self.vk_api.groups.getById(group_ids=group, v=5.92)
    for i in range(len(posts)):
        try:
            weight = self.get_post_weights_if_reply(postId=posts[i]['id'], groupId=group[0]['id'])
        except:
            weight = {}
        posts[i]['comments'] = weight
    return posts
def get_post_weights_if_reply(self, postId, groupId):
    weightsList = []
        i = 0
while True:
        addedToWeights = self.vk_api.wall.getComments(owner_id=-groupId, post_id=postId,
count=100,                                  offset=i * 100, v=5.92)['items']
        weightsList = weightsList + addedToWeights
    if len(addedToWeights) < 100:
        break
    i += 1
comments = {}
for comment in weightsList:
    nested_comments = self.get_thread(comment, postId, groupId)
    for key in nested_comments:
        if key in comments:
            comments[key] += nested_comments[key]
        else:
            comments[key] = nested_comments[key]
return comments
def get_posts_if_reply(self, group, start_date, end_date):
    posts = self.get_posts(group, start_date, end_date, '')
    group = self.vk_api.groups.getById(group_ids=group, v=5.92)
    for i in range(len(posts)):
        try:
            weight = self.get_post_weights_if_reply(postId=posts[i]['id'], groupId=group[0]['id'])
```

```
        except:
            weight = {}
        posts[i]['comments'] = weight
    return posts
```

Затем на основе этих данных строится граф. Планируется использование базы данных, чтобы запрос визуализации графа не занимал много времени при повторном вызове.

*Составление графа на основе комментариев под постами.* Здесь пользователь является вершиной, а суммарное количество комментариев под постами – его весом. Весом ребер между вершинами является количество взаимодействий пользователей в комментариях (т.е. количество ответов одного пользователя другому или участие пользователя в ветке комментариев другого пользователя). Соответственно, если пользователи между собой не взаимодействовали, то ребра не будет. Код составления графа приведен ниже:

```
def make_graph_users_json(self, posts):
    users_dict = users_dict_from_posts(posts)
    graph = {"nodes": [], "links": []}
    users = self.parser.get_users_by_ids(list(users_dict.keys()))
    for user_id, user in users_dict.items():
        cnt = user['count']
        my_user = None
        for usr in users:
            if usr['id'] == user_id:
                my_user = usr
        if my_user is not None:
            txt = my_user['first_name'] + ' ' + my_user['last_name']
            url = 'https://vk.com/' + my_user['domain']
        graph['nodes'].append(
            {"id": user_id, "title": txt, "url": url, "size": cnt, "group": random.randint(1,10)})
        for link_id, link_cnt in user['links'].items():
        graph['links'].append({"source": user_id, "target": link_id, "size": link_cnt})
        return graph
```

Для анализа графа граф был построен с помощью библиотеки NetworkX. При анализе графа были посчитаны различные свойства графа, такие как: Degree Centrality, Closeness Centrality, Betweenness Centrality, Chi Squared, количество вершин, количество ребер и плотность графа, которые можно увидеть снизу от самого графа. Большинство свойств можно получить из библиотеки NetworkX, но получение Chi Squared было реализовано вручную с помощью следующего алгоритма:

- для каждой вершины находятся все пути от соседей этой вершины до нахождения аттрактора, при этом на каждом шаге при выборе следующей вершины в пути выбирается вершина с наибольшим значением betweenness centrality;
- для каждой вершины находится максимальное значение на каждом уровне путей (под уровнем пути подразумевается глубина пути, т.е. сама вершина имеет уровень 0, а сосед имеет уровень 1);
- для каждого пути находится его betweennes centrality, который является суммой betweenness centrality его вершин;
- для каждого пути находится его локальный Chi Squared;
- суммируется Chi Squared всех путей, и сумма присваивается вершине.

На основе этих свойств графа можно находить наиболее влиятельные вершины графа, например, выбрав пользователей с наибольшим значением Chi Squared, Closeness Centrality, Betweenness Centrality. Код нахождение свойств графа приведен ниже:

```
def fill_json_with_data(self, json_of_graph, graph):
    betweenness_centrality, closeness_centrality, chi_squared, significant_paths =
data_from_graph(graph)
    json_of_graph["chi_squared"] = list({k: [get_user_name(self.parser.get_users_by_ids([k])),
v] for k, v in
                                         sorted(chi_squared.items(), key=lambda item: item[1],
reverse=True)[:40]}.items())
    json_of_graph["degree_centrality"] = list({k:
[get_user_name(self.parser.get_users_by_ids([k])), v] for k, v in
sorted(nx.degree_centrality(graph).items(), key=lambda item: item[1],
reverse=True)[:40]}.items())
    json_of_graph["closeness_centrality"] = list(
        {k: [get_user_name(self.parser.get_users_by_ids([k])), v] for k, v in
sorted(closeness_centrality.items(),
key=lambda item: item[1],
reverse=True)[:40]}.items())
    json_of_graph["betweenness_centrality"] = list(
        {k: [get_user_name(self.parser.get_users_by_ids([k])), v] for k, v in
sorted(betweenness_centrality.items(),
key=lambda
item: item[1],                                                    reverse=True)[
:40]}.items())
    json_of_graph["number_of_nodes"] = graph.number_of_nodes()
    json_of_graph["number_of_edges"] = graph.number_of_edges()
    json_of_graph["density"] = nx.density(graph)
```

Для проверки работоспособности метода авторы работали с разными датасэтами, такими как Krebs's 9/11 dataset, Caviar dataset и How ISIS Uses Twitter [13-15].

Наиболее влиятельные узлы в наборе данных Caviar [14] были обозначены как узлы N1, N12 и N3. За торговлю гашишем отвечал член банды в лице Node N1. Член банды в лице N12 отвечал за торговлю кокаином. Член банды, представленный узлом N3, был посредником между N1 и N12, а также между ними и лицами, не занимающимися торговлей. Поэтому пути, исходящие от узлов N1, N12 и N3, мы рассматривали как критичные для наземных сетей каналы связи (табл. 1).

**Таблица 1.** Проверка эффективности метода с помощью набора данных Caviar

| Chi squared | | Betweenness centrality | | Closeness centrality | Number of neighbours | |
|---|---|---|---|---|---|---|
| 10,78 | 1 | 0,63 | 1 | 0,67 | 1 | 60 |
| 10,23 | 12 | 0,29 | 3 | 0,53 | 12 | 28 |
| 8,16 | 87 | 0,11 | 12 | 0,52 | 3 | 27 |
| 7,61 | 76 | 0,11 | 76 | 0,50 | 87 | 16 |
| 7,01 | 3 | 0,10 | 41 | 0,48 | 76 | 15 |
| 6,21 | 37 | 0,07 | 87 | 0,48 | 37 | 11 |
| 5,79 | 41 | 0,06 | 89 | 0,47 | 41 | 11 |

При проверке эффективности метода на датасетах Caviar и How ISIS Uses Twitter результаты оказались с точностью 91 %. При тестировании метода на наборе данных Кребса о событиях 11 сентября показатель Хи-квадрат лидера заговора Мохамеда Атты составил 1,194, что является средним показателем. Авторы хотят повысить точность данных при работе с этим методом и на данный момент продолжают свои исследования.

*Алгоритм визуализации графа.* Сам граф был визуализирован так называемым силовым алгоритмом визуализации графа, который обеспечивает более плотное расположение связанных вершин. При этом вершины графа можно тянуть и перемещать, рассматривая индивидуальные нужные вершины. На вершинах пишется имя пользователя, указанное в социальной сети ВКонтакте. При наведении на вершину можно увидеть ссылку на личную страницу данного пользователя. Визуализация была выполнена в формате веб-приложения. Веб-приложение использует html, css и javascript для отрисовки графа. Для визуализации была использована библиотека d3.js. В качестве формата отображения графа был выбран svg, так как он позволяет приближать и отдалять изображение во сколько угодно раз, при этом не теряя в качестве. При использованиях метода графической визуализации и в идентификации вовлеченных пользователей авторы исследовали труды ученых [16-18].

В качестве бэкэнда веб-приложения используется фреймворк Django на языке Python. На html-странице был размещен элемент svg, в котором затем отрисовывался граф с помощью скрипта javascript с использованием d3.js. Еще на этапе формирования html-страницы Django передает необходимые данные, которые авторы сформировали после получения данных. Далее подбирались различные параметры, такие как цвет, сила притяжения силового алгоритма, начальная дистанция вершин графа друг от друга, центр графа в элементе svg (рис. 2).

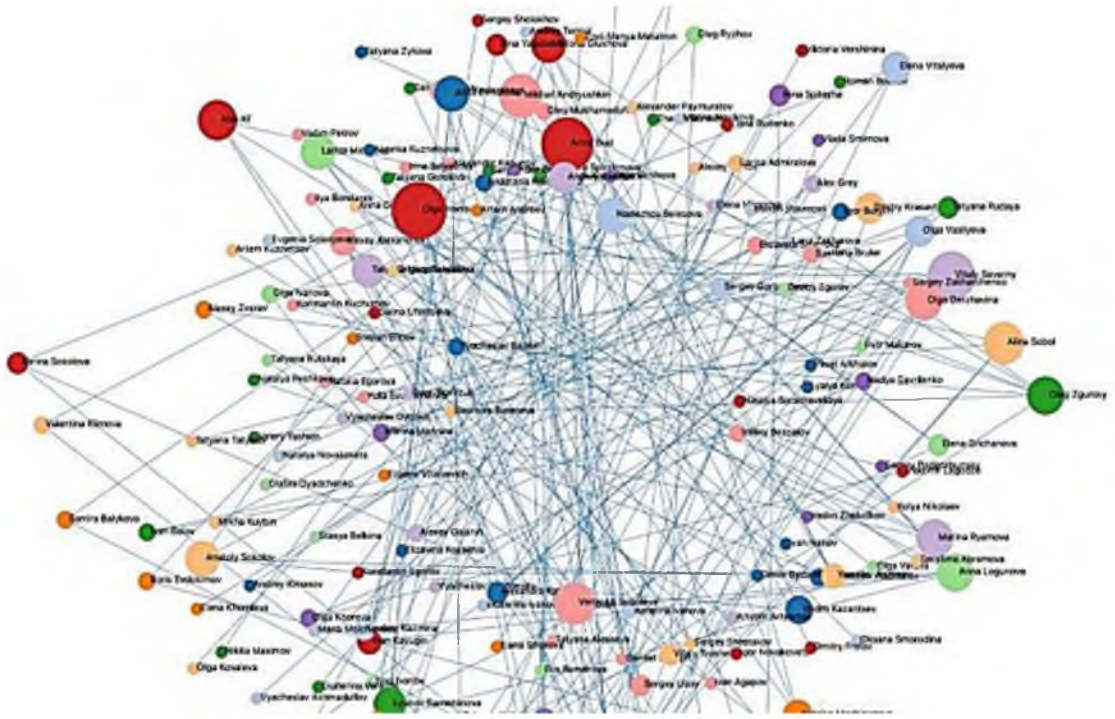

**Рисунок 2.** Визуализация графа пользователей

*Граф постов.* Сбор данных. При вводе списка групп через запятую и указании временного отрезка, в котором содержатся нужные посты из этих групп, можно получить граф постов. Здесь при сборе данных собирались посты указанных групп в определенном временном промежутке, а также информация о пользователях, оставивших комментарии или лайки. В частности, для каждой группы в списке запускался метод get_posts, который состоит из следующих этапов:

– Получение всех постов группы. Метод get_posts заключается в том, чтобы с помощью алгоритма бинарного поиска найти в списке постов, отсортированных по убыванию даты публикации, пост, начиная с которого нужные посты встречаются подряд. Таким образом обеспечиваем лучшую производительность, чем при простом получении всех постов, начиная с последнего. В среднем алгоритм потратит на нахождение нужного поста $log_2 n$ времени, где n - количество всех постов группы;

– После получения всех постов, для каждого поста получаются все его комментарии или лайки методом get_post_weights. Затем на основе этих данных строится граф. Планируется использование базы данных для того, чтобы запрос визуализации графа не занимал много времени при повторном вызове;

– Составление графа из данных. Пост здесь является вершиной, а суммарное количество комментариев или лайков под ним – его весом. Весом ребер между вершинами является количество одинаковых пользователей у постов, которые лайкнули запись или оставили комментарий. Соответственно, если общих пользователей у постов нет, то ребра не будет. Код составления графа приведен ниже:

```
def make_graph_users_json(self, posts):
    users_dict = users_dict_from_posts(posts)
    graph = {"nodes": [], "links": []}
    users = self.parser.get_users_by_ids(list(users_dict.keys()))
        for user_id, user in users_dict.items():
        cnt = user['count']
        my_user = None
        for usr in users:
            if usr['id'] == user_id:
                my_user = usr
        txt = my_user['first_name'] + ' ' + my_user['last_name']
        url = 'https://vk.com/' + my_user['domain']
        graph['nodes'].append(
            {"id": user_id, "title": txt, "url": url, "size": cnt, "group": random.randint(1, 10)})
        for link_id, link_cnt in user['links'].items():
            graph['links'].append({"source": user_id, "target": link_id, "size": link_cnt})
    return graph
```

*Анализ графа.* Для анализа граф был построен граф с помощью библиотеки NetworkX. При анализе графа были посчитаны различные его свойства, такие как: Degree Centrality, Closeness Centrality, Betweenness Centrality, количество вершин, количество ребер и плотность графа, которые можно увидеть справа от самого графа. На этом графе не находится Chi Squared. Здесь не нужно находить самых влиятельных пользователей, требуется лишь вывести дополнительную полезную информацию пользователю.

Затем нам нужно создать несколько элементов <g> внутри элемента svg. Эти элементы и будут обозначать будущие вершины и ребра графа. Для того, чтобы разграничить вершины и ребра друг от друга, нужно пометить их классами nodes и links соответственно. Затем каждый элемент заполняется соответствующими ему данными и можно задать для элементов дополнительные параметры. Например, в качестве дополнительных параметов авторы задали радиус вершин графа или толщину ребер графа (рис. 3).

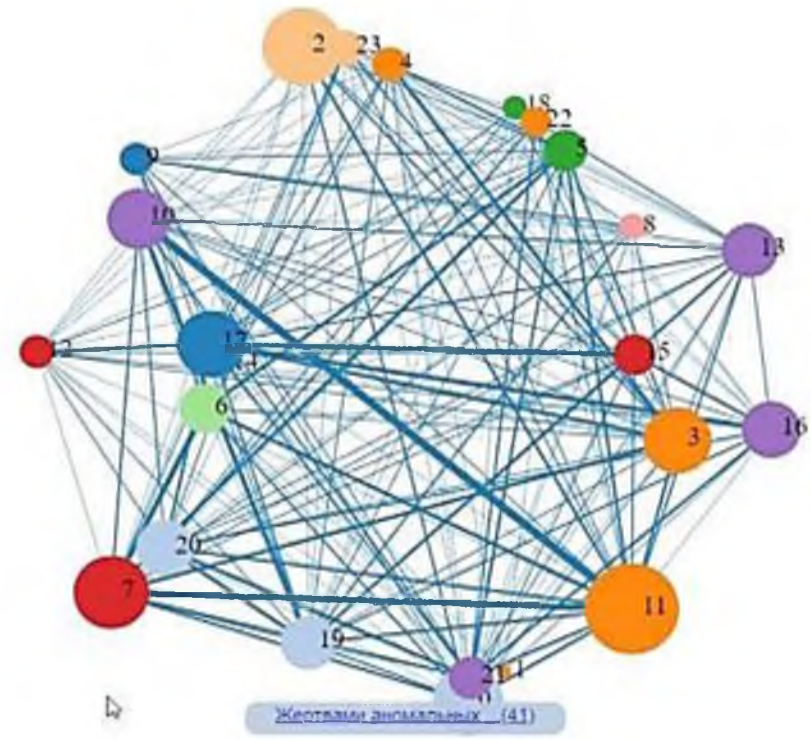


**Рисунок 3.** Визуализация графа постов

*Дерево постов и пользователей.* Сбор данных. При вводе группы и указании временного отрезка, в котором содержатся нужные посты этой группы, можно получить дерево постов пользователей. Здесь при сборе данных собирались посты указанной группы в определенном временном промежутке, а также информация о пользователях, оставивших комментарии или лайки. В частности, для группы запускался метод get_posts, который состоит из следующих этапов:

1. Получение всех постов группы. Метод get_posts заключается в том, чтобы с помощью алгоритма бинарного поиска найти в списке постов, отсортированных по убыванию даты публикации, пост, начиная с которого нужные посты встречаются подряд. Таким образом мы обеспечиваем лучшую производительность, чем при простом получении всех постов, начиная с последнего. В среднем алгоритм потратит на нахождение нужного поста $log_2 n$ времени, где n - количество всех постов группы.

2. Получение для каждого поста всех его комментариев или лайков методом get_post_weights.

3. Построение дерева на основе полученных данных.

*Визуализация данных в виде дерева связей.* Для того, чтобы можно было в удобном виде смотреть связи между постами и пользователями, не заходя в социальную сеть ВКонтакте, была реализована визуализация дерева, в котором корнем является группа, а ее детьми являются посты. При этом листьями дерева являются пользователи, оставившие комментарии под соответствующими постами. Для пользователей можно получить информацию по имени, фамилии и аватару. При желании можно перейти к дереву, построенному в обратную сторону. То есть теперь корнем является пользователь, детьми пользователя являются группы, под постами которых пользователь оставил комментарии, а листьями являются посты, к которым пользователь оставил комментарии.

Техническая реализация визуализации в виде дерева. Само дерево было визуализировано в виде кластерной дендрограммы. Визуализация была выполнена в формате веб-приложения. Веб-приложение использует html, css и javascript для отрисовки графа. На html-странице был размещен элемент svg, в котором затем отрисовывался граф с помощью скрипта javascript с использованием d3.js. Еще на этапе формирования html-страницы Django передает необходимые данные, которые мы сформировали после получения данных. Далее подбираем различные параметры, такие как ширина и высота дерева. Затем, используя метод stratify библиотеки d3.js, создаем дендрограмму из наших данных. Требуется создать несколько элементов <g> внутри элемента svg. Эти элементы и будут обозначать будущие вершины и ребра дендрограммы. Для того, чтобы разграничить вершины и ребра друг от друга, нам нужно пометить их классами nodes и links соответственно. Также для дендрограммы был использован элемент <g> с классом additionalParentLink, который добавляет ребра для детей других вершин в дендрограмме, так как может быть ситуация, что один пользователь оставил комментарий на многих постах. Затем каждый элемент заполняется соответствующими ему данными и можно задать для элементов дополнительные параметры. Например, можно задать радиус вершин графа или толщину ребер графа.

Также был реализован функционал получения информации о пользователе, который отображен в дереве. Таким образом, можно достаточно легко проанализировать, под какими постами оставил комментарии или лайки конкретный пользователь, и затем получить по нему минимальную информацию. После чего при необходимости можно перейти к пользователю в профиль. На скриншотах можно увидеть, что если выбрана группа НТВ, то будут выбраны все посты в этой группе в определенном промежутке времени. На этом скриншоте выбран период с 20 февраля 2021 года до 20 марта 2021 года. Все эти посты ветвятся от самой группы НТВ, а от постов ветвятся пользователи, которые оставили комментарии под этим постом. Для удобства просмотра пользователи не повторяются, для этого нужно было провести дополнительные ребра от постов к пользователям (рис. 4-6).

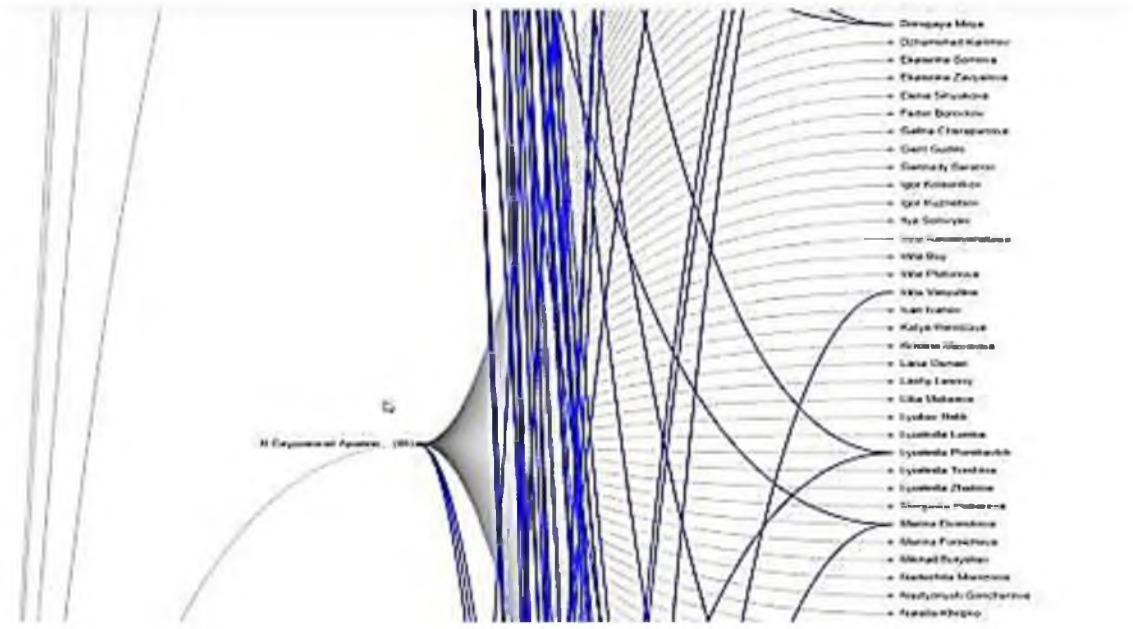

**Рисунок 4.** Визуализация дерева (пост)

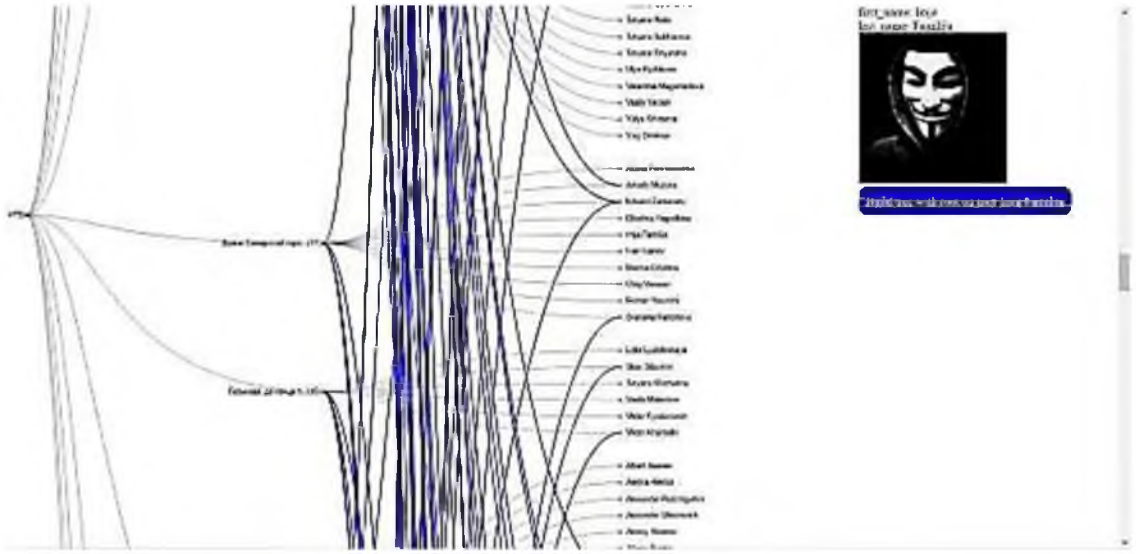

**Рисунок 5.** Визуализация дерева (пользователи и комментарии)

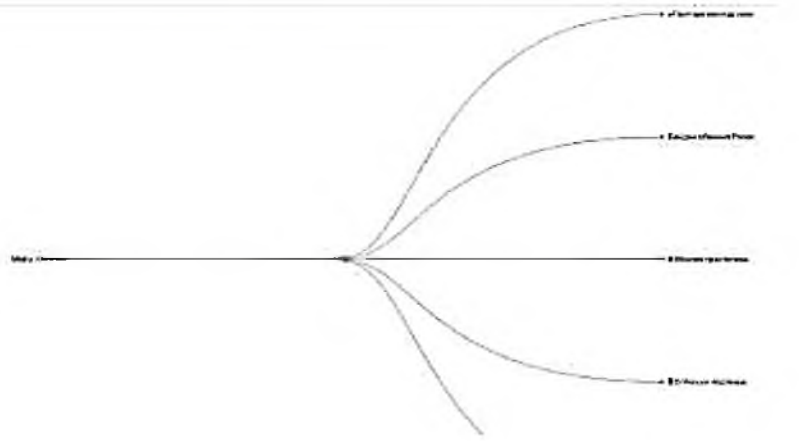

**Рисунок 6.** Визуализация дерева (дополнительные ребра от постов к пользователям)

На скриншоте можно увидеть, как можно воспользоваться функциональностью получения дополнительной информации о пользователе. Например, если нажать на пользователя Imja Familija, который оставил комментарий под постом про врачей Самарской городской больницы, то можно увидеть фото его профиля, по нажатию на которое можно перейти на его страницу в социальной сети. Также можно увидеть кнопку построения дерева от пользователя, где корнем будет пользователь, а посты, под которыми он оставил комментарии, будут от него ветвится. Таким образом, можно более детально разобрать пользователя, который оставил комментарии под многими записями.

*Заключение.* В рамках данной статьи была рассмотрена разработка алгоритма и ПО графической визуализации связей вовлеченных пользователей. Были проанализированы методы анализа социальных сетей и этапы визуализации, построены графы пользователей на основе постов в группах, графы на основе комментариев под постами и дерево постов и пользователей. При анализе графа были посчитаны различные свойства графа, такие как: Degree Centrality, Closeness Centrality, Betweenness Centrality, Chi Squared, количество вершин, количество ребер и плотность графа. Был создан алгоритм визуализации графа.

*Благодарность.* Данное исследование проведено в рамках проекта «Мультиклассификация идеологических направлений киберэкстремизма на казахском языке методами искусственного интеллекта», финансируемого Комитетом науки Министерства науки и высшего образования Республики Казахстан (грант AP19676342, руководитель проекта Мусиралиева Ш.Ж.).

References

1. Last, M. Online Propaganda Detection. In Machine Learning for Data Science Handbook: Data Mining and Knowledge Discovery Handbook (pp. 703-719). Cham: Springer International Publishing. 2023. – doi: https://doi.org/10.1007/978-3-031-24628-9_31.
2. Kumar, P., & Sinha, A. (2021). Information diffusion modeling and analysis for socially interacting networks. Social Network Analysis and Mining, 11, 1-18. – doi: https://doi.org/10.1007/s13278-020-00719-7.
3. Shahid, W., Li, Y., Staples, D., Amin, G., Hakak, S., & Ghorbani, A. (2022). Are you a cyborg, bot or human? – a survey on detecting fake news spreaders. IEEE Access, 10, 27069-27083. – doi: https://doi.org/10.1109/ACCESS.2022.3157724.
4. Sonjaya, S., Yudapratama, A., Aji, A. P., Muhardi, I., & Simbolon, W. B. (2023). Law Enforcement against the Spread of Fake News on Social Media In Relation To Law Number 19 of 2016 Concerning Information and Electronic Transactions. History of Medicine, 9(1), 1297-1307. – doi: https://doi.org/10.33258/polit.v2i1.627.
5. Garcia, L., & Rodriguez, S. (2020). Centrality Measures in Network Analysis: A Comparative Review // Journal of Network Science, 35(3), 256-278.
6. Saber, Ali & Hamid, Noor. (2023). Complex networks analysis: centrality measures // Indonesian Journal of Electrical Engineering and Computer Science. 29. 1642. 10.11591/ijeecs.v29.i3. – Pp. 1642-1647.
7. Chen Hsinchun. Dark Web Exploring and Data Mining the Dark Side of the Web. // Springer-Verlag New York. – 2012. – Vol. 30. – Pp. 286-296.
8. Taha K., Yoo P.D. Shortlisting the influential members of criminal organizations and identifying their important communication channels // IEEE Transactions on Information Forensics and Security. – 2019. – Vol. 14. No. 8. – Pp. 1988-1999.
9. Qunwei L., Kailkhura B., Thiagarajan J., Zhang Zh., Varshney P.K. Influential Node Detection in Implicit Social Networks using Multi-task Gaussian Copula Models // Journal of Machine Learning Research. – 2016. – No. 55. – Pp. 27-37.
10. Kirichenko L., Radivilova T., Carlsson A. Detecting cyber threats through social network analysis: short survey // SocioEconomic Challenges. – 2017. – Vol. 1. P.20-34.
11. Nouh M., Nurse J. Identifying Key−Players in Online Activist Groups on Facebook Social Network // IEEE Computer Society. – 2015. – Pp. 969-978.
12. Sánchez-Rebollo C., Puente A.C., Palacios R., Piriz C., Fuentes B., Juan P., Jarauta J. Detection of Jihadism in Social Networks Using Big Data Techniques Supported by Graphs and Fuzzy Clustering // Complexity. – 2019. – Pp. 1-13.
13. Krebs's 9/11 dataset. – URL: https://sites.google.com/site/ucinetsoftware/datasets/covert-networks/911hijackers (дата доступа:12.12.2022).
14. Caviar dataset. – URL: https://sites.google.com/site/ucinetsoftware/datasets/covert-networks/caviar (дата доступа:12.12.2022).
15. How ISIS Uses Twitter dataset. – URL: https://www.kaggle.com/fifthtribe/how-isis-uses-twitter (дата доступа:12.12.2022).
16. Taha K., "Disjoint community detection in networks based on the relative association of members". IEEE Transactions on Computational Social Systems, 2018. – Pp. 1-15.
17. Sánchez-Rebollo C., Puente A.C., Palacios R., Piriz C., Fuentes B., Juan P., Jarauta J., "Detection of Jihadism in social networks using Big Data Techniques supported by Graphs and Fuzzy Clustering". Complexity, 2019. – Pp. 1-13.
18. Kukkala V., Iyengar S.R.S., "Identifying influential spreaders in a social network (While Preserving Privacy)". Proceedings on Privacy Enhancing Technologies, 2020. – Pp. 537-557.